Transfer Learning

Generalizing Deep Learning across Domains and Tasks

Mingsheng Long

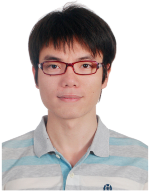
School of Software
National Engineering Lab for Big Data Software
Tsinghua University

https://github.com/thuml

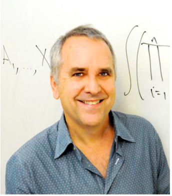
Chinese Conference on Pattern Recognition and Computer Vision
PRCV 2018

Joint Work With:

Jianmin Wang
Professor
Tsinghua University
jimwang@tsinghua.edu.cn

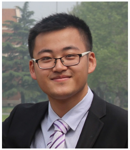


Mingsheng Long
Associate Professor
Tsinghua University
mingsheng@tsinghua.edu.cn



Michael I. Jordan
Professor
UC Berkeley
jordan@cs.berkeley.edu

Yuchen Zhang



Yue Cao

Han Zhu

Zhangjie Cao

Outline

1. Transfer Learning
2. Problem I: $P(\mathbf{X}) \neq Q(\mathbf{X})$
   - DAN: Deep Adaptation Network
3. Problem II: $P(Y|\mathbf{X}) \neq Q(Y|\mathbf{X})$
   - CDAN: Conditional Domain Adversarial Network
4. Theoretical Analysis
5. Benchmarking

Machine Learning

Learner: $f : \boldsymbol{x} \rightarrow y$ **Distribution:** $(\boldsymbol{x}, y) \sim P(\boldsymbol{x}, y)$

fish

bird

mammal

tree

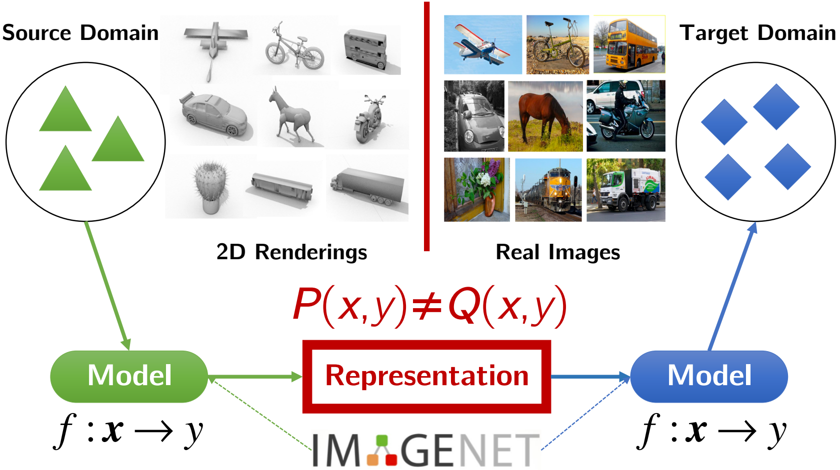
flower

……

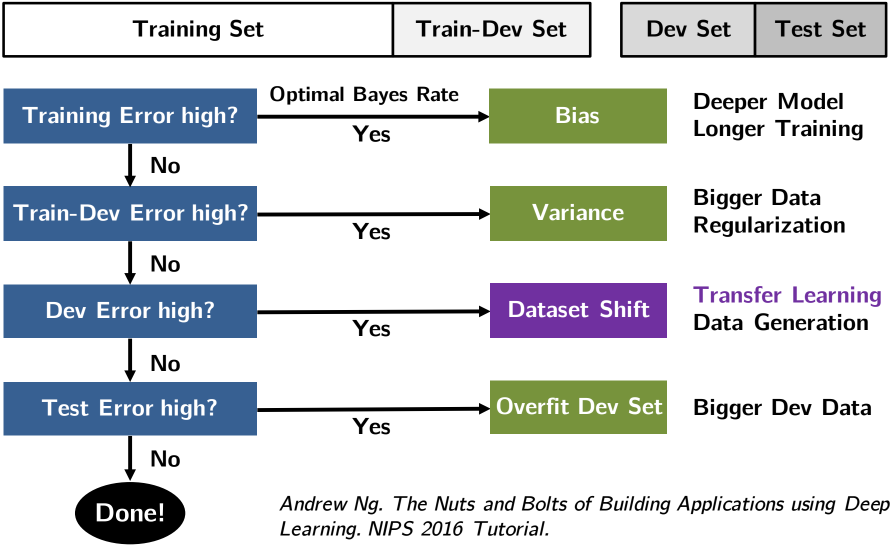
Error Bound: $\epsilon_{\text{test}} \leq \hat{\epsilon}_{\text{train}} + \sqrt{\frac{\text{complexity}}{n}}$

Transfer Learning

- Machine learning across domains of Non-IID distributions $P \neq Q$
- How to design models that effectively bound the generalization error?



