Deep Transfer Learning

Mingsheng Long

School of Software
National Engineering Lab for Big Data Software
Tsinghua University

https://github.com/thuml
Outline

1. Deep Transfer Learning

2. Problem 1: $P(X) \neq Q(X)$

3. Problem 2: $P(X, Y) \neq Q(X, Y)$
   - Joint Adaptation Network (JAN)
   - Conditional Domain Adversarial Network (CDAN)

4. Evaluation
Deep Transfer Learning

Deep Learning

Learner: $f : x \rightarrow y$  
Distribution: $(x, y) \sim P(x, y)$

Error Bound: $\epsilon_{\text{test}} \leq \hat{\epsilon}_{\text{train}} + \sqrt{\frac{\text{complexity}}{n}}$
Deep Transfer Learning

- Deep learning across domains of different distributions \( P \neq Q \)

Source Domain

[Image of 2D renderings]

Model

\[ f : x \rightarrow y \]

Target Domain

[Image of real images]

Model

\[ f : x \rightarrow y \]

Deep Transfer Learning: Why?

Training Set | Train-Dev Set | Dev Set | Test Set
---|---|---|---
Training Error high? No | Optimal Bayes Rate Yes | Bias | Deeper Model, Longer Training
Train-Dev Error high? No | Variance | Bigger Data, Regularization
Dev Error high? No | Dataset Shift | Transfer Learning, Data Generation
Test Error high? No | Overfit Dev Set | Bigger Dev Data

Done!

Andrew Ng. The Nuts and Bolts of Building Applications using Deep Learning. NIPS 2016 Tutorial.
Deep Transfer Learning: How?

- Learning predictive models on transferable features s.t. $P(x) = Q(x)$
- Distribution matching: **MMD** (ICML’15), **GAN** (ICML’15, JMLR’16)

![Diagram showing the process of deep transfer learning with examples of features and models.](image)
How Transferable Are Deep Features?

Transferability is restricted by (Yosinski et al. 2014; Glorot et al. 2011)

- **Specialization** of higher layer neurons to original task (new task ↓)
- Optimization difficulty in splitting nets between *co-adapted* neurons
- Disentangling of variations in higher layers enlarges task discrepancy
- Transferability of features decreases while task discrepancy increases

![Graph showing the relationship between layers at which network is chopped and retrained and top-1 accuracy.](https://example.com/graph)

5: Transfer + fine-tuning improves generalization
3: Fine-tuning recovers co-adapted interactions
2: Performance drops due to fragile co-adaptation
4: Performance drops due to representation specificity

![Figure 2: The results from this paper's main experiment.](https://example.com/figure2)
Distribution Mismatch

- Marginal distribution mismatch: $P(\mathbf{X}) \neq Q(\mathbf{X})$
- Conditional distribution mismatch: $P(Y|\mathbf{X}) \neq Q(Y|\mathbf{X})$

\[
P(\mathbf{x}) \neq Q(\mathbf{x}) \quad P(y|x) \neq Q(y|x) \quad P(\mathbf{x}) \approx Q(\mathbf{x}) \quad P(y|x) \approx Q(y|x)
\]
Distribution Matching

- Marginal distribution mismatch: \( P(\mathbf{X}) \neq Q(\mathbf{X}) \)
- Conditional distribution mismatch: \( P(Y|\mathbf{X}) \neq Q(Y|\mathbf{X}) \)

Kernel Embedding

Adversarial Learning

Outline

1. Deep Transfer Learning

2. Problem 1: $P(X) \neq Q(X)$

3. Problem 2: $P(X, Y) \neq Q(X, Y)$
   - Joint Adaptation Network (JAN)
   - Conditional Domain Adversarial Network (CDAN)

4. Evaluation
Deep Adaptation Network (DAN)\(^1\)

Deep adaptation: match distributions in multiple domain-specific layers

Optimal matching: maximize two-sample test power by multiple kernels

\[
d_k^2 (P, Q) \triangleq \left\| \mathbb{E}_P [\phi (x^s)] - \mathbb{E}_Q [\phi (x^t)] \right\|_2^2
\]  \hspace{1cm} (1)

\[
\min_{\theta \in \Theta} \max_{k \in K} \frac{1}{n_a} \sum_{i=1}^{n_a} J (\theta (x^a_i), y^a_i) + \lambda \sum_{\ell=l_1}^{l_2} d_k^2 (D_s^{\ell}, D_t^{\ell})
\]  \hspace{1cm} (2)

