Learning to Transfer Examples for Partial Domain Adaptation

Zhangjie Cao*[1], Kaichao You*[1], Mingsheng Long[1](✉), Jianmin Wang[1], and Qiang Yang[2]

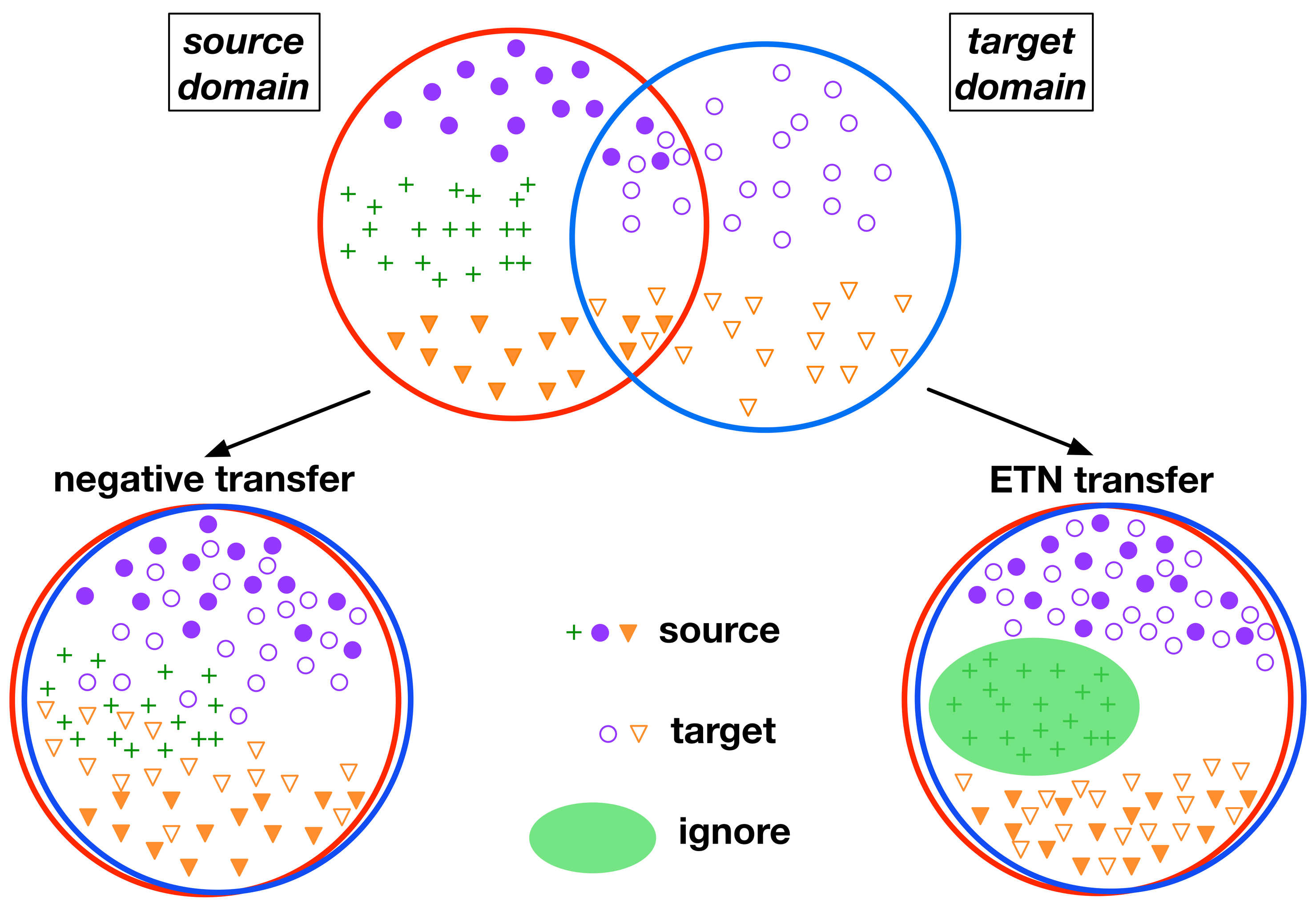
[1]KLiss, MOE; BNRist; School of Software, Tsinghua University, China
[1]Research Center for Big Data, Tsinghua University, China
[1]Beijing Key Laboratory for Industrial Big Data System and Application
[2]Hong Kong University of Science and Technology, China

