南京航空航天大學

Nanjing University of Aeronautics and Astronautics

Open Set Domain Adaptation

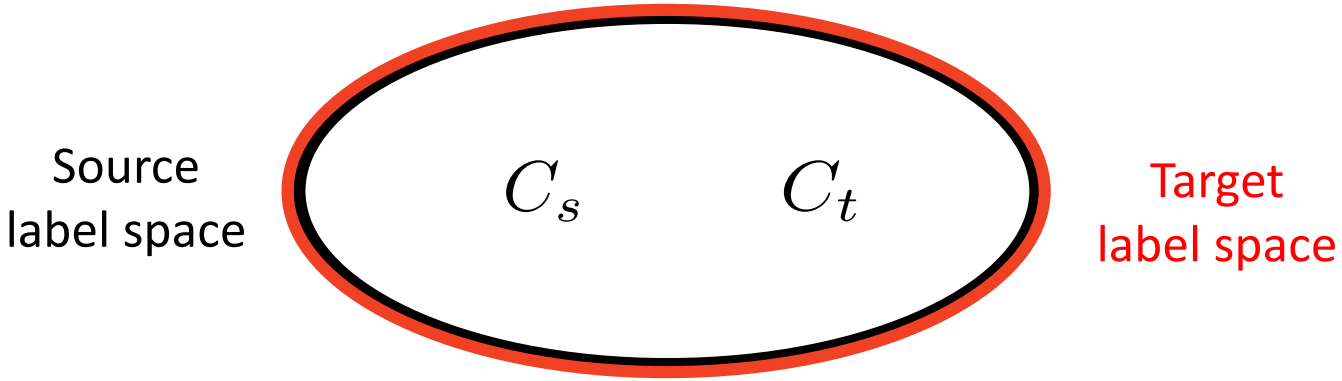
2019/11/27

Feng Sun

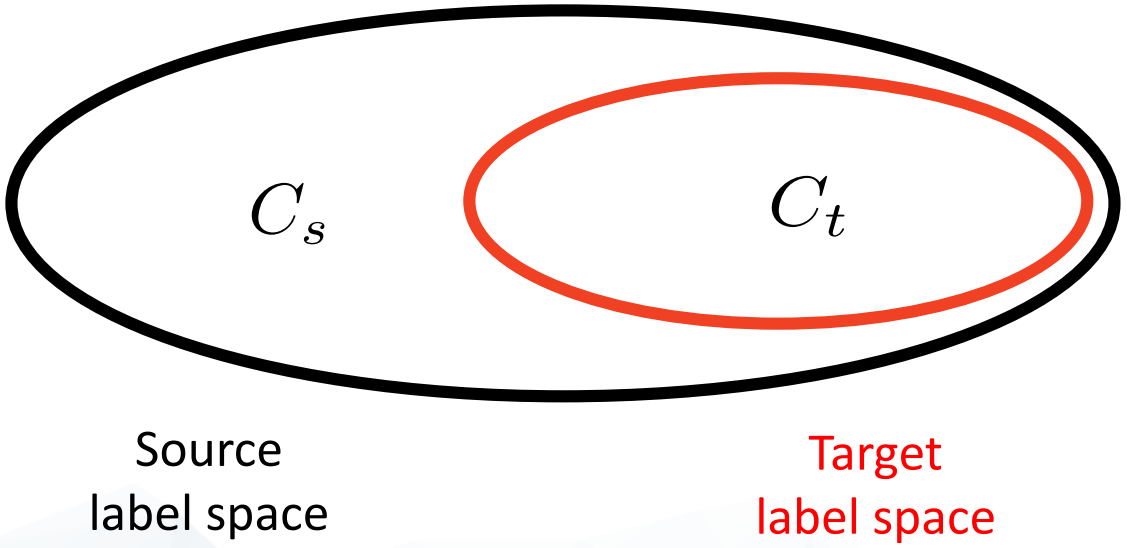
Domain Adaptation

- **Conditions**:
  - A source domain has abundant labeled examples
  - A target domain has limited or no labeled examples
- **Assumption**:
  - The two domains have same task, feature space and conditional distribution
  - The marginal distributions of two domains are different
  - Conventional domain adaptation generally assumes that the source and the target domains share identical label space, which may not hold in real world application
- **Two kinds of ways to address this problem with different views**:
  - Partial domain adaptation
  - Open set domain adaptation