\(^1\) Long et al. Learning Transferable Features with Deep Adaptation Networks. ICML '15.
**Problem 1:** \( P(X) \neq Q(X) \)

## Domain Adversarial Neural Network (DANN)

Adversarial adaptation: learning features indistinguishable across domains

\[
E(\theta_f, \theta_y, \theta_d) = \sum_{x_i \in D_s} L_y(G_y(G_f(x_i)), y_i) - \lambda \sum_{x_i \in D_s \cup D_t} L_d(G_d(G_f(x_i)), d_i) \tag{3}
\]

\[
(\hat{\theta}_f, \hat{\theta}_y) = \arg \min_{\theta_f, \theta_y} E(\theta_f, \theta_y, \theta_d) \quad (\hat{\theta}_d) = \arg \max_{\theta_d} E(\theta_f, \theta_y, \theta_d) \tag{4}
\]
Problem 1: \( P(X) \neq Q(X) \)

Residual Transfer Network (RTN)\(^3\)

\[ f_S(x) = f_T(x) + \Delta f(x) \]

Classifier Adaptation

\[ \min f_S = f_T + \Delta f \]

\[ \frac{1}{n_s} \sum_{i=1}^{n_s} L (f_S(x_i^s), y_i^s) \]

\[ + \frac{\gamma}{n_t} \sum_{i=1}^{n_t} H (f_T(x_i^t)) \]

\[ + \lambda D_L (D_s, D_t), \]

Feature Adaptation

\[ X_s^{fcb} \rightarrow X_s^{fcc} \]

M. Long (Tsinghua)

Deep Transfer Learning

March 14, 2018 13 / 32

\(^3\) Long et al. Unsupervised Domain Adaptation with Residual Transfer Networks. NIPS '16.
Problem 2: $P(X, Y) \neq Q(X, Y)$

Outline

1 Deep Transfer Learning

2 Problem 1: $P(X) \neq Q(X)$

3 Problem 2: $P(X, Y) \neq Q(X, Y)$
   - Joint Adaptation Network (JAN)
   - Conditional Domain Adversarial Network (CDAN)

4 Evaluation
## Kernel Embedding of Distributions

<table>
<thead>
<tr>
<th>Distributions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Discrete</strong></td>
</tr>
<tr>
<td>$P(X)$</td>
</tr>
<tr>
<td>$d_x \times 1$</td>
</tr>
<tr>
<td>$P(X, Y)$</td>
</tr>
<tr>
<td>$d_x \times d_y$</td>
</tr>
<tr>
<td>$P(X, Y, Z)$</td>
</tr>
<tr>
<td>$d_x \times d_y \times d_z$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Kernel Embedding</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P(X)$</td>
</tr>
<tr>
<td>$\mu_X := \mathbb{E}_X[\phi(X)]$</td>
</tr>
<tr>
<td>$\infty \times 1$</td>
</tr>
<tr>
<td>$P(X, Y)$</td>
</tr>
<tr>
<td>$C_{XY} := \mathbb{E}_{XY}[\phi(X) \otimes \phi(Y)]$</td>
</tr>
<tr>
<td>$\infty \times \infty$</td>
</tr>
<tr>
<td>$P(X, Y, Z)$</td>
</tr>
<tr>
<td>$C_{XYZ} := \mathbb{E}_{XYZ}[\phi(X) \otimes \phi(Y) \otimes \phi(Z)]$</td>
</tr>
<tr>
<td>$\infty \times \infty \times \infty$</td>
</tr>
</tbody>
</table>

---

Problem 2: $P(X, Y) \neq Q(X, Y)$

Joint Adaptation Network (JAN)

Kernel Embedding of Joint Distributions

\[
C_{YX} = \mathbb{E}[\phi(Y)] \otimes \phi(X) \approx \hat{C}_{YX} = \frac{1}{n} \sum_{i=1}^{n} \phi(y_i) \otimes \phi(x_i)
\]

\[
\mathcal{C}_{X^{1:m}}(P) \triangleq \mathbb{E}_{X^{1:m}} \left[ \otimes_{\ell=1}^{m} \phi^\ell(X^\ell) \right] \approx \hat{\mathcal{C}}_{X^{1:m}} = \frac{1}{n} \sum_{i=1}^{n} \otimes_{\ell=1}^{m} \phi^\ell(x_i^\ell)
\]