Bias-Variance-Shift Tradeoff

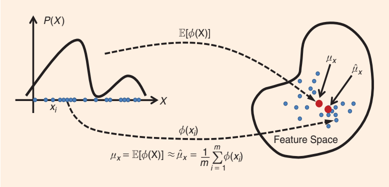


Andrew Ng. The Nuts and Bolts of Building Applications using Deep Learning. NIPS 2016 Tutorial.

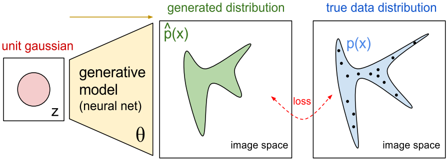
Basic Approaches to Transfer Learning

Matching distributions across source and target domains s.t. $P \approx Q$

- Reduce marginal distribution mismatch: $P(\mathbf{X}) \neq Q(\mathbf{X})$
- Reduce conditional distribution mismatch: $P(Y|\mathbf{X}) \neq Q(Y|\mathbf{X})$
- Challenge: fail to align different domains of multimodal distributions



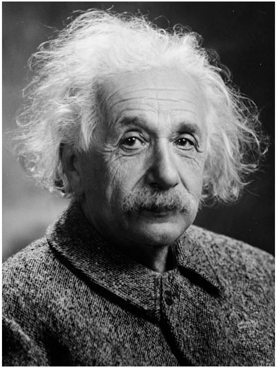
Kernel Embedding



Adversarial Learning

*Song et al. Kernel Embeddings of Conditional Distributions. **IEEE**, 2013.*
*Goodfellow et al. Generative Adversarial Networks. **NIPS** 2014.*

Basic Guidelines to Algorithm Design



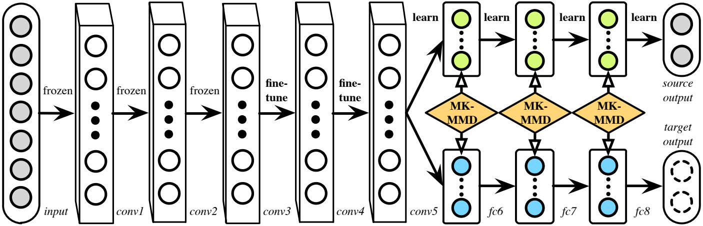
Everything should be made as simple as possible, but no simpler.

—Albert Einstein

Outline

1. Transfer Learning
2. Problem I: $P(\mathbf{X}) \neq Q(\mathbf{X})$
   - DAN: Deep Adaptation Network
3. Problem II: $P(Y|\mathbf{X}) \neq Q(Y|\mathbf{X})$
   - CDAN: Conditional Domain Adversarial Network
4. Theoretical Analysis
5. Benchmarking

DAN: Deep Adaptation Network[1]



Deep adaptation: match distributions in multiple domain-specific layers

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

$$d_k^2(P, Q) \triangleq \left\| \mathbf{E}_P[\phi(\mathbf{x}^s)] - \mathbf{E}_Q\left[\phi\left(\mathbf{x}^t\right)\right] \right\|_{\mathcal{H}_k}^2 \quad (1)$$

$$\min_{\theta \in \Theta} \max_{k \in \mathcal{K}} \frac{1}{n_a} \sum_{i=1}^{n_a} J(\theta(\mathbf{x}_i^a), y_i^a) + \lambda \sum_{\ell=l_1}^{l_2} d_k^2\left(\mathcal{D}_s^\ell, \mathcal{D}_t^\ell\right) \quad (2)$$

[1] *Long et al. Learning Transferable Features with Deep Adaptation Networks. ICML '15.*

DAN: MK-MMD

Multiple Kernel Maximum Mean Discrepancy (MK-MMD)

RKHS distance between *kernel embeddings* of distributions P_X and Q_X

$$d_k^2(P, Q) \triangleq \left\| \mathbf{E}_P \left[\phi\left(\mathbf{x}^s\right)\right] - \mathbf{E}_Q \left[\phi\left(\mathbf{x}^t\right)\right] \right\|_{\mathcal{H}_k}^2, \tag{3}$$