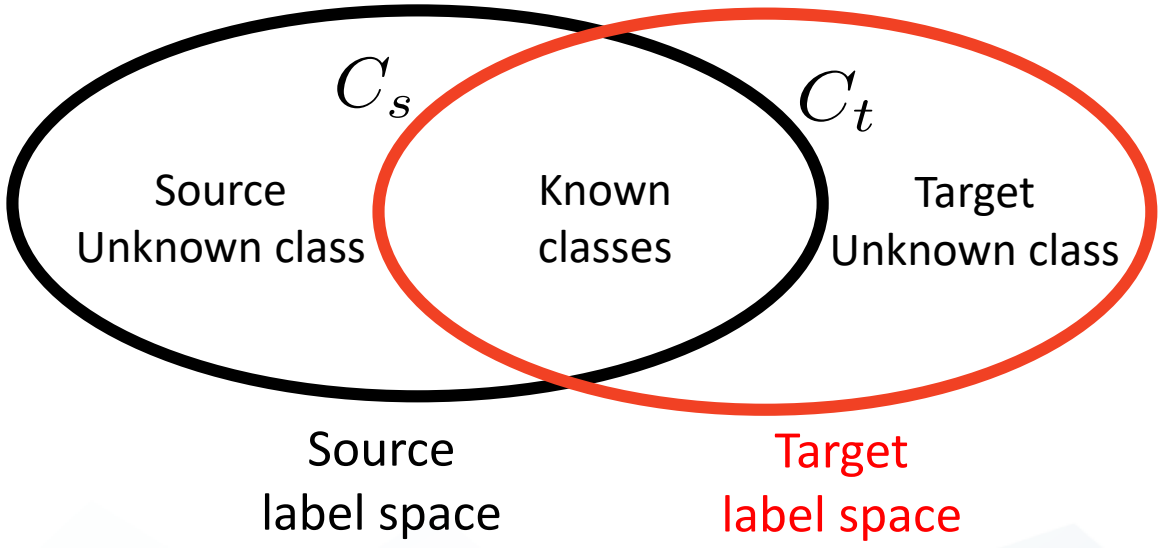
Comparison of Different Domain Adaptation



Closed Set Domain Adaptation



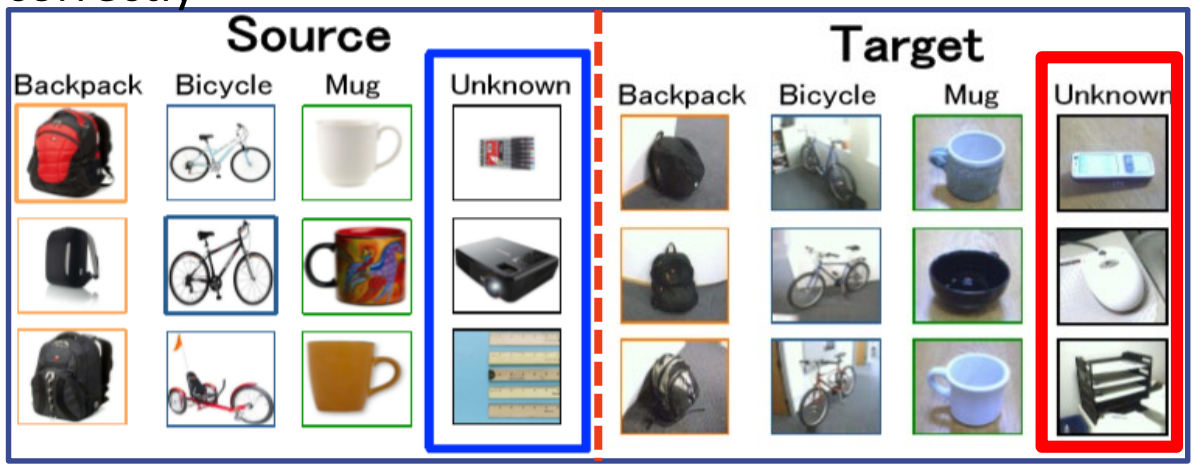
Partial Domain Adaptation



Open Set Domain Adaptation

Open Set Domain Adaptation

- **Two goals**:
  - To classify data of known classes in the target domain correctly
  - To reject data of all unknown classes as “unknown”



- **Two challenges**:
  - Negative transfer
  - Known/Unknown separation
- **There exists interaction between the two challenges**:
  - In open set domain adaptation, closed set methods will match the whole target domain with source domain, thus the unknown classes are also matched with source data. This obvious misalignment causes negative transfer.
  - The solution to negative transfer is only aligning the known classes with the source domain, which exactly gives rise to the second challenge.

