Learning Transferable Features with Deep Adaptation Networks

Mingsheng Long[1,2], Yue Cao[1], Jianmin Wang[1], and Michael I. Jordan[2]

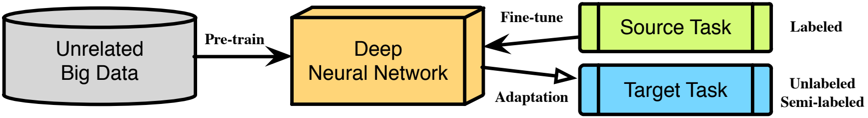
[1]School of Software, Institute for Data Science
Tsinghua University

[2]Department of EECS, Department of Statistics
University of California, Berkeley

International Conference on Machine Learning, 2015

Deep Learning for Domain Adaptation

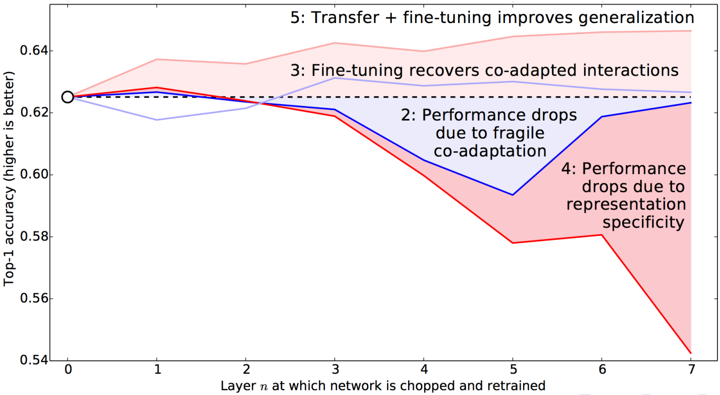
- None or very weak supervision in the target task (new domain)
  - Target classifier cannot be reliably trained due to over-fitting
  - Fine-tuning is impossible as it requires substantial supervision
- Generalize related supervised source task to the target task
  - Deep networks can learn transferable features for adaptation
- Hard to find big source task for learning deep features from scratch
  - Transfer from deep networks pre-trained on unrelated big dataset
  - Transferring features from distant tasks better than random features



How Transferable Are Deep Features?

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

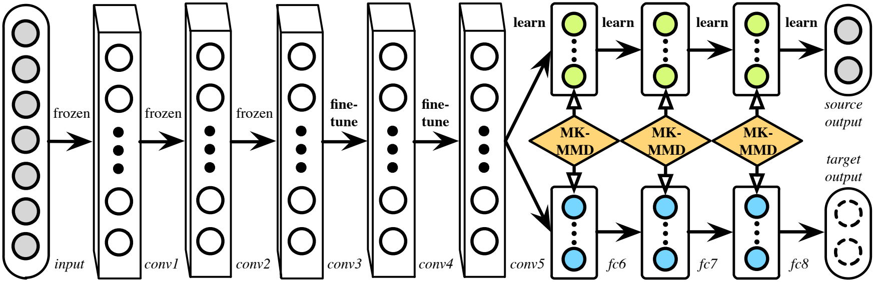
- Specialization of higher layer neurons to original task (new task ↓)
- Disentangling of variations in higher layers enlarges task discrepancy
- Transferability of features decreases while task discrepancy increases



Deep Adaptation Network (DAN)

Key Observations (AlexNet) (Krizhevsky et al. 2012)

- Convolutional layers learn general features: safely transferable
  - Safely freeze *conv*1-*conv*3 & fine-tune *conv*4-*conv*5
- Fully-connected layers fit task specificity: NOT safely transferable
  - Deeply adapt *fc*6-*fc*8 using statistically optimal two-sample matching



Objective Function

Main Problems

- Feature transferability decreases with increasing task discrepancy
- Higher layers are tailored to specific tasks, NOT safely transferable
- Adaptation effect may vanish in back-propagation of deep networks

Deep Adaptation with Optimal Matching

Deep adaptation: match distributions in multiple layers, including output

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

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

$\lambda > 0$ is a penalty, $\mathcal{D}_*^\ell = \left\{\mathbf{h}_i^{*\ell}\right\}$ is the ℓ-th layer hidden representation

MK-MMD

Multiple Kernel Maximum Mean Discrepancy (MK-MMD)