$k\left(\mathbf{x}^s, \mathbf{x}^t\right) = \left\langle \phi\left(\mathbf{x}^s\right), \phi\left(\mathbf{x}^t\right) \right\rangle$ is a convex combination of m PSD kernels

$$\mathcal{K} \triangleq \left\{ k = \sum_{u=1}^{m} \beta_u k_u : \sum_{u=1}^{m} \beta_u = 1, \beta_u \geqslant 0, \forall u \right\}. \tag{4}$$

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

- $P = Q$ *if and only if* $d_k^2(P, Q) = 0$ *(In practice,* $d_k^2(P, Q) < \epsilon$*)*
- $\max\limits_{k \in \mathcal{K}} d_k^2(P, Q)\, \sigma_k^{-2} \Leftrightarrow$ min *Type II Error* ($d_k^2(P, Q) < \epsilon$ *when* $P \neq Q$)

DAN: Feature Learning

Linear-Time Algorithm of MK-MMD (Streaming Algorithm)

$O(n^2)$: $d_k^2(p,q) = \mathbf{E}_{\mathbf{x}^s\mathbf{x}'^s} k(\mathbf{x}^s, \mathbf{x}'^s) + \mathbf{E}_{\mathbf{x}^t\mathbf{x}'^t} k(\mathbf{x}^t, \mathbf{x}'^t) - 2\mathbf{E}_{\mathbf{x}^s\mathbf{x}^t} k(\mathbf{x}^s, \mathbf{x}^t)$

$O(n)$: $d_k^2(p,q) = \frac{2}{n_s}\sum_{i=1}^{n_s/2} g_k(\mathbf{z}_i) \to$ linear-time unbiased estimate

- Quad-tuple $\mathbf{z}_i \triangleq (\mathbf{x}_{2i-1}^s, \mathbf{x}_{2i}^s, \mathbf{x}_{2i-1}^t, \mathbf{x}_{2i}^t)$
- $g_k(\mathbf{z}_i) \triangleq k(\mathbf{x}_{2i-1}^s, \mathbf{x}_{2i}^s) + k(\mathbf{x}_{2i-1}^t, \mathbf{x}_{2i}^t) - k(\mathbf{x}_{2i-1}^s, \mathbf{x}_{2i}^t) - k(\mathbf{x}_{2i}^s, \mathbf{x}_{2i-1}^t)$

Stochastic Gradient Descent (SGD)

For each layer ℓ and for each quad-tuple $\mathbf{z}_i^\ell = \left(\mathbf{h}_{2i-1}^{s\ell}, \mathbf{h}_{2i}^{s\ell}, \mathbf{h}_{2i-1}^{t\ell}, \mathbf{h}_{2i}^{t\ell}\right)$

$$\nabla_{\Theta^\ell} = \frac{\partial J(\mathbf{z}_i)}{\partial \Theta^\ell} + \lambda \frac{\partial g_k\left(\mathbf{z}_i^\ell\right)}{\partial \Theta^\ell} \tag{5}$$

DAN: Kernel Learning

Learning optimal kernel $k = \sum_{u=1}^{m} \beta_u k_u$

Maximizing test power $\triangleq$ minimizing Type II error (Gretton et al. 2012)

$$\max_{k \in \mathcal{K}} d_k^2 \left(\mathcal{D}_s^\ell, \mathcal{D}_t^\ell \right) \sigma_k^{-2}, \tag{6}$$

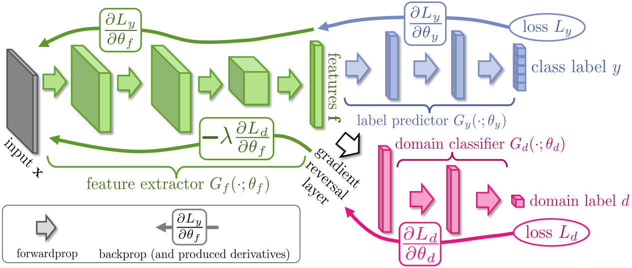
where $\sigma_k^2 = \mathbf{E}_\mathbf{z} g_k^2 (\mathbf{z}) - [\mathbf{E}_\mathbf{z} g_k (\mathbf{z})]^2$ is the estimation variance.

Quadratic Program (QP), scaling linearly to sample size: $O(m^2 n + m^3)$

$$\min_{\mathbf{d}^\mathsf{T} \boldsymbol{\beta} = 1, \boldsymbol{\beta} \geqslant \mathbf{0}} \boldsymbol{\beta}^\mathsf{T} \left(\mathbf{Q} + \epsilon \mathbf{I} \right) \boldsymbol{\beta}, \tag{7}$$

where $\mathbf{d} = (d_1, d_2, \ldots, d_m)^\mathsf{T}$, and each d_u is MMD using base kernel k_u.

