Transfer learning and Domain adaptation
Transfer from ImageNet (source)

**Transfer as generic features**
Brut Deep features (learned from ImageNet)

(== a learned embedding from Image to vector representation)

Retrieval

**Transfer learning (from source to target)**
Frozen features + SVM => solution to small datasets
Frozen features + Deep
Fine tuning not easy in that case (small datasets)
Transfer from source (=ImageNet task) to target task

**Source:** ImageNet (dataset + 100 classes) => AlexNet trained

**Target:** new dataset Cal-101 and new classification task with 101 classes => Chopped

AlexNet (layer i) + SVM trained on

=> Results better than SoA CV methods on Cal-101!
Transfer: fine-tuning of a deep model on target task

Train a deep (AlexNet) on source (ImageNet)
Keep the deep params. for target and complete with a small deep on top (fully trained on target task)
Fine-tune the whole model on target data
   Challenge: only limited target data, careful about overfitting
   Solution: Freeze the gradient’s update for AlexNet part

Source

Original Task

Pretrained Model

New Task

Copy

Fixed Weights

Target

New Classifier

Gradient Blocked
Transfer: fine-tuning of a deep model on target task

Train a deep (AlexNet) on source (ImageNet)
Keep the deep params. for target and complete with a small deep on top (fully trained on target task)
Fine-tune the whole model on target data
  Challenge: only limited target data, careful about overfitting
  Solution: Freeze the gradient’s update for AlexNet part
Other solution: use smaller gradient’s update for AlexNet part
Transfer: which parts of the deep?

Which layer(s) can be transferred (copied)?

- Speech: usually copy the last few layers
- Image: usually copy the first few layers
Transfer: which supervision?

• Task description
  • Source data: \((x^s, y^s)\) \(\quad\) A large amount
  • Target data: \((x^t, y^t)\) \(\quad\) (Very) little

Rq: Few/One-shot learning: only a few/one examples in target domain

Many different contexts:

In vision: from large dataset (ImageNet) to small datasets (VOC2007)

In speech: (supervised) speaker adaption
  • Source data: audio data and transcriptions from many speakers
  • Target data: audio data and its transcriptions of specific user
More on transfer framework

Source Data (ImageNet)

Target Data

<table>
<thead>
<tr>
<th>labelled</th>
<th>Frozen or fine-tuning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Few</td>
<td>One</td>
</tr>
<tr>
<td>Zero</td>
<td></td>
</tr>
</tbody>
</table>

Main purposes:
Similar visual domain?
Same tasks (i.e., class)?
Similar domain: ImageNet task => Dog/Cat task

Target: Dog/Cat Classifier

Data \textit{not directly related to} the task considered

ImageNet: Similar domain, different task (1000 classes but NOT Dog and Cat classes)
General Framework for Transfer Learning

Target: Dog/Cat Classifier

Data *not directly related to* the task considered

Similar domain, completely different tasks

Different domains, same task
General Framework for Transfer Learning

<table>
<thead>
<tr>
<th>Target Data</th>
<th>Source Data (not directly related to the task)</th>
</tr>
</thead>
<tbody>
<tr>
<td>labelled</td>
<td>labelled</td>
</tr>
<tr>
<td></td>
<td>Fine-tuning</td>
</tr>
<tr>
<td></td>
<td>Multitask Learning</td>
</tr>
<tr>
<td></td>
<td>Domain-adversarial training</td>
</tr>
<tr>
<td></td>
<td>Zero-shot learning</td>
</tr>
<tr>
<td>unlabeled</td>
<td>Self-supervised</td>
</tr>
<tr>
<td></td>
<td>Self-taught learning</td>
</tr>
<tr>
<td></td>
<td>Not considered here</td>
</tr>
<tr>
<td></td>
<td>Self-taught Clustering</td>
</tr>
</tbody>
</table>
# General Framework for Transfer Learning

<table>
<thead>
<tr>
<th>Target Data</th>
<th>Source Data (not directly related to the task)</th>
</tr>
</thead>
<tbody>
<tr>
<td>labelled</td>
<td>Fine-tuning</td>
</tr>
<tr>
<td></td>
<td><strong>Multitask Learning</strong></td>
</tr>
<tr>
<td></td>
<td>Not considered here</td>
</tr>
<tr>
<td>labelled</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Not considered here</td>
</tr>
<tr>
<td>labeled</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Not considered here</td>
</tr>
<tr>
<td>labeling</td>
<td></td>
</tr>
<tr>
<td>labeled</td>
<td></td>
</tr>
<tr>
<td>unlabeled</td>
<td></td>
</tr>
<tr>
<td>labeled</td>
<td></td>
</tr>
<tr>
<td>unlabeled</td>
<td></td>
</tr>
</tbody>
</table>

- **Fine-tuning**
- **Multitask Learning**
Multitask Learning

• The multi-layer structure makes NN suitable for multitask learning

![Diagram showing multitask learning with input features and tasks A and B.]
## Transfer Learning - Overview

