Contents

- Introduction
- Objective
- Related work
- DSN Model
- Evaluation
- Comments and conclusion
Introduction: Domain adaptation

Training:

Source domain $P(X)$  \[\rightarrow\] Model  \[\rightarrow\] 3  \[\rightarrow\] Source domain $P(Y|X)$

Testing:

Target domain $P(X)$  \[\rightarrow\] Model  \[\rightarrow\] ?  \[\rightarrow\] Target domain $P(Y|X)$

Other examples:
- Text prediction from News corpus to science fiction novels
- Spam filtering from user 1 to user 2
Objective

• Motivation:
  • Real world annotated data is difficult to obtain.
  • Synthetic data is readily available
  • To transfer learning ability trained on synthetic data to unlabeled real world test data without compromising performance.

• Idea: Can we learn representations that are domain invariant to transfer knowledge from a labeled source domain to an unlabeled target domain?

Can we do this without compromising performance on the source domain?
Objective

• We will specifically discuss the task of image classification and pose estimation.

• For image data, the source and target distributions can vary in many ways. The differences may arise due to:
  • “Low level” image statistics like resolution, color, illumination etc
  • High level features such as type of objects, 3D pose etc.

• Here we examine datasets where source and target domains differ only in low-level statistics.
Related work

• Should we extract a representation of data that is shared by both domains? (DANN, Ganin et. al)

Source: Ganin, Lampitsky, *Unsupervised Domain Adaptation by Backpropagation*

• Problem: Shared representation is susceptible to noise from the shared domains.
Related work

- Deep Domain confusion: Learn a feature representation that is “closer” in distance compared to the original source and target distribution such that the same classifier applies (MMD loss, Tzeng et. al)

Source: Tzeng et. al., Deep Domain Confusion: Maximizing for Domain Invariance
Related work

- Maximum Mean Discrepancy Metric (MMD) : minimize the distance between two distributions:

  \[ \text{MMD}(X_S, X_T) = \left\| \frac{1}{|X_S|} \sum_{x_s \in X_S} \phi(x_s) - \frac{1}{|X_T|} \sum_{x_t \in X_T} \phi(x_t) \right\| \]

- Modify the loss function as:

  \[ \mathcal{L} = \mathcal{L}_C(X_L, y) + \lambda \text{MMD}^2(X_S, X_T) \]
Related work

• Should we learn a mapping from the source domain representation to the target domain representation? (CORAL, Sun et. al.)

Source: Sun et. al, *Return of Frustratingly Easy Domain Adaptation*
Related work

• CORRelation Alignment (CORAL): Align the second order statistics of the distributions:

\[
\begin{align*}
\min_A \| C_S - C_T \|_F^2 \\
= \min_A \| A^T C_S A - C_T \|_F^2
\end{align*}
\]

\[
A^* = U_S E \\
= (U_S \Sigma_S^{1/2} U_S^T)(U_{T[1:r]} \Sigma_{T[1:r]}^{1/2} U_{T[1:r]}^T).
\]

• Insert CORAL layer between activations in deep architectures.
Domain Separation Networks

- Novelty of this model is that it learns both, a shared feature representation for both domains (using existing techniques) and a domain specific private feature representation (using orthogonality constraint).

- A sophisticated loss function encourages the model to keep there representations independent (orthogonal).

- For the private representations to be useful on their own, a reconstruction loss term is introduced in the loss function.
Model

- Domain Separation Networks (DSNs) jointly model both private and shared components of the domain representation:
  - Private subspace captures domain specific properties such as background and low level image statistics
  - Share subspace captures feature representations shared by both domains.
Model

- Classification output:
  \[ \hat{y} = G(E_c(x)) \]

- Reconstruction output:
  \[ \hat{x} = D(E_c(x) + E_p(x)) \]

- Loss function:
  \[ L = L_{task} + \alpha L_{recon} + \beta L_{difference} + \gamma L_{similarity} \]
Model

