Domain Adaptation

CS 330
Course Reminders

Optional homework 4 due next Monday.

Project milestone due next Wednesday

Azure: If you are close to running out of credits, proactively request more in private Ed post.
Plan for Today

**Domain Adaptation**
- Problem statements
- Algorithms
  - Data reweighting
  - Feature alignment
  - Domain translation

**Goal for by the end of lecture:** Understand different domain adaptation methods and when to use one vs. another
Problem Settings Recap

**Multi-Task Learning**
Solve multiple tasks $\mathcal{T}_1, \ldots, \mathcal{T}_T$ at once.

$$\min_\theta \sum_{i=1}^T \mathcal{L}_i(\theta, \mathcal{D}_i)$$

**Transfer Learning**
Solve target task $\mathcal{T}_b$ after solving source task(s) $\mathcal{T}_a$ by *transferring* knowledge learned from $\mathcal{T}_a$

**Meta-Learning Problem**
Transfer Learning with Many Source Tasks
Given data from $\mathcal{T}_1, \ldots, \mathcal{T}_n$, solve new task $\mathcal{T}_{\text{test}}$ more quickly / proficiently / stably
What is domain adaptation?

Perform well on target domain $p_T(x, y)$,
using training data from source domain(s) $p_S(x, y)$

A form of transfer learning, with access to target domain data during training ("transductive” learning)

Unsupervised domain adaptation: access to unlabeled target domain data
Semi-supervised domain adaptation: access to unlabeled and labeled target domain data
Supervised domain adaptation: access to labeled target domain data.

We will focus on unsupervised domain adaptation.
What is domain adaptation?

Perform well on target domain $p_T(x, y)$, using training data from source domain(s) $p_S(x, y)$

A form of **transfer learning**, with access to target domain data during training ("transductive" learning)

**Unsupervised domain adaptation**: access to unlabeled target domain data

Common assumptions:
- **Source** and **target** domain only differ in domain of the function, i.e. $p_S(y | x) = p_T(y | x)$
- There exists a single hypothesis with low error.

A “domain” is a special case of a “task”

A task: $\mathcal{T}_i \triangleq \{p_i(x), p_i(y | x), \mathcal{L}_i\}$  
A domain: $d_i \triangleq \{p_i(x), p(y | x), \mathcal{L}\}$
Example domain adaptation problems

Tumor detection & classification
Source hospital  Target hospital
varying imaging techniques, different demographics

Land use classification
Source region  Target region
appearance of buildings, plants; weather conditions, pollution

Text classification, generation
Source corpus  Target corpus
Simple English WIKIPEDIA
differring sentence structure, vocabulary, word use

Domains can also be:
- people/users
- points in time
- institutions
  (schools, companies, universities)

Revisiting assumptions:
- Access to target domain data during training.
- There exists a single hypothesis $f(y | x)$ with low error.
Plan for Today

Domain Adaptation
- Problem statements
- Algorithms
  - Data reweighting
    - Feature alignment
    - Domain translation

Goal for by the end of lecture: Understand different domain adaptation methods and when to use one vs. another
Toy domain adaptation problem

Problem:

Classifier trained on $p_S(x)$ pays little attention to examples with high probability under $p_T(s)$

How can we learn a classifier that does well on $p_T(x)$?

(using labeled data from $p_S(x)$ & unlabeled data from $p_T(x)$)

e.g. sample selection bias

Problem adapted from Blitzer & Daume ICML ‘10
Toy domain adaptation problem

Problem: Classifier trained on $p_S(x)$ pays little attention to examples with high probability under $p_T(s)$

Solution: Upweight examples with high $p_T(x)$ but low $p_S(x)$

Why does this make sense mathematically?

Problem adapted from Blitzer & Daume ICML ‘10
Domain adaptation via importance sampling

Empirical risk minimization on source data: \( \min_{\theta} \mathbb{E}_{p_{S(x,y)}}[L(f_\theta(x), y)] \)

Goal: ERM on target distribution: \( \min_{\theta} \mathbb{E}_{p_{T(x,y)}}[L(f_\theta(x), y)] \)

\[
\mathbb{E}_{p_{T(x,y)}}[L(f_\theta(x), y)] = \int p_{T(x,y)}L(f_\theta(x), y)dx\,dy
\]

\[
= \int p_{T(x,y)}\frac{p_{S(x,y)}}{p_{S(x,y)}}L(f_\theta(x), y)dx\,dy
\]

\[
= \mathbb{E}_{p_{S(x,y)}}\left[\frac{p_{T(x,y)}}{p_{S(x,y)}}L(f_\theta(x), y)\right]
\]

Note: \( p(y|x) \) cancels out if it is the same for source & target

Solution: Upweight examples with high \( p_{T(x)} \) but low \( p_{S(x)} \)
Domain adaptation via importance sampling

$$\min_{\theta} \mathbb{E}_{p_S(x,y)} \left[ \frac{p_T(x)}{p_S(x)} L(f_\theta(x), y) \right]$$

How to estimate the importance weights $\frac{p_T(x)}{p_S(x)}$?