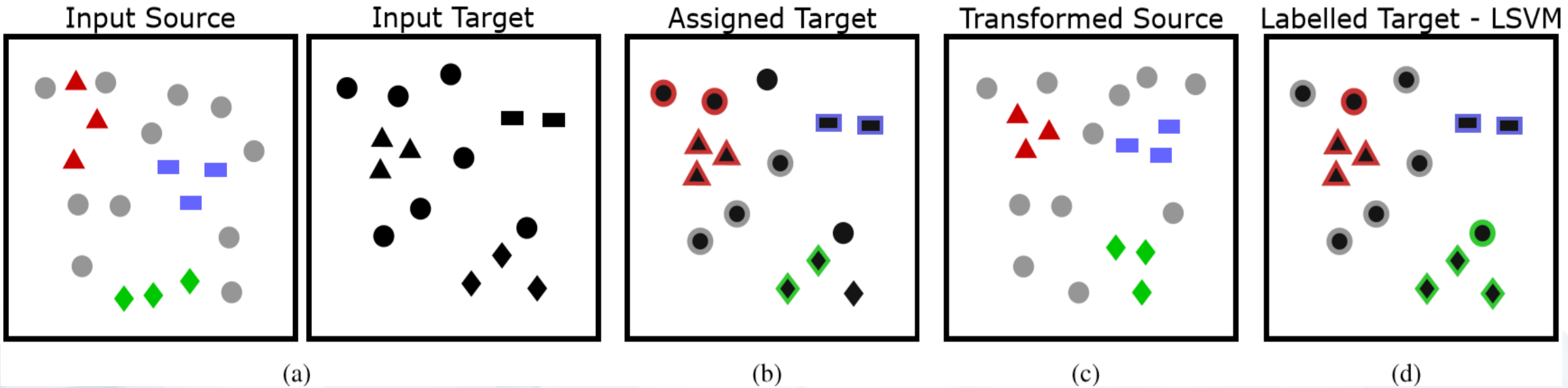
Open Set Domain Adaptation

Pau Panareda Busto[1,2]
[1]Airbus Group Innovations
Munich, Germany
pau.panareda-busto@airbus.com

Juergen Gall[2]
[2]Computer Vision Group
University of Bonn, Germany
gall@iai.uni-bonn.de

Method (ATI)

- **Key Idea:** Assign-and-Transform-Iteratively
- **Trick:** The unknown classes of the target are matched with source data of unknown classes
- **Method:**
  - Assigning a subset of the target samples to categories of the source domain
  - Computing a mapping from the source domain to the target domain
  - This iterative process is repeated until convergence



Assignment Problem

Notation:

- x_{ct} : If assign a target sample T_t to a class c, $x_{ct} = 1$ otherwise, $x_{ct} = 0$
- O_t : If declare a target sample T_t as outlier, $O_t = 1$ otherwise, $O_t = 0$
- S_c : The mean of all samples in the source domain labelled by class c
- d_{ct} : The cost of assigning a target sample T_t to a classes c

Formulation of entire assignment problem:

$$\min_{x_{ct}, o_t} \quad \sum_t \left(\sum_c d_{ct} x_{ct} + \lambda o_t \right)$$

$$\text{subject to} \quad \sum_c x_{ct} + o_t = 1 \qquad \forall t$$

$$\sum_t x_{ct} \geq 1 \qquad \forall c$$

$$x_{ct}, o_t \in \{0, 1\} \qquad \forall c, t$$

$$d_{ct} = \|S_c - T_t\|_2^2$$

- First constraints: ensures that a target sample is either assigned to one classes or declared as outlier
- Second constraints: ensures that at least one target sample is assigned to each class

Mapping Problem

- Estimating the mapping from the source to the target domain
- A linear transformation:
  - To learn a transformation matrix: $W \in R^{D \times D}$
  - The target function of learning:

$$f(W) = \frac{1}{2} \sum_{t} \sum_{c} x_{ct} \left\| W S_c - T_t \right\|_2^2$$

  - Estimating transformation matrix by minimizing the above loss function:

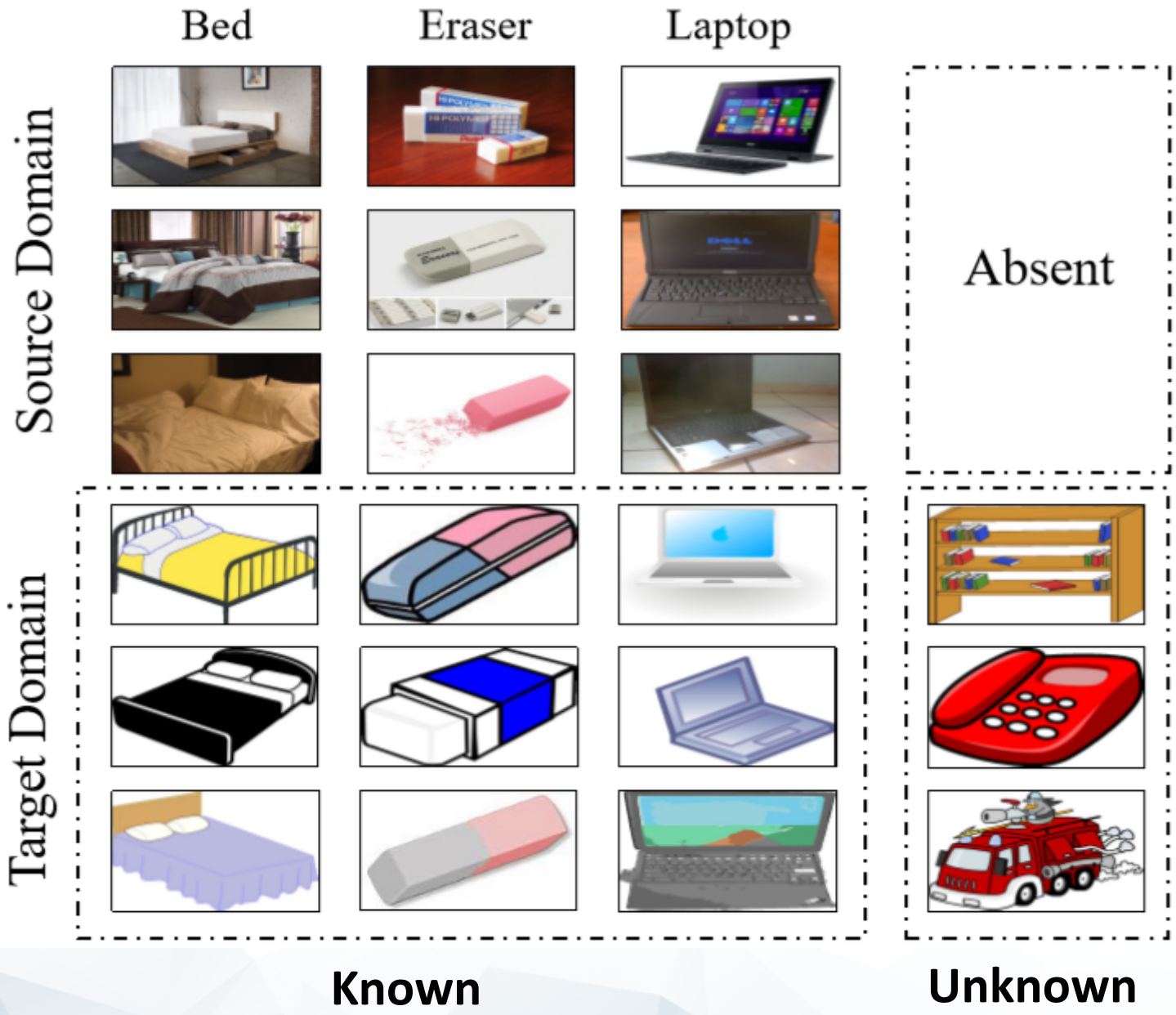
$$W^* = \arg\min_{W} f(W)$$

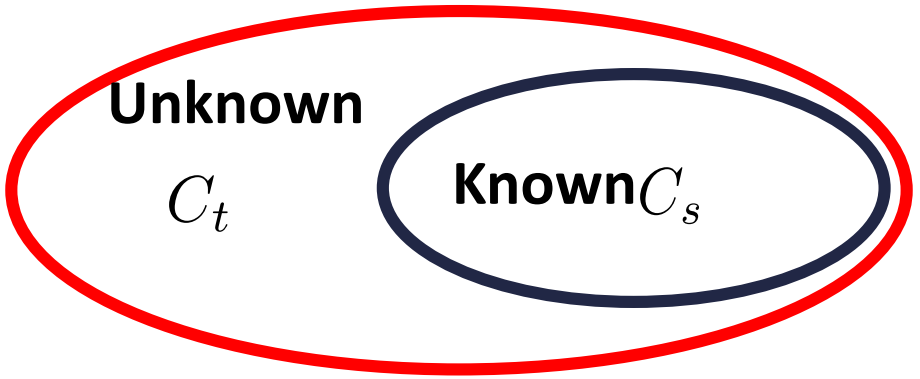
- After transformation matrix W is estimated, we map source samples to target domain
- After the approach has converged, we train linear SVMs in a one-vs-one setting on the transformed source samples.

