

Research center for Ubiquitous Computing System,
Institute of Computing Technology, Chinese Academy of Sciences

Distant Domain Transfer Learning^a

Jindong Wang
wangjindong@ict.ac.cn

May 2017

^aThis slide is based on AAAI-17 paper: *Ben Tan, Yu Zhao, Sinno Jialin Pan and Qiang Yang: Distant domain transfer learning.*

Background of Transfer learning

Introduction

Author

DDTL: Distant Domain Transfer Learning

Selective Learning Algorithm for DDTL

Instance Selection via Reconstruction Error

Incorporation of Side Information

Learning Algorithm

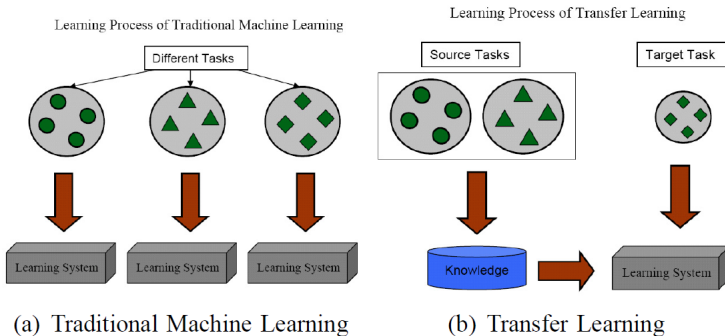
Experiments and Analysis

Related Work

Conclusion

Problems

- ▶ Building every model from scratch is time-consuming and expensive.
- ▶ But there are many existing knowledge. Can we reuse them?



Common Definition

- ▶ Wikipedia: research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a **different** but **related** problem [wik].

TL vs Traditional ML

Traditional ML:

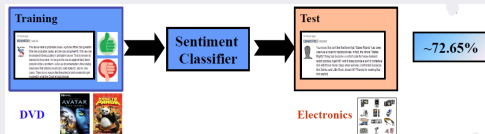
- ▶ Training and testing samples must be in the **same** feature distributions.
- ▶ Training samples must be **enough**.

Transfer learning:

- ▶ Source and target domains do **not** need to be in the same distributions.
- ▶ **Less** training samples, even **none**.

Example: sentiment classification

DVD → Electronics: Only got sentiment on DVD, how to transfer it to electronics?



Proceedings

- ▶ Data mining: ACM SIGKDD, IEEE ICDM, PKDD
- ▶ ML & AI: ICML, NIPS, AAAI, IJCAI, ECML
- ▶ Applications: ACM TIST, ACM SIGIR, WWW, ACL, IEEE TKDE

Many apps include image classification, natural language processing, activity recognition, and Wifi localization.

There are 4 authors of this paper:

Tan Ben

- ▶ Ph.D candidate at HKUST



Yu Zhang

- ▶ Research associate at HKUST



Sinno Jialin Pan

- ▶ Assistant professor at NTU
- ▶ Google scholar citations: 4,000+
- ▶ <http://www.ntu.edu.sg/home/sinnopan/>



Qiang Yang

- ▶ Head of CSE HKUST
- ▶ Fellow of AAAI/IEEE/AAAS/IAPR
- ▶ Google scholar citations: 30,000+
- ▶ <http://www.cs.ust.hk/~qyang/>



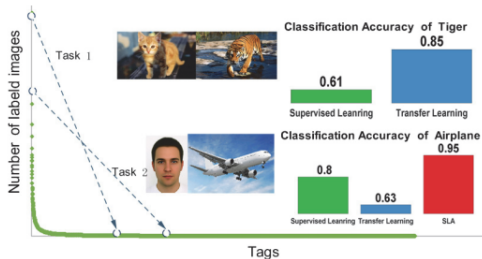
Introduction

Distant domain transfer learning



Traditional TL: the source and target domain are close [PY10]

DDTL: the source and target domain can be totally different!



- ▶ Task 1: Cat → Tiger, good performance for traditional TL.
- ▶ Task 2: Face → Airplane, bad performance for traditional TL.

Traditional TL: the source and target domain are close [PY10]

DDTL: the source and target domain can be totally different!



- ▶ Task 1: Cat → Tiger, good performance for traditional TL.
- ▶ Task 2: Face → Airplane, bad performance for traditional TL.

How to conduct transfer learning in such scenario when source and target domain are totally different?

Distant domain transfer learning (DDTL): exploit the unlabeled data in the intermediate domains to build a bridge between source and target domain.