CVPR JUNE 16-21, 2019 LONG BEACH • CALIFORNIA

Summary

- Partial domain adaptation: Deep learning across domains with different label spaces $\mathcal{C}_s \supset \mathcal{C}_t$
- Two main challenges:
  - Preserve positive transfer across domains in shared label space $\mathcal{C}_t$;
  - Avoid negative transfer across domains in outlier label space $\mathcal{C}_s \backslash \mathcal{C}_t$
- State-of-the-art results on partial domain adaptation datasets.
- Main contributions:
  - Propose an novel transferability quantifier to evaluate the transferability of each source example based on domain and class information;
  - Develop a weighting framework to re-weight each source example in classification and distribution alignment across domains.
- Code available @ `https://github.com/thuml/ETN`

Partial Domain Adaptation



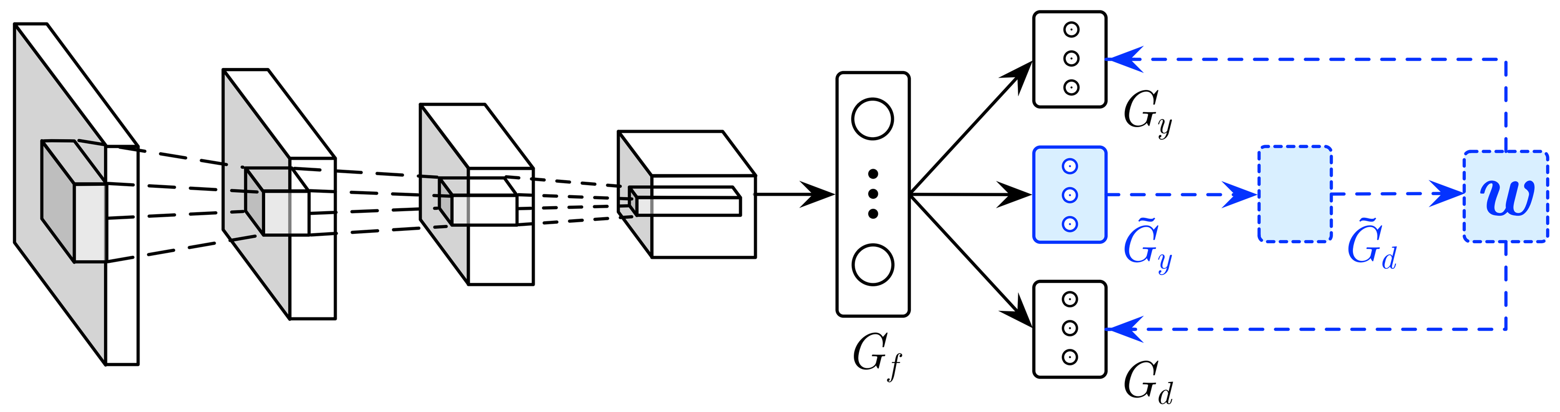
Challenge:

- Positive transfer is not exploited if no adaptation is employed;
- Negative transfer can happen if previous domain adaptation methods are used.

Solution:

- Matching source and target distributions within $\mathcal{C}_t$;
- Ignore source data in $\mathcal{C}_s \backslash \mathcal{C}_t$ in distribution alignment.