Joint Maximum Mean Discrepancy (JMMMD)

Distance between embeddings of \( P(Z^{s_1}, \ldots, Z^{s_{|\mathcal{L}|}}) \) and \( Q(Z^{t_1}, \ldots, Z^{t_{|\mathcal{L}|}}) \)

\[
D_{\mathcal{L}}(P, Q) \triangleq \left\| C_{Z^{s,1:|\mathcal{L}|}}(P) - C_{Z^{t,1:|\mathcal{L}|}}(Q) \right\|_2^{2|\mathcal{L}|} \otimes_{\ell=1}^{H_{\ell}}. 
\]

\[
\hat{D}_{\mathcal{L}}(P, Q) = \frac{1}{n_s^2} \sum_{i=1}^{n_s} \sum_{j=1}^{n_s} \prod_{\ell \in \mathcal{L}} k_{\ell}(z^{s_{\ell}}_i, z^{s_{\ell}}_j) + \frac{1}{n_t^2} \sum_{i=1}^{n_t} \sum_{j=1}^{n_t} \prod_{\ell \in \mathcal{L}} k_{\ell}(z^{t_{\ell}}_i, z^{t_{\ell}}_j) - \frac{2}{n_s n_t} \sum_{i=1}^{n_s} \sum_{j=1}^{n_t} \prod_{\ell \in \mathcal{L}} k_{\ell}(z^{s_{\ell}}_i, z^{t_{\ell}}_j). 
\]

Theorem (Two-Sample Test (Gretton et al. 2012))

- \( P = Q \) if and only if \( \hat{D}_{\mathcal{L}}(P, Q) = 0 \) (In practice, \( \hat{D}_{\mathcal{L}}(P, Q) < \varepsilon \))
How to Understand JMMD?

- Set last-layer features $Z = Z^{L-1}$, classifier predictions $Y = Z^L \in \mathbb{R}^C$
- We can understand JMMD$(Z, Y)$ by simplifying it to linear kernel
- This interpretation assumes classifier predictions $Y$ be one-hot vector

\[
\hat{D}_L (P, Q) \triangleq \left\| \frac{1}{n_s} \sum_{i=1}^{n_s} z_s^i \otimes y_s^i - \frac{1}{n_t} \sum_{j=1}^{n_t} z_t^j \otimes y_t^j \right\|^2
\]

\[
= \sum_{c=1}^C \left\| \frac{1}{n_s} \sum_{i=1}^{n_s} y_{i,c}^s z_i^s - \frac{1}{n_t} \sum_{j=1}^{n_t} y_{j,c}^t z_j^t \right\|^2
\]

\[
\approx \sum_{c=1}^C \hat{D} (P_{Z|Y=c}, Q_{Z|Y=c})
\]

- Equivalent to matching $P$ and $Q$ conditioned on each class
- JMMD process with continuous softmax activations (probability)
Connection to Wasserstein-GAN (WGAN)

Different function spaces, and different powers in comparing distributions

- Wasserstein distance
  \[ D_W (P, Q) \overset{\Delta}{=} \sup_{\|\phi\|_L \leq 1} (\mathbb{E}_{Z^s} [\phi (Z^s)] - \mathbb{E}_{Z^t} [\phi (X^t)]) \] (9)

- MMD
  \[ D_H (P, Q) \overset{\Delta}{=} \sup_{\|\phi\|_H \leq 1} (\mathbb{E}_{Z^s} [\phi (Z^s)] - \mathbb{E}_{Z^t} [\phi (Z^t)]) \] (10)

- Joint MMD (JMMD)
  \[ D_L (P, Q) \overset{\Delta}{=} \sup_{\|\phi^\ell\|_H \leq 1} \left( \mathbb{E}_{Z^s} \left[ \bigotimes_{\ell=1}^{\mathcal{L}} \phi^\ell (Z^{s\ell}) \right] - \mathbb{E}_{Z^t} \left[ \bigotimes_{\ell=1}^{\mathcal{L}} (Z^{t\ell}) \right] \right) \] (11)
Problem 2: $P(X, Y) \neq Q(X, Y)$

Joint Adaptation Network (JAN)