• The classification loss trains the model to predict the output labels we are ultimately interested in. Since the target domain is unlabeled, this is only defined for the source domain

\[ L_{task} = \sum_{i=0}^{N} y_i \log \hat{y}_i \]
Model

- For the reconstruction loss, a scale invariant mean squared error is defined:

\[ \mathcal{L}_{\text{recon}} = \sum_{i=1}^{N_s} \mathcal{L}_{\text{si_mse}}(x_i^s, \hat{x}_i^s) + \sum_{i=1}^{N_t} \mathcal{L}_{\text{si_mse}}(x_i^t, \hat{x}_i^t) \]

\[ \mathcal{L}_{\text{si_mse}}(x, \hat{x}) = \frac{1}{k} \|x - \hat{x}\|_2^2 - \frac{1}{k^2} ([x - \hat{x}] \cdot 1_k)^2, \]

where k is the number of pixels.

- While traditional mse penalizes predictions upto a scaling term, the scale invariant mse penalizes differences between pairs of pixels.
Model

• The difference loss encourages the shared and private representations to encode different aspects of the input:

\[ L_{\text{difference}} = \| H_c^s H_p^s \|_F^2 + \| H_c^t H_p^t \|_F^2, \]

• Frobenius norm is a euclidean norm defined for matrices:

\[ \| A \|_F = \sqrt{\text{tr}(AA^H)} \]

• This loss will be zero if the shared and private representations are orthogonal to each other.
Model

- Two similarity losses are evaluated.

- Domain adversarial similarity loss: Confusing a domain classifier using Gradient Reversal Layer (GRL)

\[
\mathcal{L}_{\text{DANN}}^{\text{similarity}} = \sum_{i=0}^{N_s+N_t} \left\{ d_i \log \hat{d}_i + (1 - d_i) \log(1 - \hat{d}_i) \right\}.
\]

\[
x \to h_c = E_c(x; \theta_c) \to d = Z(Q(h_c); \theta_Z)
\]

- Maximum Mean Discrepancy (MMD) loss is a kernel-based distance function between pairs of samples.

\[
\mathcal{L}_{\text{MMD}}^{\text{similarity}} = \frac{1}{(N_s)^2} \sum_{i,j=0}^{N_s} \kappa(h_{ci}^s, h_{cj}^s) - \frac{2}{N_sN_t} \sum_{i,j=0}^{N_s,N_t} \kappa(h_{ci}^s, h_{cj}^t) + \frac{1}{(N_t)^2} \sum_{i,j=0}^{N_t} \kappa(h_{ci}^t, h_{cj}^t),
\]
The model was evaluated for image classification task by training on clean (mostly synthetic) data and testing on noisy real world data.

<table>
<thead>
<tr>
<th>Training</th>
<th>Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>MNIST-M</td>
</tr>
<tr>
<td>Synthetic digits</td>
<td>SVHN</td>
</tr>
<tr>
<td>MNIST</td>
<td>SVHN</td>
</tr>
<tr>
<td>Synthetic traffic signs</td>
<td>GTSRB</td>
</tr>
<tr>
<td>Synthetic objects</td>
<td>LineMod</td>
</tr>
</tbody>
</table>
Evaluation

• How do we evaluate target domain performance? A small set of labeled target data was used as validation set to optimize hyperparameters.

• Although the target domain data is unlabeled during training, labeled test data was used to evaluate the mean classification accuracy.

<table>
<thead>
<tr>
<th>Model</th>
<th>MNIST to MNIST-M</th>
<th>Synth Digits to SVHN</th>
<th>SVHN to MNIST</th>
<th>Synth Signs to GTSRB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source-only</td>
<td>56.6 (52.2)</td>
<td>86.7 (86.7)</td>
<td>59.2 (54.9)</td>
<td>85.1 (79.0)</td>
</tr>
<tr>
<td>CORAL [27]</td>
<td>57.7</td>
<td>85.2</td>
<td>63.1</td>
<td>86.9</td>
</tr>
<tr>
<td>MMD [30, 18]</td>
<td>76.9</td>
<td>88.0</td>
<td>71.1</td>
<td>91.1</td>
</tr>
<tr>
<td>DANN [8]</td>
<td>77.4 (76.6)</td>
<td>90.3 (91.0)</td>
<td>70.7 (73.8)</td>
<td>92.9 (88.6)</td>
</tr>
<tr>
<td>DSN w/ MMD (ours)</td>
<td>80.5</td>
<td>88.5</td>
<td>72.2</td>
<td>92.6</td>
</tr>
<tr>
<td>DSN w/ DANN (ours)</td>
<td>83.2</td>
<td>91.2</td>
<td>82.7</td>
<td>93.1</td>
</tr>
<tr>
<td>Target-only</td>
<td>98.7</td>
<td>92.4</td>
<td>99.5</td>
<td>99.8</td>
</tr>
</tbody>
</table>
For the synthetic objects to LineMod adaptation, the model learns both classification and pose estimation.