Option 1: Estimate likelihoods $p_T(x)$ and $p_S(x)$, then divide. But, difficult to estimate accurately.

Can we estimate the ratio without training a generative model?

Bayes rule:

$$p(x \mid \text{target}) = \frac{p(\text{target} \mid x)p(x)}{p(\text{target})}$$

$$p(x \mid \text{source}) = \frac{p(\text{source} \mid x)p(x)}{p(\text{source})}$$

$$\frac{p_T(x)}{p_S(x)} = \frac{p(x \mid \text{target})}{p(x \mid \text{source})} = \frac{p(\text{target} \mid x)p(\text{source})}{p(\text{source} \mid x)p(\text{target})}$$

Can estimate with a constant binary classifier!

12

Bickel, Bruckner, Scheffer. Discriminative Learning Under Covariate Shift. JMLR ‘09
Domain adaptation via importance sampling

\[
\min_{\theta} \mathbb{E}_{p_S(x,y)} \left[ \frac{p_T(x)}{p_S(x)} L(f_\theta(x), y) \right]
\]

\[
\frac{p_T(x)}{p_S(x)} = \frac{p(x \mid \text{target})}{p(x \mid \text{source})} = \frac{p(\text{target} \mid x)p(\text{source})}{p(\text{source} \mid x)p(\text{target})}
\]

Full algorithm:

1. Train binary classifier \( c(\text{source} \mid x) \) to discriminate between source and target data.

2. Reweight or resample data \( \mathcal{D}_S \) according to \( \frac{1 - c(\text{source} \mid x)}{c(\text{source} \mid x)} \).

3. Optimize loss \( L(f_\theta(x), y) \) on reweighted or resampled data.

Bickel, Bruckner, Scheffer. Discriminative Learning Under Covariate Shift. JMLR ’09
What assumption does this make?

\[
\min_{\theta} \mathbb{E}_{p_{S}(x,y)} \left[ \frac{p_T(x)}{p_S(x)} L(f_{\theta}(x), y) \right]
\]

Source \( p_S(x) \) needs to cover the target \( p_T(x) \).

Formally: if \( p_T(x) \neq 0 \), then \( p_S(x) \neq 0 \).

Text classification, generation

Source corpus | Target corpus

—> May have enough coverage of distr.

Tumor detection & classification

Source hospital | Target hospital

—> Source probably won’t cover target distr!
Plan for Today

Domain Adaptation
- Problem statements
- Algorithms
  - Data reweighting
- Feature alignment
  - Domain translation

Goal for by the end of lecture: Understand different domain adaptation methods and when to use one vs. another
Domain adaptation if support is not shared?

Can we align the features?

Source classifier in *aligned feature space* is more accurate in target domain.

How to align the features?
Domain adaptation if support is not shared?

How to align the features?

Source encoder $f_{θ_S}$  
Target encoder $f_{θ_T}$

Need to match features at population-level.

i.e. make encoded samples $f_{θ_S}(x), x \sim p_S(\cdot)$ indistinguishable from $f_{θ_T}(x), x \sim p_T(\cdot)$

Key idea: Try to fool a domain classifier $c(d = \text{source} | f(x))$.

If samples are indistinguishable to discriminator, then distributions are the same.
Domain adaptation via feature alignment

**Key idea:** Try to fool a domain classifier $c(d = \text{source} \mid f(x))$.

Minimize label prediction error & maximize “domain confusion”

Ganin et al. Domain-Adversarial Training of Neural Networks. JMLR '16
Domain adaptation via feature alignment

Full algorithm:

1. Randomly initialize encoder(s) $f_{\theta}$, label classifier $g_{\theta_y}$, domain classifier $c_{\phi}$
2. Update domain classifier: $\min_{\phi} \mathcal{L}_c = - \mathbb{E}_{x \sim D_s} [\log c_{\phi}(f(x))] - \mathbb{E}_{x \sim D_t} [1 - \log c_{\phi}(f(x))]$
3. Update label classifier & encoder: $\min_{\theta, \theta_y} \mathbb{E}_{(x,y) \sim D_s} [L(g_{\theta_y}(f_{\theta}(x)), y)] - \lambda \mathcal{L}_c$
4. Repeat steps 2 & 3.