Joint adaptation: match joint distributions of multiple task-specific layers

$$
\min_f \frac{1}{n_s} \sum_{i=1}^{n_s} J(f(x^s_i), y^s_i) + \lambda \hat{D}_\mathcal{L}(P, Q)
$$

$$
D_\mathcal{L}(P, Q) \triangleq \|C_{Z^s,1:|\mathcal{L}|}(P) - C_{Z^t,1:|\mathcal{L}|}(Q)\|_2^2 \otimes_{\ell=1}^{|\mathcal{L}|} \mathcal{H}_\ell
$$

Problem 2: \( P(X, Y) \neq Q(X, Y) \)

Joint Adaptation Network (JAN)

Adversarial Joint Adaptation Network (JAN-A)

Optimal matching: maximize JMMD as semi-parametric domain adversary

\[
\min_f \max_{\theta} \frac{1}{n_s} \sum_{i=1}^{n_s} J(f(x^s_i), y^s_i) + \lambda \hat{D}_L (P, Q; \theta) \quad (14)
\]

\[
\hat{D}_L (P, Q; \theta) = \frac{2}{n} \sum_{i=1}^{n/2} d \left( \{ \theta^\ell (z_{2i-1}^s, z_{2i}^s, z_{2i}^t, z_{2i-1}^t) \}_{\ell \in \mathcal{L}} \right) \quad (15)
\]
Learning Algorithm

Linear-Time $O(n)$ Algorithm of JMMD (Streaming Algorithm)

$$
\hat{D}_L(P, Q) = \frac{2}{n} \sum_{i=1}^{n/2} \left( \prod_{\ell \in \mathcal{L}} k^\ell(z_{2i-1}^{s\ell}, z_{2i}^{s\ell}) + \prod_{\ell \in \mathcal{L}} k^\ell(z_{2i-1}^{t\ell}, z_{2i}^{t\ell}) \right) \\
- \frac{2}{n} \sum_{i=1}^{n/2} \left( \prod_{\ell \in \mathcal{L}} k^\ell(z_{2i-1}^{s\ell}, z_{2i}^{t\ell}) + \prod_{\ell \in \mathcal{L}} k^\ell(z_{2i-1}^{t\ell}, z_{2i}^{s\ell}) \right) \\
= \frac{2}{n} \sum_{i=1}^{n/2} d \left( \{z_{2i-1}^{s\ell}, z_{2i}^{s\ell}, z_{2i-1}^{t\ell}, z_{2i}^{t\ell}\} \right)_{\ell \in \mathcal{L}}
$$

**SGD:** for each layer $\ell$ and for each quad-tuple $(z_{2i-1}^{s\ell}, z_{2i}^{s\ell}, z_{2i-1}^{t\ell}, z_{2i}^{t\ell})$

$$
\nabla W^\ell = \frac{\partial J(z_{2i-1}^{s\ell}, z_{2i}^{s\ell}, y_{2i-1}^{s}, y_{2i}^{s})}{\partial W^\ell} + \lambda \frac{\partial d \left( \{z_{2i-1}^{s\ell}, z_{2i}^{s\ell}, z_{2i-1}^{t\ell}, z_{2i}^{t\ell}\} \right)}{\partial W^\ell}
$$
Problem 2: \( P(X, Y) \neq Q(X, Y) \)

Conditional Domain Adversarial Network (CDAN)

Multilinear Conditioning

\[
\begin{align*}
\min_G E(G) - \lambda E(D, G) \\
\min_D E(D, G)
\end{align*}
\]

\[
E(D, G) = -\frac{1}{n_s} \sum_{i=1}^{n_s} \log(D(f^s_i \otimes g^s_i)) - \frac{1}{n_t} \sum_{j=1}^{n_t} \log(1 - D(f^t_j \otimes g^t_j))
\]