$\triangleq$ RKHS distance between *kernel embeddings* of distributions p and q

$$d_k^2(p,q) \triangleq \left\| \mathbf{E}_p\left[\phi\left(\mathbf{x}^s\right)\right] - \mathbf{E}_q\left[\phi\left(\mathbf{x}^t\right)\right] \right\|_{\mathcal{H}_k}^2, \tag{2}$$

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

$$\mathcal{K} \triangleq \left\{ k = \sum_{u=1}^{m} \beta_u k_u : \sum_{u=1}^{m} \beta_u = 1, \beta_u \geqslant 0, \forall u \right\}. \tag{3}$$

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

- $p = q$ *if and only if* $d_k^2(p,q) = 0$ *(In practice,* $d_k^2(p,q) < \varepsilon$*)*
- $\max_{k \in \mathcal{K}} d_k^2(p,q)\, \sigma_k^{-2} \Leftrightarrow$ min *Type II Error* ($d_k^2(p,q) < \varepsilon$ *when* $p \neq q$)

Learning CNN

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

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

$O(n)$: $d_k^2(p,q) = \frac{2}{n_s}\sum_{i=1}^{n_s/2} g_k(\mathbf{z}_i)$ → 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 $\mathbf{z}_i^\ell = (\mathbf{h}_{2i-1}^{s\ell}, \mathbf{h}_{2i}^{s\ell}, \mathbf{h}_{2i-1}^{t\ell}, \mathbf{h}_{2i}^{t\ell})$

$$\nabla_{\Theta^\ell} = \frac{\partial J(\mathbf{z}_i)}{\partial \Theta^\ell} + \lambda \frac{\partial g_k(\mathbf{z}_i^\ell)}{\partial \Theta^\ell} \quad (4)$$

Learning Kernel

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

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

$$\max_{k \in \mathcal{K}} d_k^2 \left(\mathcal{D}_s^\ell, \mathcal{D}_t^\ell\right) \sigma_k^{-2}, \tag{5}$$

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

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

$$\min_{\mathbf{d}^\top \boldsymbol{\beta} = 1, \boldsymbol{\beta} \geqslant \mathbf{0}} \boldsymbol{\beta}^\top \left(\mathbf{Q} + \varepsilon \mathbf{I}\right) \boldsymbol{\beta}, \tag{6}$$

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

Analysis

Theorem (Adaptation Bound)

(Ben-David et al. 2010) Let $\theta \in \mathcal{H}$ be a hypothesis, $\epsilon_s(\theta)$ and $\epsilon_t(\theta)$ be the expected risks of source and target respectively, then

$$\epsilon_t(\theta) \leqslant \epsilon_s(\theta) + d_{\mathcal{H}}(p, q) + C_0 \leqslant \epsilon_s(\theta) + 2d_k(p, q) + C, \tag{7}$$

where C is a constant for the complexity of hypothesis space, the empirical estimate of $\mathcal{H}$-divergence, and the risk of an ideal hypothesis for both tasks.

Two-Sample Classifier: Nonparametric vs. Parametric

- Nonparametric MMD directly approximates $d_{\mathcal{H}}(p, q)$
- Parametric classifier: adversarial training to approximate $d_{\mathcal{H}}(p, q)$

Experiment Setup

- **Datasets:** pre-trained on ImageNet, fined-tuned on Office&Caltech
- **Tasks:** 12 adaptation tasks → An unbiased look at dataset bias
- **Variants:** DAN; single-layer: DAN_7, DAN_8; single-kernel: DAN_{SK}
- **Protocols:** unsupervised adaptation vs semi-supervised adaptation
- **Parameter selection:** cross-validation by jointly assessing
  - test errors of source classifier and two-sample classifier (MK-MMD)

(Fei-Fei et al. 2012) (Jia et al. 2014) (Saenko et al. 2010)

Results and Discussion

Learning transferable features by deep adaptation and optimal matching

- Deep adaptation of multiple domain-specific layers (DAN) vs. shallow adaptation of one hard-to-tweak layer (DDC)
- Two samples can be matched better by MK-MMD vs. SK-MMD

Table: Accuracy on *Office-31* dataset via standard protocol (Gong et al. 2013)