DANN: Domain Adversarial Neural Network[2]



Adversarial adaptation: learning features indistinguishable across domains

$$E(\theta_f, \theta_y, \theta_d) = \sum_{\mathbf{x}_i \in \mathcal{D}_s} L_y(G_y(G_f(\mathbf{x}_i)), y_i) - \lambda \sum_{\mathbf{x}_i \in \mathcal{D}_s \cup \mathcal{D}_t} L_d(G_d(G_f(\mathbf{x}_i)), d_i) \tag{8}$$

$$(\hat{\theta}_f, \hat{\theta}_y) = \arg\min_{\theta_f, \theta_y} E(\theta_f, \theta_y, \theta_d) \quad (\hat{\theta}_d) = \arg\max_{\theta_d} E(\theta_f, \theta_y, \theta_d) \tag{9}$$

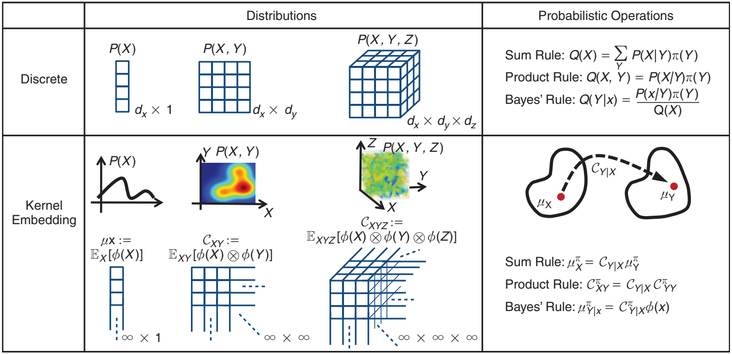
[2] *Ganin et al. Domain Adversarial Training of Neural Networks. JMLR '16.*

Outline

CDAN: Conditional Domain Adversarial Network[3]

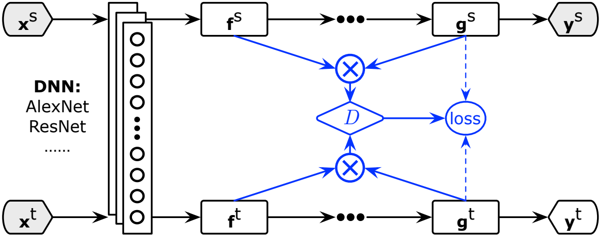
Main Idea of This Work: Distribution Embeddings with Statistics

- Capture cross-covariance statistics across multiple random vectors
  - Concatenation: $\mathbb{E}_{\mathbf{XY}}[\mathbf{X} \oplus \mathbf{Y}] = \mathbb{E}_{\mathbf{X}}[\mathbf{X}] \oplus \mathbb{E}_{\mathbf{Y}}[\mathbf{Y}]$
  - Multilinear: $\mathbb{E}_{\mathbf{XY}}[\mathbf{X} \otimes \mathbf{Y}] = \mathbb{E}_{\mathbf{X}}[\mathbf{X}|Y=1] \oplus \ldots \oplus \mathbb{E}_{\mathbf{X}}[\mathbf{X}|Y=C]$



[3] *Long et al. Conditional Adversarial Domain Adaptation. NIPS '18.*

CDAN: Multilinear Conditioning

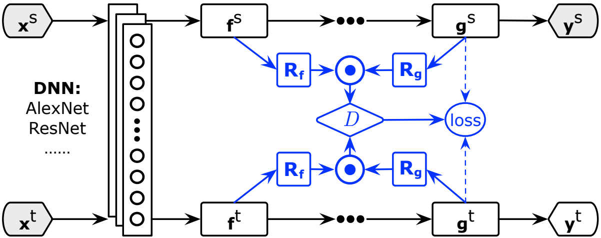


Conditional adaptation of distributions over representation & prediction

$$\begin{aligned} &\min_G E_G - \lambda E_{D,G} \\ &\min_D E_{D,G} \end{aligned} \tag{10}$$

$$E_{D,G} = -\frac{1}{n_s}\sum_{i=1}^{n_s} \log\left(D\left(\mathbf{f}_i^s \otimes \mathbf{g}_i^s\right)\right) - \frac{1}{n_t}\sum_{j=1}^{n_t} \log\left(1 - D\left(\mathbf{f}_j^t \otimes \mathbf{g}_j^t\right)\right) \tag{11}$$