Example Transfer Network



- G_f: feature extractor
- G_y: label predictor
- G_d: domain discriminator
- $\hat{G}_y$, $\hat{G}_d$: auxiliary domain classifier

Transferability Weighting Framework

- Classification loss (H is the Entropy Minimization criterion).

$$E_{G_y} = \frac{1}{n_s}\sum_{i=1}^{n_s} w\left(\mathbf{x}_i^s\right) L\left(G_y\left(G_f\left(\mathbf{x}_i^s\right)\right), \mathbf{y}_i^s\right) + \frac{\gamma}{n_t}\sum_{j=1}^{n_t} H\left(G_y\left(G_f\left(\mathbf{x}_j^t\right)\right)\right). \quad (1)$$

- Weighted domain adversarial loss

$$E_{G_d} = -\frac{1}{n_s}\sum_{i=1}^{n_s} w\left(\mathbf{x}_i^s\right) \log\left(G_d\left(G_f\left(\mathbf{x}_i^s\right)\right)\right) - \frac{1}{n_t}\sum_{j=1}^{n_t} \log\left(1 - G_d\left(G_f\left(\mathbf{x}_j^t\right)\right)\right). \quad (2)$$

- Leaky-softmax activation of $\tilde{G}_y$:

$$\tilde{\sigma}\left(\mathbf{z}\right) = \frac{\exp\left(\mathbf{z}\right)}{|\mathcal{C}_s| + \sum_{c=1}^{|\mathcal{C}_s|}\exp\left(z_c\right)}, \quad (3)$$

- Relation of auxiliary label predictor $\tilde{G}_y$ and auxiliary domain discriminator $\tilde{G}_d$

$$\tilde{G}_d\left(G_f\left(\mathbf{x}_i\right)\right) = \sum_{c=1}^{|\mathcal{C}_s|} \tilde{G}_y^c\left(G_f\left(\mathbf{x}_i\right)\right), \quad (4)$$

- Loss on $\tilde{G}_y$ ($|\mathcal{C}_s|$ one-vs-rest binary classification tasks) and $\tilde{G}_d$

$$E_{\tilde{G}_y} = -\frac{\lambda}{n_s}\sum_{i=1}^{n_s}\sum_{c=1}^{|\mathcal{C}_s|}\left[y_{i,c}^s \log \tilde{G}_y^c\left(G_f\left(\mathbf{x}_i^s\right)\right) + \left(1 - y_{i,c}^s\right)\log\left(1 - \tilde{G}_y^c\left(G_f\left(\mathbf{x}_i^s\right)\right)\right)\right]. \quad (5)$$

$$E_{\tilde{G}_d} = -\frac{1}{n_s}\sum_{i=1}^{n_s}\log\left(\tilde{G}_d\left(G_f\left(\mathbf{x}_i^s\right)\right)\right) - \frac{1}{n_t}\sum_{j=1}^{n_t}\log\left(1 - \tilde{G}_d\left(G_f\left(\mathbf{x}_j^t\right)\right)\right). \quad (6)$$

- Weight:

$$w\left(\mathbf{x}_i^s\right) = 1 - \tilde{G}_d\left(G_f\left(\mathbf{x}_i^s\right)\right). \quad (7)$$

- Optimization Problem:

$$\left(\hat{\theta}_f, \hat{\theta}_y\right) = \arg\min_{\theta_f,\theta_y} E_{G_y} - E_{G_d}, \left(\hat{\theta}_d\right) = \arg\min_{\theta_d} E_{G_d}, \left(\hat{\theta}_{\tilde{y}}\right) = \arg\min_{\theta_{\tilde{y}}} E_{\tilde{G}_y} + E_{\tilde{G}_d}. \quad (8)$$