$$L_{\text{task}} = \sum_{i=0}^{N_s} \{-y_i^s \cdot \log \hat{y}_i^s + \xi \log(1 - |q_i^s \cdot \hat{q}_i^s|)\},$$

where $q$ is the unit quaternion representing the 3D pose.

<table>
<thead>
<tr>
<th>Method</th>
<th>Classification Accuracy</th>
<th>Mean Angle Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source-only</td>
<td>47.33%</td>
<td>89.2°</td>
</tr>
<tr>
<td>MMD</td>
<td>72.35%</td>
<td>70.62°</td>
</tr>
<tr>
<td>DANN</td>
<td>99.90%</td>
<td>56.58°</td>
</tr>
<tr>
<td>DSN w/ MMD (ours)</td>
<td>99.72%</td>
<td>66.49°</td>
</tr>
<tr>
<td>DSN w/ DANN (ours)</td>
<td>100.00%</td>
<td>53.27°</td>
</tr>
<tr>
<td>Target-only</td>
<td>100.00%</td>
<td>6.47°</td>
</tr>
</tbody>
</table>
Evaluation

• The DSN model with DANN similarity loss outperforms all the other models to obtain superior classification accuracy.

• DSN with DANN similarity performed better than with MMD similarity. This result is consistent with the unsupervised domain adaptation as well.

• The effectiveness of the scale invariant mse compared to the traditional mse and the difference loss are evident in these comparisons.

<table>
<thead>
<tr>
<th>Model</th>
<th>MNIST to MNIST-M</th>
<th>Synth. Digits to SVHN</th>
<th>SVHN to MNIST</th>
<th>Synth. Signs to GTSRB</th>
</tr>
</thead>
<tbody>
<tr>
<td>All terms</td>
<td>83.23</td>
<td>91.22</td>
<td>82.78</td>
<td>93.01</td>
</tr>
<tr>
<td>No $\mathcal{L}_{\text{difference}}$</td>
<td>80.26</td>
<td>89.21</td>
<td>80.54</td>
<td>91.89</td>
</tr>
<tr>
<td>With $\mathcal{L}_{\text{recon}}$</td>
<td>80.42</td>
<td>88.98</td>
<td>79.45</td>
<td>92.11</td>
</tr>
</tbody>
</table>
Evaluation

- Decoding the shared and private representations individually, the reconstructed images are compared:

  - MNIST to MNIST-M: clearly separates the foreground from the background and produces a shared space that is very similar to the source domain.
  - Synth objects to LINEMOD: is able to produce visualizations of the shared representation that look very similar between source and target domains, which are useful for classification and pose estimation.
Evaluation

- Shared only reconstruction (column 3), captures high level features.

- Private only (column 4) reconstruction captures low level image statistics.
Conclusion

• Strengths
  • Sophisticated architecture with well defined functions for various subparts.
  • Carefully crafted loss function in a adversarial training setting allows control over network performance.
  • By jointly learning both shared and private representations, the model provides an “interpretable” feature representation that gives insight into the distribution of data in both domains.

• Weaknesses:
  • Some dependence on labeled data from target domain.
  • Large number of hyperparameters ($\alpha, \beta, \gamma, \delta \ldots$)
• **Follow up:**
  - Interesting to see effectiveness of the model for more and more different distributions (moving up the feature hierarchy).
  
  - Consider one source and multiple target domains: the model can be used as a tool to compare similarity between two unlabeled data distributions/domains as well as the belongingness of a new example to each domain.
  
  - Can be extended to NLP tasks to apply towards Text-to-image synthesis, visual Q&A etc.