Ganin et al. Domain-Adversarial Training of Neural Networks. JMLR ’16
Domain adaptation via feature alignment

\[ \frac{dL_y}{d\theta} \]

\[ \frac{dL_y}{d\theta} \]

\[ g_{\theta_y}(y | f(x)) \]

\[ c_{\phi}(d = \text{source} | f(x)) \]

“gradient reversal”

Can learn separate source and target encoder

Source encoder \( f_{\theta_S} \)  Target encoder \( f_{\theta_T} \)

Make encoded samples \( f_{\theta_S}(x), x \sim p_S(\cdot) \)

indistinguishable from \( f_{\theta_T}(x), x \sim p_T(\cdot) \)

\( \rightarrow \) can give model more flexibility

Different forms of domain adversarial training.

Option 1: Maximize domain classifier loss (gradient reversal, same as GANs)

Option 2: Optimize for 50/50 guessing


Ganin et al. Domain-Adversarial Training of Neural Networks. JMLR ’16
Domain adaptation via feature alignment

Toy example

source domain: +, —
target domain data: ⋅

standard NN training

domain adversarial training

<table>
<thead>
<tr>
<th>Method</th>
<th>Source Target</th>
<th>MNIST</th>
<th>SYN Numbers</th>
<th>SVHN</th>
<th>SYN Signs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source only</td>
<td>MNIST-M</td>
<td>.5225</td>
<td>.8674</td>
<td>.5490</td>
<td>.7900</td>
</tr>
<tr>
<td>DANN</td>
<td>MNIST</td>
<td>.7666  (52.9%)</td>
<td>.9109 (79.7%)</td>
<td>.7385 (42.6%)</td>
<td>.8865 (46.4%)</td>
</tr>
<tr>
<td>Train on target</td>
<td></td>
<td>.9596</td>
<td>.9220</td>
<td>.9942</td>
<td>.9980</td>
</tr>
</tbody>
</table>

Ganin et al. Domain-Adversarial Training of Neural Networks. JMLR ‘16
Importance weighting

\[
\min_{\theta} \mathbb{E}_{p_S(x,y)} \left[ \frac{p_T(x)}{p_S(x)} L(f_\theta(x), y) \right]
\]

+ simple, can work well
- requires source distr. to cover target

Feature alignment

+ fairly simple to implement, can work quite well
+ doesn’t require source data coverage
- involves adversarial optimization
- requires clear alignment in data
Plan for Today

**Domain Adaptation**
- Problem statements
- Algorithms
  - Data reweighting
  - Feature alignment
- **Domain translation**

**Goal for by the end of lecture:** Understand different domain adaptation methods and when to use one vs. another
What if it is hard to align features?

Idea: translate between domains

i.e. $F : X_S \rightarrow X_T$ or $G : X_T \rightarrow X_S$

If you could translate source examples to target examples:

1. Translate labeled source dataset to target domain with $F'$.
2. Train predictor on translated dataset.
3. Deploy predictor.

Alternatively, if you could translate from target to source:

1. Train predictor on source dataset.
2. Translate target example to source domain with $G$.
3. Evaluate predictor on translated example.

Key question: How to translate between domains?
Domain Translation with CycleGAN

Idea: translate between domains
i.e. $F : X_S \rightarrow X_T$ or $G : X_T \rightarrow X_S$

Key question: How to translate between domains?

**Step 1:** Train $F$ to generate images from $p_T(x)$
and $G$ to generate images from $p_S(x)$

Using GAN objective: $\mathcal{L}_{GAN} = \mathbb{E}_{x \sim p_T} [\log D_T(x)] + \mathbb{E}_{x \sim p_S} [1 - \log D_T(F(x))]$

**Challenge:** The mapping is underconstrained, can be arbitrary.
Can we encourage models to learn a consistent, bijective mapping?

**Step 2:** Train $F$ and $G$ to be cyclically consistent.
\[ F(G(x)) \approx x \text{ and } G(F(x)) \approx x \]

Zhu, Park, Isola, Efros. CycleGAN. ICCV 2017
Domain Translation with CycleGAN

Idea: translate between domains
i.e. \( F : X_S \rightarrow X_T \) or \( G : X_T \rightarrow X_S \)

Step 1: Train \( F \) to generate images from \( p_T(x) \)

and \( G \) to generate images from \( p_S(x) \)

Using GAN objective: \( \mathcal{L}_{GAN} = \mathbb{E}_{x \sim p_T(\cdot)}[\log D_T(x)] + \mathbb{E}_{x \sim p_S(\cdot)}[1 - \log D_T(F(x))] \)

Step 2: Train \( F \) and \( G \) to be cyclically consistent.