Problem 2: $P(X, Y) \neq Q(X, Y)$

**Conditional Domain Adversarial Network (CDAN)**

**Randomized Multilinear Conditioning**

\[
T \otimes (f, g) = f \otimes g
\]

\[
T \odot (f, g) = \frac{1}{\sqrt{d}} (R_f f) \odot (R_g g)
\]

\[
\phi (h) = \begin{cases} 
T \otimes (f, g) & \text{if } d_f \times d_g \leq 4096 \\
T \odot (f, g) & \text{otherwise}
\end{cases}
\]
Inverse Focal Discriminator

\[ E(D, G) = -\frac{1}{n_s} \sum_{i=1}^{n_s} \exp(D(\phi(h^s_i))) \log(D(\phi(h^s_i))) \]
\[ - \frac{1}{n_t} \sum_{j=1}^{n_t} \exp(1 - D(\phi(h^t_j))) \log(1 - D(\phi(h^t_j))) \]
Optimization Problem

\[
\begin{align*}
\min_G & \quad \frac{1}{n_s} \sum_{i=1}^{n_s} L(G(x^s_i), y^s_i) \\
& \quad + \frac{\lambda}{n_s} \sum_{i=1}^{n_s} \exp(D(\phi(h^s_i))) \log(D(\phi(h^s_i))) \\
& \quad + \frac{\lambda}{n_t} \sum_{j=1}^{n_t} \exp(1 - D(\phi(h^t_j))) \log(1 - D(\phi(h^t_j))) \\
\max_D & \quad \frac{1}{n_s} \sum_{i=1}^{n_s} \exp(D(\phi(h^s_i))) \log(D(\phi(h^s_i))) \\
& \quad + \frac{1}{n_t} \sum_{j=1}^{n_t} \exp(1 - D(\phi(h^t_j))) \log(1 - D(\phi(h^t_j)))
\end{align*}
\]

(24)
Evaluation

Outline

1 Deep Transfer Learning

2 Problem 1:  $P(X) \neq Q(X)$

3 Problem 2:  $P(X, Y) \neq Q(X, Y)$
   - Joint Adaptation Network (JAN)
   - Conditional Domain Adversarial Network (CDAN)

4 Evaluation
Datasets

- VisDA Challenge 2017
- ImageCLEF Challenge 2014
- Office-Caltech
- Spoon, Sink, Mug, Pen, Knife, Bed, Bike, Kettle, TV, Keyboard
- Fine-tune

- Office-Home
- Caffe
- Pre-train
- Real World, Product, Clipart, Art
- Spoon, Sink, Mug, Pen, Knife, Bed, Bike, Kettle, TV, Keyboard, Alarm-Clock, Desk-Lamp, Hammer, Chair, Fan
- Fine-tune