<table>
<thead>
<tr>
<th>Source Data (not directly related to the task)</th>
<th>labelled</th>
<th>unlabeled</th>
</tr>
</thead>
<tbody>
<tr>
<td>labelled</td>
<td>Fine-tuning</td>
<td></td>
</tr>
<tr>
<td>Multitask Learning</td>
<td>Not considered here</td>
<td></td>
</tr>
<tr>
<td>Target Data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>labelled</td>
<td>Domain adaptation-adversarial training</td>
<td>Not considered here</td>
</tr>
<tr>
<td>unlabeled</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Unsupervised Domain Adaptation (UDA)

Source data: \((x^s, y^s)\) → Training data
Target data: \((x^t)\)

\{ Same task, domain mismatch \}

MNIST

**SOURCE**

**TARGET**

with labels
without labels

Final test on target domain!
Unsupervised Domain adaptation (UDA): objectives

Main principle: diminish the domain shift in the learned features, encourage domain confusion
UDA strategy: align both domains
UDA strategy: 1/ domain-adversarial training

Add to the feature generator (G) a domain classifier (discriminant D) for which labels are available!

Learn G and D:
- G tries to align domains
- D tries to identify domains

Rq: Similar to GAN (coming soon)
UDA strategy: 1/ domain-adversarial training
2/ classification task (same for source and target here)

Maximize label classification accuracy + minimize domain classification accuracy

Not only cheat the domain classifier, but satisfying label classifier at the same time

Maximize label classification accuracy

Maximize domain classification accuracy

feature extractor

input $x$

Label predictor

Domain classifier

Source only!
UDA strategy: joint learning

Domain classifier fails in the end
It should struggle ……

Yaroslav Ganin, Victor Lempitsky, Unsupervised Domain Adaptation by Backpropagation, ICML, 2015

Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, Domain-Adversarial Training of Neural Networks, JMLR, 2016
# Domain-adversarial training

<table>
<thead>
<tr>
<th>Method</th>
<th>Source Only</th>
<th>SA (Fernando et al., 2013)</th>
<th>Proposed Approach</th>
<th>Train on Target</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MNIST</td>
<td>SYN Numbers</td>
<td>SVHN</td>
<td>SYN Signs</td>
</tr>
<tr>
<td>Source</td>
<td>Target</td>
<td>Source</td>
<td>Target</td>
<td>Source</td>
</tr>
<tr>
<td>MNIST</td>
<td>MNIST-M</td>
<td>.5749</td>
<td>.6078 (7.9%)</td>
<td>.8149 (57.9%)</td>
</tr>
<tr>
<td>SYN Numbers</td>
<td>SVHN</td>
<td>.8665</td>
<td>.8672 (1.3%)</td>
<td>.9048 (66.1%)</td>
</tr>
<tr>
<td>SVHN</td>
<td>MNIST</td>
<td>.5919</td>
<td>.6157 (5.9%)</td>
<td>.7107 (29.3%)</td>
</tr>
<tr>
<td>SYN Signs</td>
<td>GTSRB</td>
<td>.7400</td>
<td>.7635 (9.1%)</td>
<td>.8866 (56.7%)</td>
</tr>
</tbody>
</table>

Yaroslav Ganin, Victor Lempitsky, Unsupervised Domain Adaptation by Backpropagation, ICML, 2015

Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, Domain-Adversarial Training of Neural Networks, JMLR, 2016
Domain adaptation

Main principle: diminish the domain shift in the learned features, encourage domain confusion

Another example: Adversarial Discriminative Domain Adaptation [Tzeng et al. 2017]
Domain adaptation

Other architecture
Domain adaptation

Other architecture: Image translation for Domain adaptation [Murez 2017]
Domain adaptation

Other architecture: Image translation for Domain adaptation [Murez 2017]
Use-Case: Domain adaptation for Autonomous driving
Context: Neural network-based autonomous driving system framework

Inputs:
- Local history
- Point clouds
- RGB Video
- Object detections
- Semantic segmentations
- Depth maps
- Bird-eye-view

Learning:
- Imitation learning with a dataset
- Reinforcement learning with a simulator

Outputs:
- Vehicle controls
- Future trajectory

Sensors:
- Camera
- RADAR
- LiDAR
- IMU
- GPS
Challenges for perception

Multi-sensor perception
• Sensor fusion; Camera, radar and Lidar

3D dynamic understanding
• 3D object detection; Motion forecast; Intention prediction

Frugal learning
• Training with limited data or supervision; Domain adaptation

Reliability
• Robustness; Uncertainty estimation; Failure prediction

Explainability
• Decision interpretation; Post-hoc or by-design
Domain gap

Different, though *related* input data distributions

Source domain $\rightarrow$ Target domain

- Different weather, light, location, sensor’s spec/setup
Domain gap

Different, though related input data distributions

Source domain → Target domain

- Different weather, light, location, sensor’s spec/setup
Domain gap

Different, though *related* input data distributions

Source domain → Target domain

- Different weather, light, location, sensor’s spec/setup
Domain gap

Different, though *related* input data distributions

Source domain → Target domain

- Different weather, light, location, sensor’s spec/setup
Different, though *related* input data distributions