Input:

- ▶ Labeled source domain $\mathcal{S} = \{(\mathbf{x}_S^1, y_S^1), \dots, (\mathbf{x}_S^{n_S}, y_S^{n_S})\}$
- ▶ Unlabeled target domain $\mathcal{T} = \{(\mathbf{x}_T^1, y_T^1), \dots, (\mathbf{x}_T^{n_T}, y_T^{n_T})\}$
- ▶ Mixture of unlabeled intermediate domains:
 $\mathcal{I} = \{(\mathbf{x}_I^1), \dots, (\mathbf{x}_I^{n_I})\}$

Output:

- ▶ labels of target domain

Constraints:

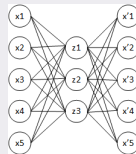
- ▶ $p_T(\mathbf{x}) \neq p_S(\mathbf{x}), p_T(\mathbf{x}) \neq p_I(\mathbf{x})$ and $p_T(y|\mathbf{x}) \neq p_S(y|\mathbf{x})$
- ▶ similarity between \mathcal{S} and \mathcal{T} is very small

Selective Learning Algorithm (SLA) is proposed to solve the DDTL problem, which is based on autoencoder.

Autoencoder

An unsupervised feed-forward neural network with an input layer, hidden layer and output layer.

- ▶ Encoding: $\mathbf{h} = f_e(\mathbf{x})$
- ▶ Decoding: $\hat{\mathbf{x}} = f_d(\mathbf{h})$
- ▶ Objective: $\min \sum_{i=1}^n \|\hat{\mathbf{x}}_i - \mathbf{x}_i\|_2^2$



To capture spatial information, a convolutional autoencoder is desired.

SLA algorithm

Instance selection via reconstruction error



Motivation: if data from source / intermediate domain is similar and useful to the target domain, then one should be able to find a pair of encoding and decoding functions that have small reconstruction error.

Objective: learn a pair of encoding and decoding functions by minimizing reconstruction errors on source, intermediate and target domain simultaneously.

Motivation: if data from source / intermediate domain is similar and useful to the target domain, then one should be able to find a pair of encoding and decoding functions that have small reconstruction error.

Objective: learn a pair of encoding and decoding functions by minimizing reconstruction errors on source, intermediate and target domain simultaneously.

$$\mathcal{J}_1(f_e, f_d, v_S, v_T) = \frac{1}{n_S} v_S^i \|\hat{x}_S^i - x_S^i\|_2^2 + \frac{1}{n_I} v_I^i \|\hat{x}_I^i - x_I^i\|_2^2 + \frac{1}{n_T} \|\hat{x}_T^i - x_T^i\|_2^2 + R(v_S, v_T) \quad (1)$$

v_S and v_T are selection indicators. $R(\cdot, \cdot)$ is the regularization term.

$$R(v_S, v_T) = -\frac{\lambda_S}{n_S} \sum_{i=1}^{n_S} v_S^i - \frac{\lambda_T}{n_T} \sum_{i=1}^{n_T} v_T^i \quad (2)$$

Learning in Eq. (1) is in unsupervised manner, consider to add some **side information**:

$$\begin{aligned}\mathcal{J}_2(f_c, f_e, f_d) = & \frac{1}{n_S} \sum_{i=1}^{n_S} v_S^i \mathcal{L}(y_S^i, f_c(\mathbf{h}_S^i)) + \frac{1}{n_T} \sum_{i=1}^{n_T} v_T^i \mathcal{L}(y_T^i, f_c(\mathbf{h}_T^i)) \\ & + \frac{1}{n_I} \sum_{i=1}^{n_I} v_I^i g(f_c(\mathbf{h}_I^i))\end{aligned}\tag{3}$$

$f_c(\cdot)$ is a classification function, $g(\cdot)$ is the entropy function:
 $g(z) = -z \ln z - (1 - z) \ln(1 - z)$ for $0 \leq z \leq 1$.

Overall objective function:

$$\begin{aligned}\min_{\Theta, v} \mathcal{J} &= \mathcal{J}_1 + \mathcal{J}_2 \\ \text{s.t. } & v_S^i, v_T^i \in \{0, 1\}\end{aligned}\tag{4}$$

Where Θ denotes all parameters $(f_c(\cdot), f_d(\cdot), f_e(\cdot))$ and $v = \{v_S, v_T\}$.

Technique: **Block Coordinate Decent (BCD)**, where in each iteration, variables in each block are optimized sequentially while keeping other variables fixed.

- ▶ fix v , update Θ using back propagation;
- ▶ fix Θ , obtain v as follows:

$$v_S^i = \begin{cases} 1 & \text{if } \mathcal{L}(y_s^i, f_c(f_e(x_S^i))) + \|\hat{x}_S^i - x_S^i\|_2^2 < \lambda_S \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

$$v_T^i = \begin{cases} 1 & \text{if } \|\hat{x}_T^i - x_T^i\|_2^2 + g(f_c(f_e(x_T^i))) < \lambda_T \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

Based on the above equations, only samples with low reconstruction error and high prediction confidence will be selected and used.