CDAN: Randomized Multilinear Conditioning



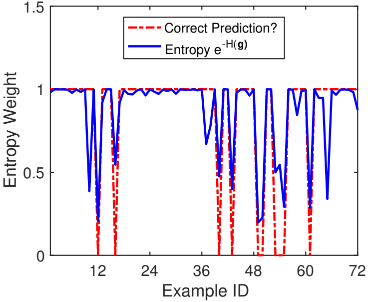
Conditional adaptation of distributions over representation & prediction

$$T_{\otimes}(\mathbf{f}, \mathbf{g}) = \mathbf{f} \otimes \mathbf{g} \tag{12}$$

$$T_{\odot}(\mathbf{f}, \mathbf{g}) = \frac{1}{\sqrt{d}} (\mathbf{R_f f}) \odot (\mathbf{R_g g}) \tag{13}$$

$$T(\mathbf{h}) = \begin{cases} T_{\otimes}(\mathbf{f}, \mathbf{g}) & \text{if } d_f \times d_g \leqslant 4096 \\ T_{\odot}(\mathbf{f}, \mathbf{g}) & \text{otherwise} \end{cases} \tag{14}$$

CDAN: Entropy Conditioning



Control the uncertainty of classifier prediction to guarantee transferability

$$\max_{D} \frac{1}{n_s}\sum_{i=1}^{n_s} e^{-H(\mathbf{g}_i^s)} \log\left[D\left(T\left(\mathbf{h}_i^s\right)\right)\right] + \frac{1}{n_t}\sum_{j=1}^{n_t} e^{-H\left(\mathbf{g}_j^t\right)} \log\left[1 - D\left(T\left(\mathbf{h}_j^t\right)\right)\right] \tag{15}$$

CDAN: Minimax Optimization Problem

Principled approaches: Conditional Domain Adversarial Networks (CDAN)

- Multilinear Conditioning: capture the cross-covariance between feature representation & classifier prediction to boost discriminability
- Entropy Conditioning: control the uncertainty of classifier prediction to guarantee transferability (entropy minimization principle)

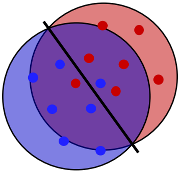
$$\begin{aligned}\min_G \ & \frac{1}{n_s}\sum_{i=1}^{n_s} L\left(G\left(\mathbf{x}_i^s\right), \mathbf{y}_i^s\right) \\ & + \frac{\lambda}{n_s}\sum_{i=1}^{n_s} e^{-H(\mathbf{g}_i^s)} \log\left[D\left(T\left(\mathbf{h}_i^s\right)\right)\right] + \frac{\lambda}{n_t}\sum_{j=1}^{n_t} e^{-H\left(\mathbf{g}_j^t\right)} \log\left[1 - D\left(T\left(\mathbf{h}_j^t\right)\right)\right]\end{aligned} \tag{16}$$

$$\max_D \ \frac{1}{n_s}\sum_{i=1}^{n_s} e^{-H(\mathbf{g}_i^s)} \log\left[D\left(T\left(\mathbf{h}_i^s\right)\right)\right] + \frac{1}{n_t}\sum_{j=1}^{n_t} e^{-H\left(\mathbf{g}_j^t\right)} \log\left[1 - D\left(T\left(\mathbf{h}_j^t\right)\right)\right] \tag{17}$$

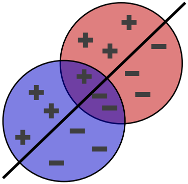
Outline

Notations and Assumptions

- Source risk: $\epsilon_P(G) = \mathbb{E}_{(\mathbf{f},\mathbf{y})\sim P}\left[G(\mathbf{f}) \neq \mathbf{y}\right]$
- Target risk: $\epsilon_Q(G) = \mathbb{E}_{(\mathbf{f},\mathbf{y})\sim Q}\left[G(\mathbf{f}) \neq \mathbf{y}\right]$
- Disagreement on source: $\epsilon_P(G_1, G_2) = \mathbb{E}_{(\mathbf{f},\mathbf{y})\sim P}\left[G_1(\mathbf{f}) \neq G_2(\mathbf{f})\right]$
- Disagreement on target: $\epsilon_Q(G_1, G_2) = \mathbb{E}_{(\mathbf{f},\mathbf{y})\sim Q}\left[G_1(\mathbf{f}) \neq G_2(\mathbf{f})\right]$
- Idea hypothesis: $G^* = \arg\min_G \epsilon_P(G) + \epsilon_Q(G)$
- Assumption: idea hypothesis has small risk $\epsilon_{ideal} = \epsilon_P(G^*) + \epsilon_Q(G^*)$