Method	A → W	D → W	W → D	A → D	D → A	W → A	Average
TCA	21.5±0.0	50.1±0.0	58.4±0.0	11.4±0.0	8.0±0.0	14.6±0.0	27.3
GFK	19.7±0.0	49.7±0.0	63.1±0.0	10.6±0.0	7.9±0.0	15.8±0.0	27.8
CNN	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
LapCNN	60.4±0.3	94.7±0.5	**99.1**±0.2	63.1±0.6	51.6±0.4	48.2±0.5	69.5
DDC	61.8±0.4	95.0±0.5	98.5±0.4	64.4±0.3	52.1±0.8	52.2±0.4	70.6
DAN_7	63.2±0.2	94.8±0.4	98.9±0.3	65.2±0.4	52.3±0.4	52.1±0.4	71.1
DAN_8	63.8±0.4	94.6±0.5	98.8±0.6	65.8±0.4	52.8±0.4	51.9±0.5	71.3
DAN_{SK}	63.3±0.3	95.6±0.2	99.0±0.4	65.9±0.7	53.2±0.5	52.1±0.4	71.5
DAN	**68.5**±0.4	**96.0**±0.3	99.0±0.2	**67.0**±0.4	**54.0**±0.4	**53.1**±0.3	**72.9**

Results and Discussion

Semi-supervised adaptation: source supervision vs. target supervision?

- Limited target supervision is prone to over-fitting the target task
- Source supervision can provide strong but inaccurate inductive bias
- Via source inductive bias, target supervision is much more powerful
- Two-sample matching is more effective for bridging dissimilar tasks

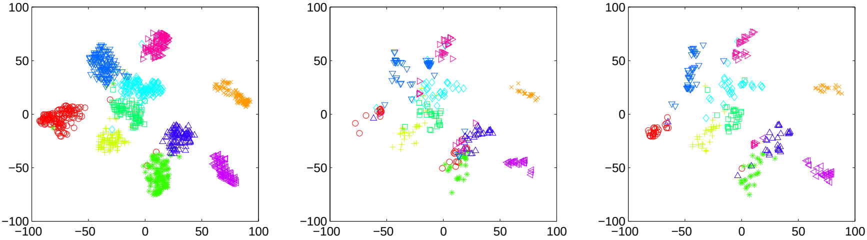
Table: Accuracy on *Office-31* dataset via down-sample protocol (Saenko et al.)

Paradigm	Method	$A \to W$	$D \to W$	$W \to D$	Average
Un-supervised	DDC	59.4±0.8	92.5±0.3	91.7±0.8	81.2
	DAN	**66.0**± 0.4	**93.5**±0.2	**95.3**±0.3	**84.9**
Semi-Supervised	DDC	84.1±0.6	95.4±0.4	96.3±0.3	91.9
	DAN	**85.7**±0.3	**97.2**±0.2	**96.4**±0.2	**93.1**

Visualization

How transferable are DAN features? t-SNE embedding for visualization

- With DAN features, target points form clearer class boundaries
- With DAN features, target points can be classified more accurately
    - Source and target categories are aligned better with DAN features

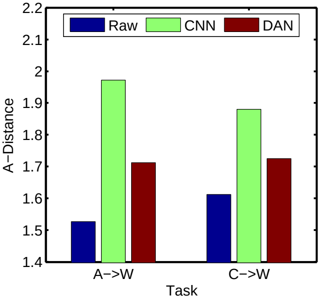


(a) CNN on Source (b) DDC on Target (c) DAN on Target

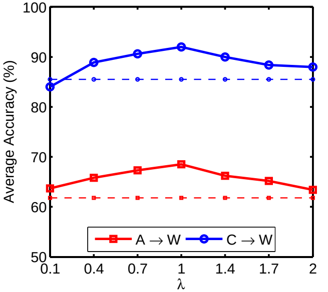
$\mathcal{A}$-distance $\hat{d}_{\mathcal{A}}$

How is generalization performance related to two-sample discrepancy?

- $\hat{d}_{\mathcal{A}}$ on CNN & DAN features is larger than $\hat{d}_{\mathcal{A}}$ on Raw features
  - Deep features are salient for both category & domain discrimination
- $\hat{d}_{\mathcal{A}}$ on DAN feature is much smaller than $\hat{d}_{\mathcal{A}}$ on CNN feature
  - Domain adaptation can be boosted by reducing domain discrepancy



(d) Cross-Domain $\mathcal{A}$-distance



(e) Accuracy vs. MMD Penalty λ

Summary

- A deep adaptation network for learning transferable features
- Two important improvements:
  - Deep adaptation of multiple task-specific layers (including output)
  - Optimal adaptation using multiple kernel two-sample matching
- A brief analysis of learning bound for the proposed deep network

- Open Problems
  - Principled way of deciding the boundary of generality and specificity
  - Deeper adaptation of convolutional layers to enhance transferability
  - Fine-grained adaptation using structural embeddings of distributions