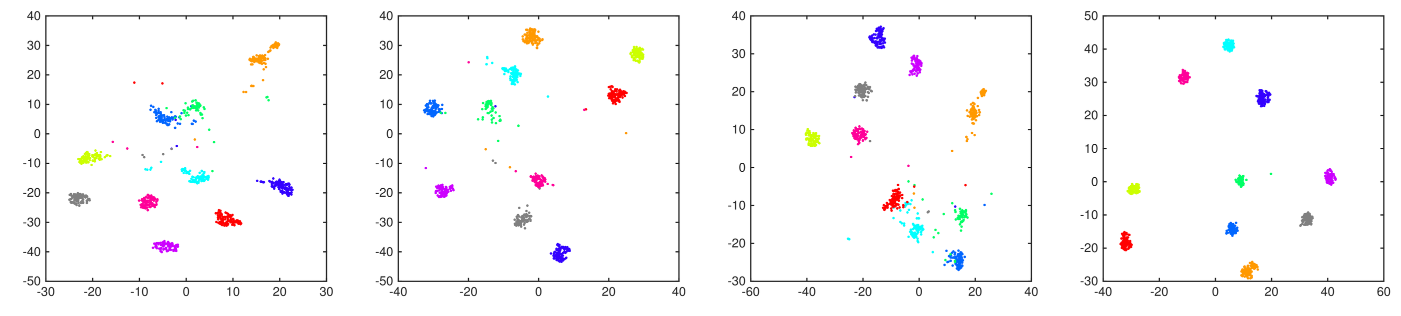
Experimental Results

Table: Classification Accuracy (%) on Office-Home Dataset

Method	Office-Home												
	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	Avg
ResNet	46.33	67.51	75.87	59.14	59.94	62.73	58.22	41.79	74.88	67.40	48.18	74.17	61.35
DANN	43.76	67.90	77.47	63.73	58.99	67.59	56.84	37.07	76.37	69.15	44.30	77.48	61.72
ADDA	45.23	68.79	79.21	64.56	60.01	68.29	57.56	38.89	77.45	70.28	45.23	78.32	62.82
RTN	49.31	57.70	**80.07**	63.54	63.47	73.38	65.11	41.73	75.32	63.18	43.57	80.50	63.07
IWAN	53.94	54.45	78.12	61.31	47.95	63.32	54.17	52.02	81.28	**76.46**	56.75	82.90	63.56
SAN	44.42	68.68	74.60	**67.49**	64.99	**77.80**	59.78	44.72	80.07	72.18	50.21	78.66	65.30
PADA	51.95	67.00	78.74	52.16	53.78	59.03	52.61	43.22	78.79	73.73	56.60	77.09	62.06
ETN	**59.24**	**77.03**	79.54	62.92	**65.73**	75.01	**68.29**	**55.37**	**84.37**	75.72	**57.66**	**84.54**	**70.45**

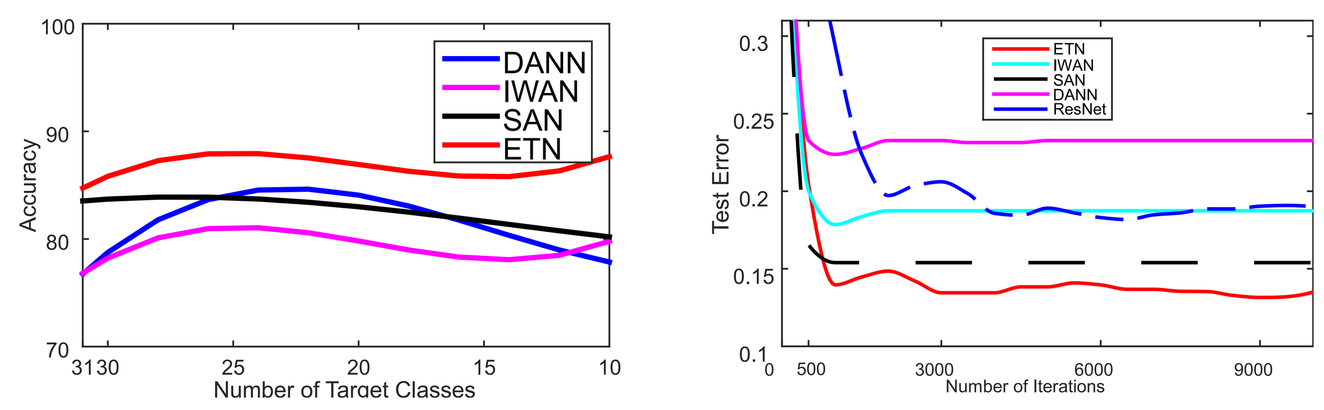
Table: Classification Accuracy (%) of ETN and Its Variants for Partial Domain Adaptation on Office-Home Dataset (**ResNet-50**)