Distribution discrepancy



Ideal hypothesis with small error

Generalization Bound

Theorem

The probabilistic bound of the target risk $\epsilon_Q(G)$ of hypothesis G is given by the source risk $\epsilon_P(G)$ plus the distribution discrepancy:

$$\epsilon_Q(G) \leqslant \epsilon_P(G) + [\epsilon_P(G^*) + \epsilon_Q(G^*)] + |\epsilon_P(G, G^*) - \epsilon_Q(G, G^*)| \tag{18}$$

Proof.

By using the triangle inequalities, we have

$$\begin{aligned}\epsilon_Q(G) &\leqslant \epsilon_Q(G^*) + \epsilon_Q(G, G^*) \\ &\leqslant \epsilon_Q(G^*) + \epsilon_P(G, G^*) + \epsilon_Q(G, G^*) - \epsilon_P(G, G^*) \\ &\leqslant \epsilon_Q(G^*) + \epsilon_P(G, G^*) + |\epsilon_Q(G, G^*) - \epsilon_P(G, G^*)| \\ &\leqslant \epsilon_P(G) + [\epsilon_P(G^*) + \epsilon_Q(G^*)] + |\epsilon_P(G, G^*) - \epsilon_Q(G, G^*)|\end{aligned} \tag{19}$$

$\square$

Joint Distribution Discrepancy

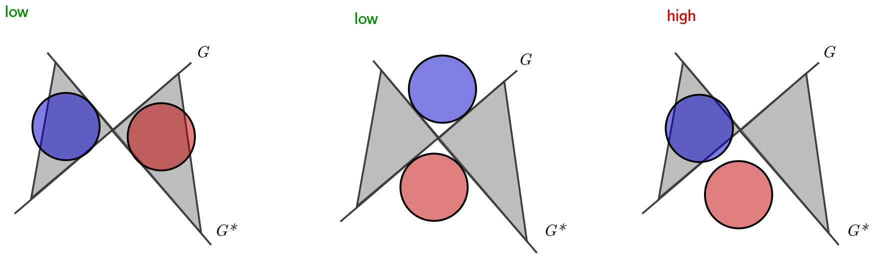
Define the proxies of the joint distributions $P(\mathbf{x}, \mathbf{y})$ and $Q(\mathbf{x}, \mathbf{y})$

- $P_G = (\mathbf{f}, G(\mathbf{f}))_{\mathbf{f} \sim P(\mathbf{f})}$, $Q_G = (\mathbf{f}, G(\mathbf{f}))_{\mathbf{f} \sim Q(\mathbf{f})}$
- $\epsilon_P(G, G^*) = \epsilon_{P_G}(G^*)$, $\epsilon_Q(G, G^*) = \epsilon_{Q_G}(G^*)$

Proof.

$$\epsilon_P(G, G^*) = \mathbb{E}_{(\mathbf{f},\mathbf{y}) \sim P}[G(\mathbf{f}) \neq G^*(\mathbf{f})] = \mathbb{E}_{(\mathbf{f},\mathbf{g}) \sim P_G}[\mathbf{g} \neq G^*(\mathbf{f})] = \epsilon_{P_G}(G^*) \qquad \square$$

How to bound the distribution discrepancy $|\epsilon_P(G, G^*) - \epsilon_Q(G, G^*)|$?

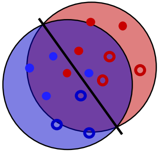


Upper-Bounding the Distribution Discrepancy

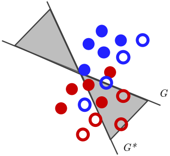
The distribution discrepancy $|\epsilon_P(G, G^*) - \epsilon_Q(G, G^*)|$ is bounded by