VisDA Challenge 2017
### Results

**Table:** Accuracy (%) on *Office-31* for unsupervised domain adaptation

<table>
<thead>
<tr>
<th>Method</th>
<th>A → W</th>
<th>D → W</th>
<th>W → D</th>
<th>A → D</th>
<th>D → A</th>
<th>W → A</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlexNet</td>
<td>61.6 ± 0.5</td>
<td>95.4 ± 0.3</td>
<td>99.0 ± 0.2</td>
<td>63.8 ± 0.5</td>
<td>51.1 ± 0.6</td>
<td>49.8 ± 0.4</td>
<td>70.1</td>
</tr>
<tr>
<td>TCA</td>
<td>61.0 ± 0.0</td>
<td>93.2 ± 0.0</td>
<td>95.2 ± 0.0</td>
<td>60.8 ± 0.0</td>
<td>51.6 ± 0.0</td>
<td>50.9 ± 0.0</td>
<td>68.8</td>
</tr>
<tr>
<td>GFK</td>
<td>60.4 ± 0.0</td>
<td>95.6 ± 0.0</td>
<td>95.0 ± 0.0</td>
<td>60.6 ± 0.0</td>
<td>52.4 ± 0.0</td>
<td>48.1 ± 0.0</td>
<td>68.7</td>
</tr>
<tr>
<td>DAN</td>
<td>68.5 ± 0.5</td>
<td>96.0 ± 0.3</td>
<td>99.0 ± 0.3</td>
<td>67.0 ± 0.4</td>
<td>54.0 ± 0.5</td>
<td>53.1 ± 0.5</td>
<td>72.9</td>
</tr>
<tr>
<td>RTN</td>
<td>73.3 ± 0.3</td>
<td>96.8 ± 0.2</td>
<td>99.6 ± 0.1</td>
<td>71.0 ± 0.2</td>
<td>50.5 ± 0.3</td>
<td>51.0 ± 0.1</td>
<td>73.7</td>
</tr>
<tr>
<td>DANN</td>
<td>73.0 ± 0.5</td>
<td>96.4 ± 0.3</td>
<td>99.2 ± 0.3</td>
<td>72.3 ± 0.3</td>
<td>53.4 ± 0.4</td>
<td>51.2 ± 0.5</td>
<td>74.3</td>
</tr>
<tr>
<td>ADDA</td>
<td>73.5 ± 0.6</td>
<td>96.2 ± 0.4</td>
<td>98.8 ± 0.4</td>
<td>71.6 ± 0.4</td>
<td>54.6 ± 0.5</td>
<td>53.5 ± 0.6</td>
<td>74.7</td>
</tr>
<tr>
<td>JAN</td>
<td>74.9 ± 0.3</td>
<td>96.6 ± 0.2</td>
<td>99.5 ± 0.2</td>
<td>71.8 ± 0.2</td>
<td><strong>58.3</strong> ± 0.3</td>
<td>55.0 ± 0.4</td>
<td>76.0</td>
</tr>
<tr>
<td><strong>CDAN-RM</strong></td>
<td><strong>77.9</strong> ± 0.3</td>
<td><strong>96.9</strong> ± 0.2</td>
<td><strong>100.0</strong> ± 0.0</td>
<td><strong>74.6</strong> ± 0.2</td>
<td><strong>55.1</strong> ± 0.3</td>
<td><strong>57.5</strong> ± 0.4</td>
<td><strong>77.0</strong></td>
</tr>
<tr>
<td>CDAN-M</td>
<td>77.6 ± 0.2</td>
<td><strong>97.2</strong> ± 0.1</td>
<td><strong>100.0</strong> ± 0.0</td>
<td>73.0 ± 0.1</td>
<td>57.3 ± 0.2</td>
<td>56.1 ± 0.3</td>
<td>76.9</td>
</tr>
<tr>
<td>ResNet-50</td>
<td>68.4 ± 0.2</td>
<td>96.7 ± 0.1</td>
<td>99.3 ± 0.1</td>
<td>68.9 ± 0.2</td>
<td>62.5 ± 0.3</td>
<td>60.7 ± 0.3</td>
<td>76.1</td>
</tr>
<tr>
<td>TCA</td>
<td>72.7 ± 0.0</td>
<td>96.7 ± 0.0</td>
<td>99.6 ± 0.0</td>
<td>74.1 ± 0.0</td>
<td>61.7 ± 0.0</td>
<td>60.9 ± 0.0</td>
<td>77.6</td>
</tr>
<tr>
<td>GFK</td>
<td>72.8 ± 0.0</td>
<td>95.0 ± 0.0</td>
<td>98.2 ± 0.0</td>
<td>74.5 ± 0.0</td>
<td>63.4 ± 0.0</td>
<td>61.0 ± 0.0</td>
<td>77.5</td>
</tr>
<tr>
<td>DAN</td>
<td>80.5 ± 0.4</td>
<td>97.1 ± 0.2</td>
<td>99.6 ± 0.1</td>
<td>78.6 ± 0.2</td>
<td>63.6 ± 0.3</td>
<td>62.8 ± 0.2</td>
<td>80.4</td>
</tr>
<tr>
<td>RTN</td>
<td>84.5 ± 0.2</td>
<td>96.8 ± 0.1</td>
<td>99.4 ± 0.1</td>
<td>77.5 ± 0.3</td>
<td>66.2 ± 0.2</td>
<td>64.8 ± 0.3</td>
<td>81.6</td>
</tr>
<tr>
<td>DANN</td>
<td>82.0 ± 0.4</td>
<td>96.9 ± 0.2</td>
<td>99.1 ± 0.1</td>
<td>79.7 ± 0.4</td>
<td>68.2 ± 0.4</td>
<td>67.4 ± 0.5</td>
<td>82.2</td>
</tr>
<tr>
<td>ADDA</td>
<td>86.2 ± 0.5</td>
<td>96.2 ± 0.3</td>
<td>98.4 ± 0.3</td>
<td>77.8 ± 0.3</td>
<td>69.5 ± 0.4</td>
<td>68.9 ± 0.5</td>
<td>82.9</td>
</tr>
<tr>
<td>JAN</td>
<td>85.4 ± 0.3</td>
<td>97.4 ± 0.2</td>
<td>99.8 ± 0.2</td>
<td>84.7 ± 0.3</td>
<td>68.6 ± 0.3</td>
<td>70.0 ± 0.4</td>
<td>84.3</td>
</tr>
<tr>
<td><strong>CDAN-RM</strong></td>
<td><strong>93.0</strong> ± 0.2</td>
<td><strong>98.4</strong> ± 0.2</td>
<td><strong>100.0</strong> ± 0.0</td>
<td>89.2 ± 0.3</td>
<td>70.2 ± 0.4</td>
<td>69.4 ± 0.4</td>
<td><strong>86.7</strong></td>
</tr>
<tr>
<td>CDAN-M</td>
<td><strong>93.1</strong> ± 0.1</td>
<td><strong>98.6</strong> ± 0.1</td>
<td><strong>100.0</strong> ± 0.0</td>
<td><strong>93.4</strong> ± 0.2</td>
<td><strong>71.0</strong> ± 0.3</td>
<td><strong>70.3</strong> ± 0.3</td>
<td><strong>87.7</strong></td>
</tr>
</tbody>
</table>
**Results**