Separate to Adapt: Open Set Domain Adaptation via Progressive Separation

Hong Liu[1*], Zhangjie Cao[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
h-l17@mails.tsinghua.edu.cn, {mingsheng, jimwang}@tsinghua.edu.cn, qyang@cse.ust.hk

Problem Setting





Target label space

Source label space

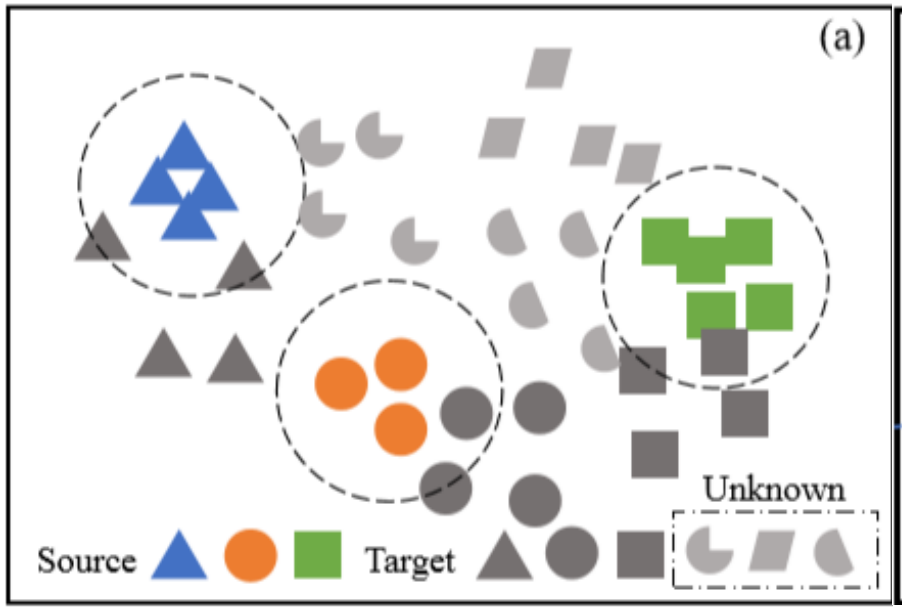
- **Problem Setting:** An example of open set domain adaptation problem, where all source classes are in the target classes and target have unknown classes.

Method (STA)

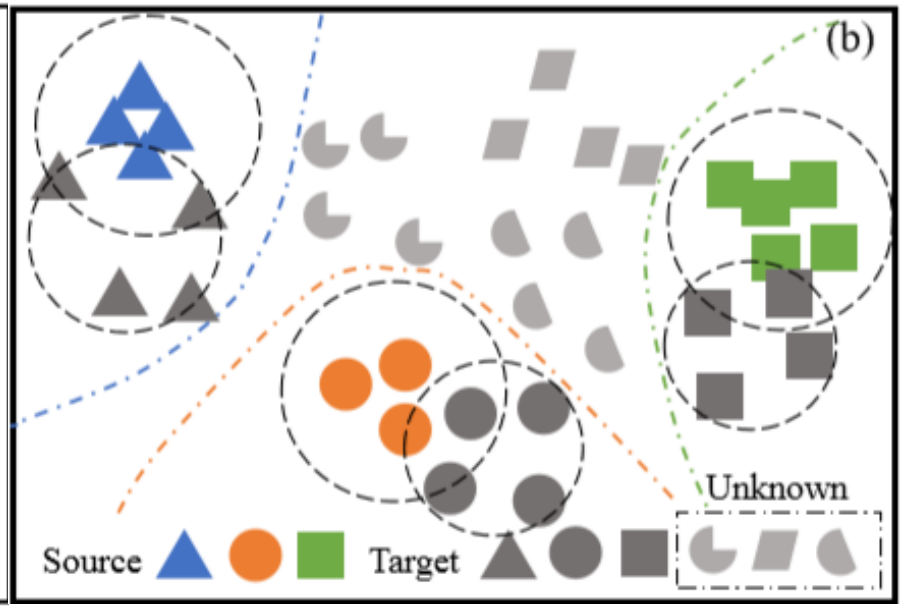
- **Key Idea:** Instance-level weight. Adversarial domain adaptation.
- **Motivation**: The main differences between the known and unknown classes
  - The known classes differ from the source domain only with distribution shift
  - The unknown classes deviate from the source domain much farther with both domain gap and semantic gap
- **Method**: A coarse-to-fine weighting mechanism. Separate to Adapt (STA)
  - **First step:** training a multi-binary classifier with source data to estimate the similarity between target data and each source class.
  - **Second step:** selecting the data with extremely high and low similarity as data of known and unknown classes, and train binary classifier with them to perform fine separation on all target samples.
  - Then, using instance-level weights generated by the above two steps to reject samples of unknown classes in adversarial domain adaptation.