$$
\begin{aligned}
|\epsilon_P(G, G^*) - \epsilon_Q(G, G^*)| &= \left|\mathbb{E}_{(\mathbf{f},\mathbf{g})\sim P_G}[\mathbf{g} \neq G^*(\mathbf{f})] - \mathbb{E}_{(\mathbf{f},\mathbf{g})\sim Q_G}[\mathbf{g} \neq G^*(\mathbf{f})]\right| \\
&\leqslant \sup_{G^* \in \mathcal{H}} \left|\mathbb{E}_{(\mathbf{f},\mathbf{g})\sim P_G}[|\mathbf{g} - G^*(\mathbf{f})| \neq 0] - \mathbb{E}_{(\mathbf{f},\mathbf{g})\sim Q_G}[|\mathbf{g} - G^*(\mathbf{f})| \neq 0]\right| \\
&\leqslant \sup_{\delta \in \Delta} \left|\mathbb{E}_{(\mathbf{f},\mathbf{g})\sim P_G}[\delta(\mathbf{f}, \mathbf{g}) \neq 0] - \mathbb{E}_{(\mathbf{f},\mathbf{g})\sim Q_G}[\delta(\mathbf{f}, \mathbf{g}) \neq 0]\right| \\
&\leqslant \sup_{D \in \mathcal{H}_D} \left|\mathbb{E}_{(\mathbf{f},\mathbf{g})\sim P_G}[D(\mathbf{f}, \mathbf{g}) \neq 0] - \mathbb{E}_{(\mathbf{f},\mathbf{g})\sim Q_G}[D(\mathbf{f}, \mathbf{g}) \neq 0]\right|
\end{aligned}
$$

The upper-bound can be yielded by training the domain discriminator D.



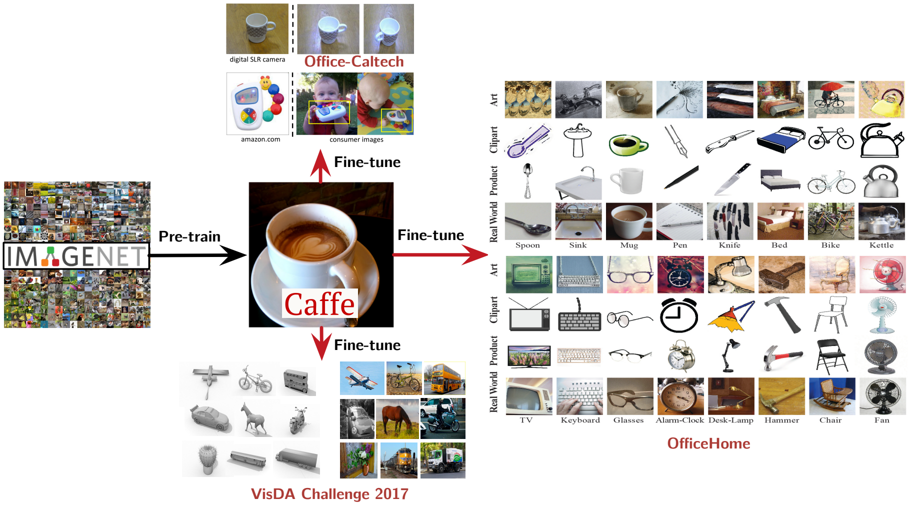
Distribution discrepancy



Hypothesis-based distribution discrepancy

Outline

Datasets

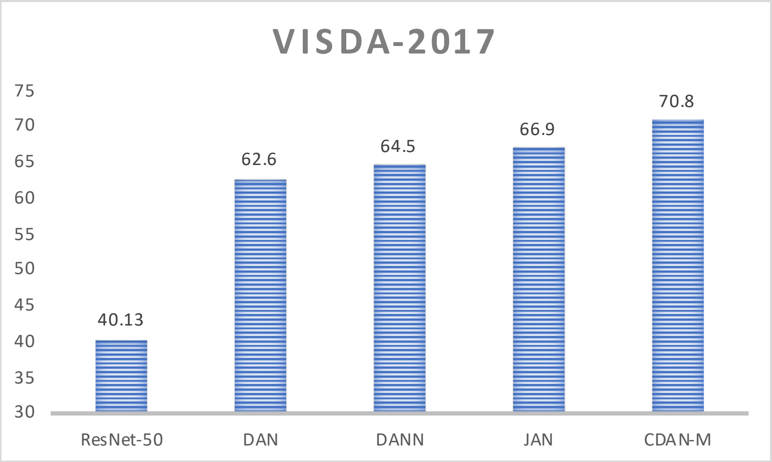

digital SLR camera
Office-Caltech
amazon.com
consumer images
Fine-tune
IMAGENET
Pre-train
Caffe
Fine-tune
Fine-tune
VisDA Challenge 2017
Art
Clipart
Product
Real World
Spoon
Sink
Mug
Pen
Knife
Bed
Bike
Kettle
TV
Keyboard
Glasses
Alarm-Clock
Desk-Lamp
Hammer
Chair
Fan
OfficeHome