\( F(G(x)) \approx x \) and \( G(F(x)) \approx x \)

i.e. \( \mathbb{E}_{x \sim p_S(\cdot)} \| G(F(x)) - x \|_1 + \mathbb{E}_{x \sim p_T(\cdot)} \| F(G(x)) - x \|_1 \)

Full objective: \( \mathcal{L}_{GAN}(F, D_T) + \mathcal{L}_{GAN}(G, D_S) + \lambda \mathcal{L}_{cyc}(F, G) \)

Zhu, Park, Isola, Efros. CycleGAN. ICCV 2017
Domain Translation with CycleGAN

Idea: translate between domains

i.e. $F : X_S \rightarrow X_T$ or $G : X_T \rightarrow X_S$

Zhu, Park, Isola, Efros. CycleGAN. ICCV 2017
CycleGAN for Domain Adaptation

Robotics sim2real policy adaptation

<table>
<thead>
<tr>
<th>Simulation-to-Real Model</th>
<th>Robot 1 Grasp Success</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sim-Only [19]</td>
<td>21%</td>
</tr>
<tr>
<td>Randomized Sim [19]</td>
<td>37%</td>
</tr>
<tr>
<td>GAN</td>
<td>29%</td>
</tr>
<tr>
<td>CycleGAN</td>
<td>61%</td>
</tr>
<tr>
<td>GraspGAN</td>
<td>63%</td>
</tr>
<tr>
<td>RL-CycleGAN</td>
<td>70%</td>
</tr>
</tbody>
</table>
CycleGAN for Domain Adaptation

Human-robot domain adaptation

Input human images

Generated images in robot domain

Smith, Dhawan, Zhang, Abbeel, Levine. RSS 2020
Importance weighting

Feature alignment

Domain translation

+ simple, can work well
  — requires source distr. to cover target
+ fairly simple to implement, can work quite well
  + doesn’t require source coverage
  — involves adversarial optimization
  — requires clear alignment in data
+ conceptually neat, can work quite well
+ interpretable (easier to debug, cool pictures)
-- involves generative modeling & adversarial optimization
-- requires clear alignment in data
CycleGAN & DANN for Domain Adaptation

CyCADA: incorporates both cycle consistency & domain adversarial training

<table>
<thead>
<tr>
<th>Character recognition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
</tr>
<tr>
<td>Source only</td>
</tr>
<tr>
<td>DANN (Ganin et al., 2016)</td>
</tr>
<tr>
<td>DTN (Taigman et al., 2017a)</td>
</tr>
<tr>
<td>CoGAN (Liu &amp; Tuzel, 2016b)</td>
</tr>
<tr>
<td>ADDA (Tzeng et al., 2017)</td>
</tr>
<tr>
<td>PixelDA (Bousmalis et al., 2017)</td>
</tr>
<tr>
<td>UNIT (Liu et al., 2017)</td>
</tr>
<tr>
<td>CyCADA (Ours)</td>
</tr>
<tr>
<td>Target Fully Supervised</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model</th>
<th>Source-only</th>
<th>CyCADA model</th>
<th>Ground truth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test image</td>
<td>Source-only</td>
<td>CyCADA model</td>
<td>Ground truth</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GTA5 → Cityscapes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source only</td>
</tr>
<tr>
<td>FCN-wld (Hoffman et al., 2016)</td>
</tr>
<tr>
<td>CDA (Zhang et al., 2017b)</td>
</tr>
<tr>
<td>FCNTN (Zhang et al., 2017a)</td>
</tr>
<tr>
<td>CyCADA (Ours)</td>
</tr>
<tr>
<td>Oracle - Target Supervised</td>
</tr>
</tbody>
</table>

| Source only | B 42.7 26.3 51.7 5.5 6.8 13.8 23.6 6.9 75.5 11.5 36.8 49.3 0.9 46.7 3.4 5.0 0.0 5.0 1.4 21.7 47.4 62.5 |
| CyCADA (Ours) | B 79.1 33.1 77.9 23.4 17.3 32.1 33.3 31.8 81.5 26.7 69.0 62.8 14.7 74.5 20.9 25.6 6.9 18.8 20.4 39.5 72.4 82.3 |
| Oracle - Target Supervised | B 97.3 79.8 88.6 32.5 48.2 56.3 63.6 73.3 89.0 58.9 93.0 78.2 55.2 92.2 45.0 67.3 39.6 49.9 73.6 67.4 89.6 94.3 |

Table 4: Adaptation between GTA5 and Cityscapes, showing IoU for each class and mean IoU, freq-weighted IoU and pixel accuracy. CyCADA significantly outperforms baselines, nearly closing the gap to the target-trained oracle on pixel accuracy.
Plan for Today

Domain Adaptation
- Problem statements
- Algorithms
  - Data reweighting
  - Feature alignment
  - Domain translation

Goal for by the end of lecture: Understand different domain adaptation methods and when to use one vs. another
Course Reminders

Optional homework 4 due next **Monday**.

Project milestone due next **Wednesday**

**Azure**: If you are close to running out of credits, proactively request more in private Ed post.

*Next time*: Domain generalization

by Huaxiu Yao