The learning algorithm is as follows:

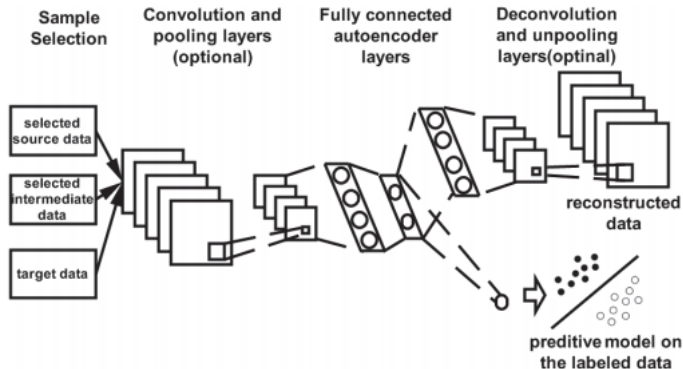
Algorithm 1 The Selective Learning Algorithm (SLA)

- 1: **Input:** Data in \mathcal{S} , \mathcal{T} and \mathcal{I} , and parameters λ_S , λ_I , and T ;
 - 2: Initialize Θ , $v_S = \mathbf{1}$, $v_I = \mathbf{0}$; *// All source data are used*
 - 3: **while** $t < T$ **do**
 - 4: Update Θ via the BP algorithm; *// Update the network*
 - 5: Update v by Eqs. (4) and (5); *// Select “useful” instances*
 - 6: $t = t + 1$
 - 7: **end while**
 - 8: **Output:** Θ and v .
-

Add convolution layers to the network, it can be viewed as a generalized autoencoder or convolutional autoencoder with side information.

SAN: supervised autoencoder, only autoencoder

SCAN: supervised convolutional autoencoder, using convolution



2 datasets

- ▶ **Caltech-256**: 30,607 images from 256 classes
- ▶ **Animals with Attributes (AwA)**: 30,475 images with 50 classes

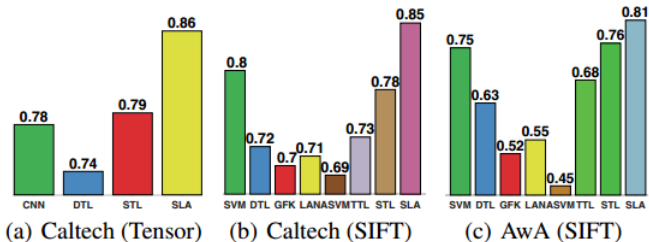
3 categories of baseline methods

- ▶ **Supervised learning**: SVM and CNN
- ▶ **Transfer learning**: ASVM, GFK, LAN, DTL and TTL
- ▶ **Self-taught learning method**

3 experiments

- ▶ Source and target domain are distant
- ▶ Visualize some intermediate domain data
- ▶ Evaluate the learning order of SLA

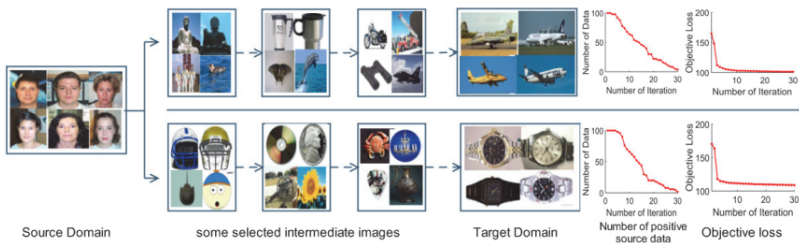
Average accuracies of different algorithms on Caltech-256 and AwA:



	SVM	DTL	GFK	LAN	ASVM	TTL	STL	SLA
'horse-to-face'	84 ± 2	88 ± 2	77 ± 3	79 ± 2	76 ± 4	78 ± 2	86 ± 3	92 ± 2
'airplane-to-gorilla'	75 ± 1	62 ± 3	67 ± 5	66 ± 4	51 ± 2	65 ± 2	76 ± 3	84 ± 2
'face-to-watch'	75 ± 7	68 ± 3	61 ± 4	63 ± 4	60 ± 5	67 ± 4	75 ± 5	88 ± 4
'zebra-to-collie'	71 ± 3	69 ± 2	56 ± 2	57 ± 3	59 ± 2	70 ± 3	72 ± 3	76 ± 2

Conclusion: For distant domains, SLA algorithm **outperforms** other methods.