Results

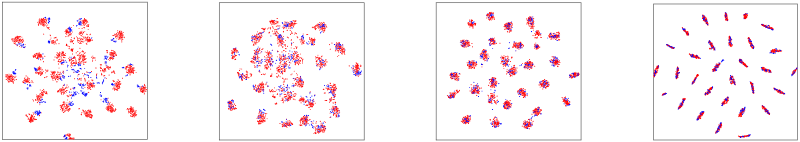
Table: Accuracy (%) on *Office-31* for unsupervised domain adaptation

Method	A → W	D → W	W → D	A → D	D → A	W → A	Avg
AlexNet	61.6±0.5	95.4±0.3	99.0±0.2	63.8±0.5	51.1±0.6	49.8±0.4	70.1
DAN	68.5±0.5	96.0±0.3	99.0±0.3	67.0±0.4	54.0±0.5	53.1±0.5	72.9
RTN	73.3±0.3	96.8±0.2	99.6±0.1	71.0±0.2	50.5±0.3	51.0±0.1	73.7
DANN	73.0±0.5	96.4±0.3	99.2±0.3	72.3±0.3	53.4±0.4	51.2±0.5	74.3
ADDA	73.5±0.6	96.2±0.4	98.8±0.4	71.6±0.4	54.6±0.5	53.5±0.6	74.7
JAN	74.9±0.3	96.6±0.2	99.5±0.2	71.8±0.2	**58.3**±0.3	55.0±0.4	76.0
CDAN-RM	**77.9**±0.3	96.9±0.2	**100.0**±.0	**74.6**±0.2	55.1±0.3	**57.5**±0.4	**77.0**
CDAN-M	77.6±0.2	**97.2**±0.1	**100.0**±.0	73.0±0.1	57.3±0.2	56.1±0.3	76.9
ResNet-50	68.4±0.2	96.7±0.1	99.3±0.1	68.9±0.2	62.5±0.3	60.7±0.3	76.1
DAN	80.5±0.4	97.1±0.2	99.6±0.1	78.6±0.2	63.6±0.3	62.8±0.2	80.4
RTN	84.5±0.2	96.8±0.1	99.4±0.1	77.5±0.3	66.2±0.2	64.8±0.3	81.6
DANN	82.0±0.4	96.9±0.2	99.1±0.1	79.7±0.4	68.2±0.4	67.4±0.5	82.2
ADDA	86.2±0.5	96.2±0.3	98.4±0.3	77.8±0.3	69.5±0.4	68.9±0.5	82.9
JAN	85.4±0.3	97.4±0.2	99.8±0.2	84.7±0.3	68.6±0.3	70.0±0.4	84.3
JAN-A	92.6±0.2	98.2±0.1	99.8±0.2	86.3±0.1	71.4±0.2	72.4±0.1	86.8
CDAN-RM	93.0±0.2	98.4±0.2	**100.0**±.0	89.2±0.3	70.2±0.4	69.4±0.4	86.7
CDAN-M	**93.1**±0.1	**98.6**±0.1	**100.0**±.0	**93.4**±0.2	**71.0**±0.3	**70.3**±0.3	**87.7**

Results

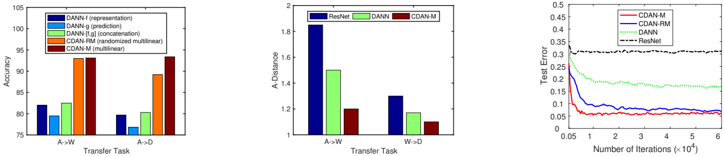

VISDA-2017
75
70
65
60
55
50
45
40
35
30
40.13
62.6
64.5
66.9
70.8
ResNet-50
DAN
DANN
JAN
CDAN-M

Analysis



(a) ResNet (b) DANN (c) CDAN-f (d) CDAN-fg

Figure: T-SNE on features by (a) ResNet, (b) DANN, (c) CDAN-f, (d) CDAN-fg.



(a) Conditioning (b) Discrepancy (c) Convergence

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

Open Problems

- Conditional Shift: $P(Y^s|\mathbf{X}^s) \neq Q(Y^t|\mathbf{X}^t)$
- Simulation-to-Real: $P(\mathbf{X}^s_{\text{low-level}}) \neq Q(\mathbf{X}^t_{\text{low-level}})$
- Open-Set/Zero-Shot (auxiliary info): $\mathbf{Y}^s \neq \mathbf{Y}^t$
- Heterogeneous (almost impossible): $\mathbf{X}^s \neq \mathbf{X}^t$
- Learning Transferable Architectures: BN, Skip-connection, etc.

- Xlearn library is available: `https://github.com/thuml/Xlearn`