Unsupervised Domain Adaptation with Residual Transfer Networks

Mingsheng Long1, Han Zhu1, Jianmin Wang1, and Michael I. Jordan2

1School of Software, Tsinghua University, China 2Department of EECS, UC Berkeley, USA

Summary

- A residual transfer network for unsupervised domain adaptation
- An end-to-end deep architecture for jointly learning
  - Transferable features with joint distribution adaptation
  - Adaptive classifiers with deep residual learning and entropy minimization
- Open problem
  - More efficient joint distribution adaptation with fast kernel approximation
  - Extending residual transfer network to semi-supervised domain adaptation

Unsupervised Domain Adaptation

- Source domain: \( D_s = \{ (x_i, y_i) \}_{i=1}^{n_s} \) of \( n_s \) labeled examples
- Target domain: \( D_t = \{ (x_i, y_i) \}_{i=1}^{n_t} \) of \( n_t \) unlabeled examples
- Setting: different feature distributions \( P_s(x, y) \neq P_t(x, y) \)
- Challenge: different classification models \( f_s(x) \neq f_t(x) \)
- Problem: bound the target risk \( R_t(\hat{f}) = \mathbb{E}_{(x, y) \sim P_t} [\ell(\hat{f}(x), y)] \neq f_t(x) \)
- Key: jointly learn transferable features and adaptive classifiers such that \( P_t(x, y) \approx P_s(x, y) \) and \( f_s(x) = f_t(x) + \Delta f(x) \)

Deep Learning for Domain Adaptation

- Feature transferability decreases in multiple task-specific layers
- Feature transferability decreases as domain difference increases

Residual Transfer Network (RTN)

- Goal: end-to-end learning of transferable features and classifiers
- Modeling the joint distribution of features in multiple layers
  - Feature fusion via tensor product: \( z^i = \bigotimes_{m \in \mathbb{M}} \phi_m(x_i) \)
  - Maximum Mean Discrepancy (MMD) to compare distributions \( P_s \) and \( P_t \) in RKHS: \( D(P_s, P_t) = ||\mathbb{E}_{x \sim P_s} [\phi(x)] - \mathbb{E}_{x \sim P_t} [\phi(x)]||_2^2 \)
- Feature adaptation via MMD over fused features (tensor MMD)
  \[
  \min_{f, \Delta f} D_{m}(D_s, D_t) = \sum_{i=1}^{n_s} \sum_{j=1}^{n_t} k(z_i^s, z_j^t) + \frac{\gamma}{n_t} \sum_{j=1}^{n_t} H(f_t(x_j)) + \lambda D_{\phi}(D_s, D_t)
  \]

Classifier Adaptation

- Modeling cross-domain classifier shift by deep residual learning
  - Build the connection across classifiers: \( f_s(x) = f_t(x) + \Delta f(x) \)
  - Set residual block: \( x \triangleq f_r(x), F(x) \triangleq f_s(x), \Delta f(x) \triangleq \Delta f(x) \)
  - Probabilistic predictions: \( f_s(x) \triangleq \sigma(f_s(x)), f_t(x) \triangleq \sigma(f_t(x)) \)
  - Low-density separation of target data by entropy minimization
  \[
  \min_{f, \gamma} \frac{1}{n_t} \sum_{j=1}^{n_t} H(f_t(x_j))
  \]
- The class-conditional distribution \( f_t^j(x_j) = p(y_i = j | x_i; f_t) \)
- The entropy function \( H(f_t(x_j)) = -\sum_{j=1}^{c} f_t^j(x_j) \log f_t^j(x_j) \)

Results

- RTN can learn adaptive classifiers and transferable features
- Entropy minimization can make residual transfer more effective

Table: Accuracy on Office-31 dataset for unsupervised domain adaptation

<table>
<thead>
<tr>
<th>Method</th>
<th>A-D</th>
<th>D-A</th>
<th>A-W</th>
<th>W-A</th>
<th>A-C</th>
<th>C-A</th>
<th>TCA</th>
<th>GEF</th>
<th>AlexNet</th>
<th>DDC</th>
<th>DAN</th>
<th>RevGrad</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>95.0</td>
<td>92.1</td>
<td>89.9</td>
<td>91.8</td>
<td>91.0</td>
<td>89.7</td>
<td>91.6</td>
<td>90.8</td>
<td>91.6</td>
<td>91.7</td>
<td>90.1</td>
<td>90.4</td>
</tr>
<tr>
<td>Pre-train</td>
<td>91.4</td>
<td>92.9</td>
<td>91.0</td>
<td>92.1</td>
<td>94.2</td>
<td>94.3</td>
<td>94.3</td>
<td>94.4</td>
<td>94.2</td>
<td>94.3</td>
<td>94.3</td>
<td>94.3</td>
</tr>
<tr>
<td>Fine-tune</td>
<td>95.2</td>
<td>93.2</td>
<td>94.2</td>
<td>93.3</td>
<td>93.5</td>
<td>93.4</td>
<td>93.4</td>
<td>93.4</td>
<td>93.4</td>
<td>93.4</td>
<td>93.4</td>
<td>93.4</td>
</tr>
</tbody>
</table>

Discussion

- (a) DAN: Da→A
- (b) DAN: Dn→W
- (c) RTN: Da→A
- (d) RTN: Dn→W

Figure: Prediction visualization: (a)-(b) t-SNE of DAN; (c)-(d) t-SNE of RTN.