Source domain $\rightarrow$ Target domain

- Synthetic vs. real
Domain gap for **VISUAL SEGMENTATION**

Different, though *related* input data distributions

Source domain → Target domain

- Synthetic vs. real
Domain gap

Different, though *related* input data distributions

Source domain → Target domain

- Synthetic vs. real
Unsupervised Domain Adaptation (UDA)

Labelled source domain data

Unlabelled target domain data
Unsupervised Domain Adaptation (UDA)

**TRAIN**

Source labelled data

**TEST**

Source

Target

learned segmentation model
Unsupervised Domain Adaptation (UDA)

Source labelled data

<table>
<thead>
<tr>
<th>TRAIN</th>
<th>TEST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>Target</td>
</tr>
</tbody>
</table>

learned segmentation model
Unsupervised Domain Adaptation (UDA)

**TRAIN**

Source labelled data

**TEST**

Source

Target

learned segmentation model
Unsupervised Domain Adaptation (UDA)

**TRAIN**
- Source labelled data

**TEST**
- Source
- Target

*learned segmentation model*
Expected results with UDA training

<table>
<thead>
<tr>
<th>TRAIN</th>
<th>TEST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source labelled data</td>
<td></td>
</tr>
<tr>
<td><img src="source_labelled_data.png" alt="Image" /></td>
<td><img src="target.png" alt="Image" /></td>
</tr>
<tr>
<td><img src="learned_segmentation_model.png" alt="Image" /></td>
<td><img src="source_test.png" alt="Image" /></td>
</tr>
</tbody>
</table>
Unsupervised Domain Adaptation (UDA)
Qualitative results

input image       without UDA       with UDA
UDA Results (with Adversarial Entropy)

FCNs in the wild [Hoffman et al., CVPR’16] 27.1
CyCADA [Hoffman et al. JMRL’17] 34.8
Curriculum learning [Zhang et al., ICCV’17] 29.0
Adversarial Adaptation on output [Tsai et al. CVPR’18] 42.4
Ours 44.8

GTA5 → Cityscapes
Extension: Zero shot + Domain adaptation

## Transfer Learning - Overview

<table>
<thead>
<tr>
<th>Target Data</th>
<th>Source Data (not directly related to the task)</th>
</tr>
</thead>
<tbody>
<tr>
<td>labelled</td>
<td>labelled</td>
</tr>
<tr>
<td></td>
<td>Fine-tuning</td>
</tr>
<tr>
<td></td>
<td><em>Multitask Learning</em></td>
</tr>
<tr>
<td></td>
<td>Domain adaptation-adversarial training</td>
</tr>
<tr>
<td></td>
<td><em>Zero-shot learning</em></td>
</tr>
<tr>
<td></td>
<td>Not considered here</td>
</tr>
<tr>
<td></td>
<td>Not considered here</td>
</tr>
<tr>
<td>unlabeled</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Zero-shot Learning

- Source data: \((x^s, y^s)\) \rightarrow Training data
- Target data: \((\emptyset)\) usually same domain

Different tasks

Training time:
\[
\begin{align*}
    x^s: & \quad \text{cat} \\
    y^s: & \quad \text{dog} \\
\end{align*}
\]

Test time \(x^t\):
\[
\text{=> Fish class!}
\]
Zero-shot Learning

- Representing each class by its attributes

**Training**

```
Class: 
1 0 0 1 1 1
furry 4 legs tail furry 4 legs tail
```

+ Database attributes

<table>
<thead>
<tr>
<th></th>
<th>furry</th>
<th>4 legs</th>
<th>tail</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dog</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>Fish</td>
<td>X</td>
<td>X</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>Chimp</td>
<td>O</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

... sufficient attributes for one to one mapping
Zero-shot Learning

- Representing each class by its attributes

**Testing**

Find the class with the most similar attributes

- Attributes:
  
<table>
<thead>
<tr>
<th>class</th>
<th>furry</th>
<th>4 legs</th>
<th>tail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dog</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>Fish</td>
<td>X</td>
<td>X</td>
<td>O</td>
</tr>
<tr>
<td>Chimp</td>
<td>O</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

sufficient attributes for one to one mapping
Zero-shot Learning

- Attribute embedding + class (word name) embedding
Zero-shot Learning

- Attribute embedding

\[ f(*) \] and \[ g(*) \] can be NN.

Training target:

\[ f(x^n) \] and \[ g(y^n) \] as close as possible

\[ f(x^1) \] \[ g(y^1) \]  
\[ f(x^2) \] \[ g(y^2) \]  
\[ f(x^3) \] \[ g(y^3) \]

\( y^i \) are linked together by a class relationship (e.g. class name embedding as W2v)