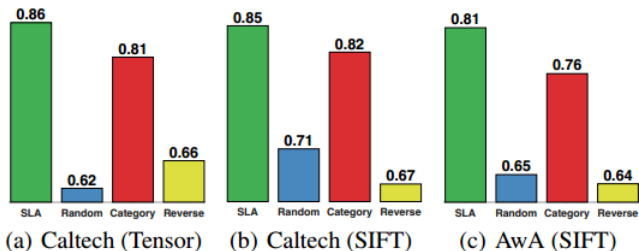
Visualization of the selected intermediate data over iterations of face-to-airplane and face-to-watch:



Conclusion:

- ▶ At the beginning, the intermediate domains are similar to **source domain**. In the end, it's more similar to the **target domain**.
- ▶ The number of positive examples in source domain **decreases**, and the value of objective function also **decreases**

Comparison results with different learning orders on the Caltech-256 and AWA datasets (Orders of intermediate domains changed):



Conclusion: Three types of different orders obtain worse results than SLA, and Category is close to SLA because this strategy is close to SLA.

DDTL is a novel difficult problem with many state-of-the-art methods **not applying** to it.

- ▶ **Typical transfer learning** approaches like instance reweighting [DYXY07] and feature mapping [PTKY11] do not apply to this problem, as they assume the domains are close.
- ▶ **Transitive transfer learning** [TSZY15]: manually select one intermediate domain as the bridge; **ours** automatically select many domains.
- ▶ **TLMS** [MMR09]: all the source domains in TLMS are labeled and closely related to the target domain.
- ▶ **Self-taught learning** [RBL⁺07]: use all domain data to learn; ours use intermediate domains.
- ▶ **Semi-supervised autoencoder** [WRMC12]: uses labeled and unlabeled data for learning; ours use intermediate domains and we use convolutional layers.

Contributions of this paper

- ▶ **First** work to study DDTL problem using mixture intermediate domains
- ▶ Propose **SLA** algorithm for DDTL problem
- ▶ Extensive experiments on real-world datasets show the **effectiveness** of SLA

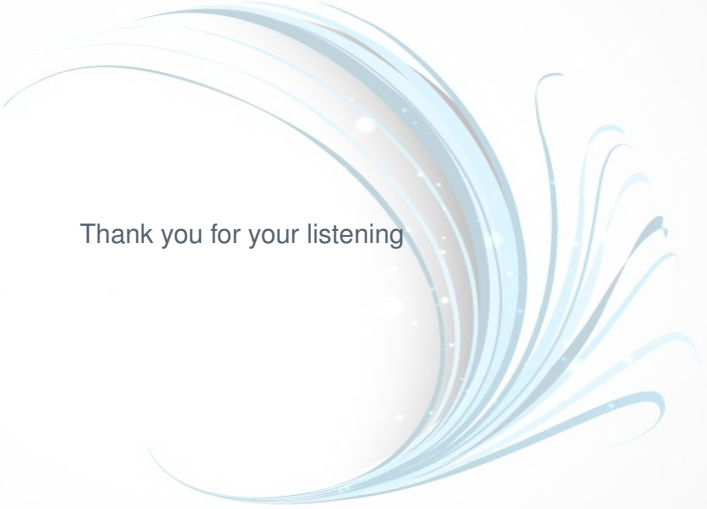
What we should learn from this paper

- ▶ Good layout, consider it as a template for **algorithm-paper**
- ▶ Introduction, tables and figures are good
- ▶ Experiment: **more datasets, more analysis**

- [DYXY07] Wenyan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu.
Boosting for transfer learning.
In *Proceedings of the 24th international conference on Machine learning*, pages 193–200. ACM, 2007.
- [MMR09] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh.
Domain adaptation with multiple sources.
In *Advances in neural information processing systems*, pages 1041–1048, 2009.
- [PTKY11] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang.
Domain adaptation via transfer component analysis.
IEEE Transactions on Neural Networks, 22(2):199–210, 2011.
- [PY10] Sinno Jialin Pan and Qiang Yang.
A survey on transfer learning.
Knowledge and Data Engineering, IEEE Transactions on, 22(10):1345–1359, 2010.
- [RBL⁺07] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y Ng.
Self-taught learning: transfer learning from unlabeled data.
In *Proceedings of the 24th international conference on Machine learning*, pages 759–766. ACM, 2007.
- [TSZY15] Ben Tan, Yangqiu Song, Erheng Zhong, and Qiang Yang.
Transitive transfer learning.
In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1155–1164. ACM, 2015.
- [wik] https://en.wikipedia.org/wiki/Inductive_transfer.
- [WRMC12] Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert.
Deep learning via semi-supervised embedding.
In *Neural Networks: Tricks of the Trade*, pages 639–655. Springer, 2012.



Q & A



Thank you for your listening