![VISDA CHALLENGE 2017](image)

- **AlexNet**
  - CNN: 28.7
  - DAN: 51.6
  - RTN: 53
  - DANN: 56.3
  - JAN: 59.5
  - CDAN-M: 64.8

- **ResNet-50**
  - CNN: 43.9
  - DAN: 55
  - RTN: 57.6
  - DANN: 59
  - JAN: 61.1
  - CDAN-M: 64.8

M. Long (Tsinghua) Deep Transfer Learning March 14, 2018
Analysis

Table: Accuracy (%) of CDAN variants for unsupervised domain adaptation

<table>
<thead>
<tr>
<th>Method</th>
<th>A → W</th>
<th>D → W</th>
<th>W → D</th>
<th>A → D</th>
<th>D → A</th>
<th>W → A</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDAN-RM (bernoulli)</td>
<td>87.8 ± 0.3</td>
<td>97.2 ± 0.3</td>
<td>99.4 ± 0.1</td>
<td>85.1 ± 0.4</td>
<td><strong>70.9 ± 0.5</strong></td>
<td>71.7 ± 0.5</td>
<td>85.3</td>
</tr>
<tr>
<td>CDAN-RM (gaussian)</td>
<td>88.0 ± 0.1</td>
<td>97.4 ± 0.1</td>
<td>99.7 ± 0.1</td>
<td>86.4 ± 0.2</td>
<td>70.6 ± 0.3</td>
<td>71.4 ± 0.3</td>
<td>85.6</td>
</tr>
<tr>
<td>CDAN-RM (uniform)</td>
<td><strong>93.0 ± 0.2</strong></td>
<td><strong>98.4 ± 0.2</strong></td>
<td><strong>100.0 ± 0.0</strong></td>
<td><strong>89.2 ± 0.3</strong></td>
<td>70.2 ± 0.4</td>
<td>69.4 ± 0.4</td>
<td><strong>86.7</strong></td>
</tr>
<tr>
<td>CDAN-M (no focal loss)</td>
<td>91.7 ± 0.2</td>
<td>98.3 ± 0.1</td>
<td><strong>100.0 ± 0.0</strong></td>
<td>92.5 ± 0.2</td>
<td>70.0 ± 0.2</td>
<td>67.8 ± 0.2</td>
<td>86.8</td>
</tr>
<tr>
<td>CDAN-M (focal loss)</td>
<td><strong>93.1 ± 0.1</strong></td>
<td><strong>98.6 ± 0.1</strong></td>
<td><strong>100.0 ± 0.0</strong></td>
<td><strong>93.4 ± 0.2</strong></td>
<td><strong>71.0 ± 0.3</strong></td>
<td><strong>70.3 ± 0.3</strong></td>
<td><strong>87.7</strong></td>
</tr>
</tbody>
</table>

(a) Conditioning

(b) Discrepancy

(c) Convergence

Figure: Analysis of CDAN: (a) Conditioning, (b) Discrepancy, (c) Convergence.
Open Problems

- Heterogeneous Transfer Learning
  \[ X_s \neq X_t \lor Y_s \neq Y_t \]

- Pixel-Level Transfer Learning
  \[ P(X) \neq Q(X) \lor P(Z) \neq Q(Z) \]

- Learning Transferable Architectures

- Code available at: https://github.com/thuml/Xlearn