Method	Office-Home												
	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	Avg
ETN w/o classifier	56.18	71.93	79.32	**65.11**	65.57	73.66	65.47	52.90	82.88	72.93	56.93	82.91	68.93
ETN w/o auxiliary	48.36	50.42	79.13	56.57	45.88	65.49	56.38	49.07	77.53	75.57	**58.81**	78.32	61.79
ETN	**59.24**	**77.03**	**79.54**	62.92	**65.73**	**75.01**	**68.29**	**55.37**	**84.37**	**75.72**	57.66	**84.54**	**70.45**



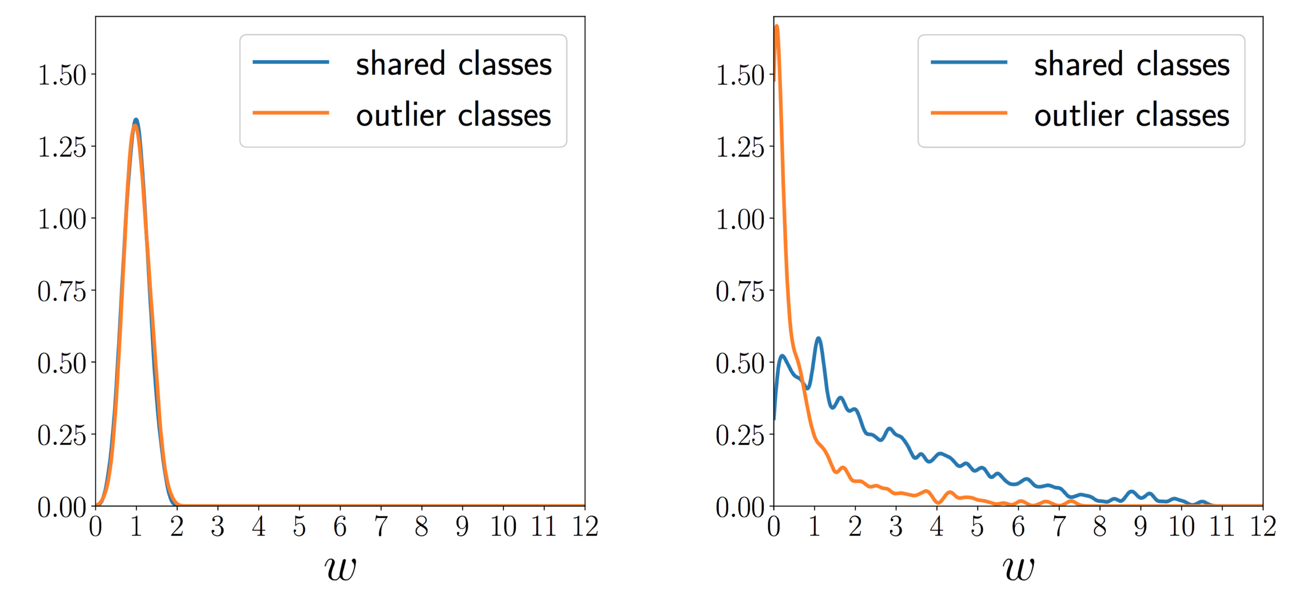
(a) DANN (b) SAN (c) IWAN (d) ETN

Figure: Visualization of features learned by DANN, SAN, IWAN, and ETN (class information is denoted by different colors).



(a) Accuracy w.r.t #target classes (b) Target error w.r.t. #iterations

Figure: Analytical results of ETN and baseline methods



(a) IWAN (b) ETN

Figure: Density function of the importance weights.