The overview of STA

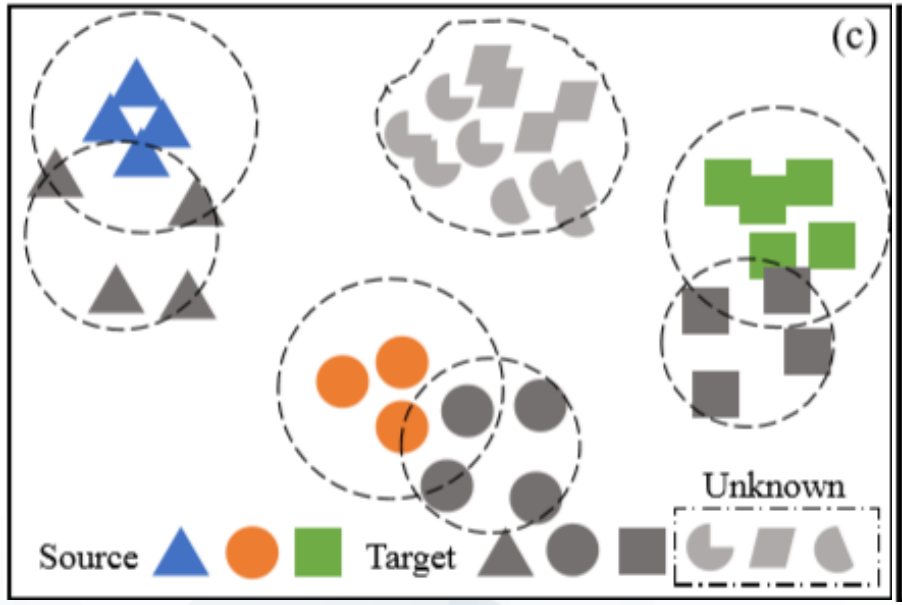
Problem Setting



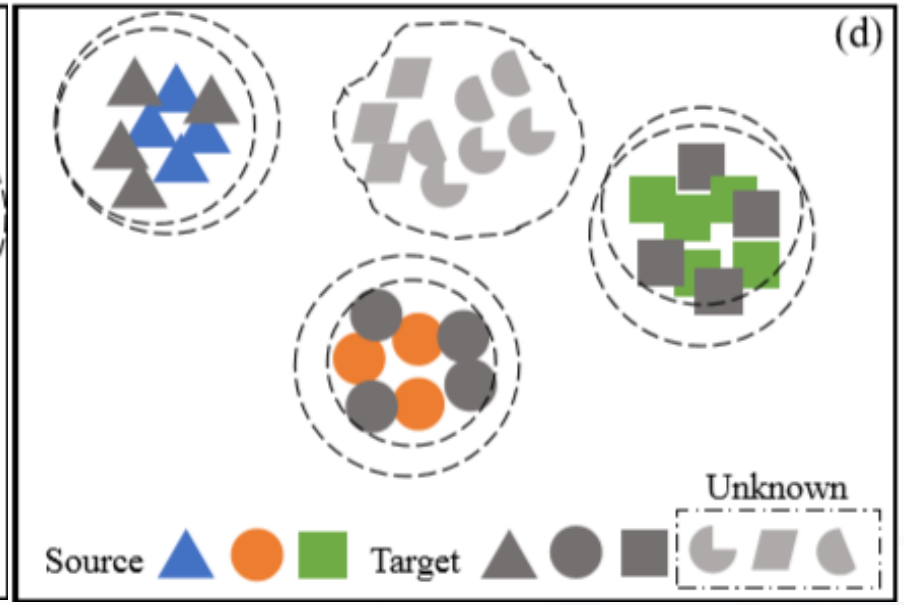
Coarse Separation



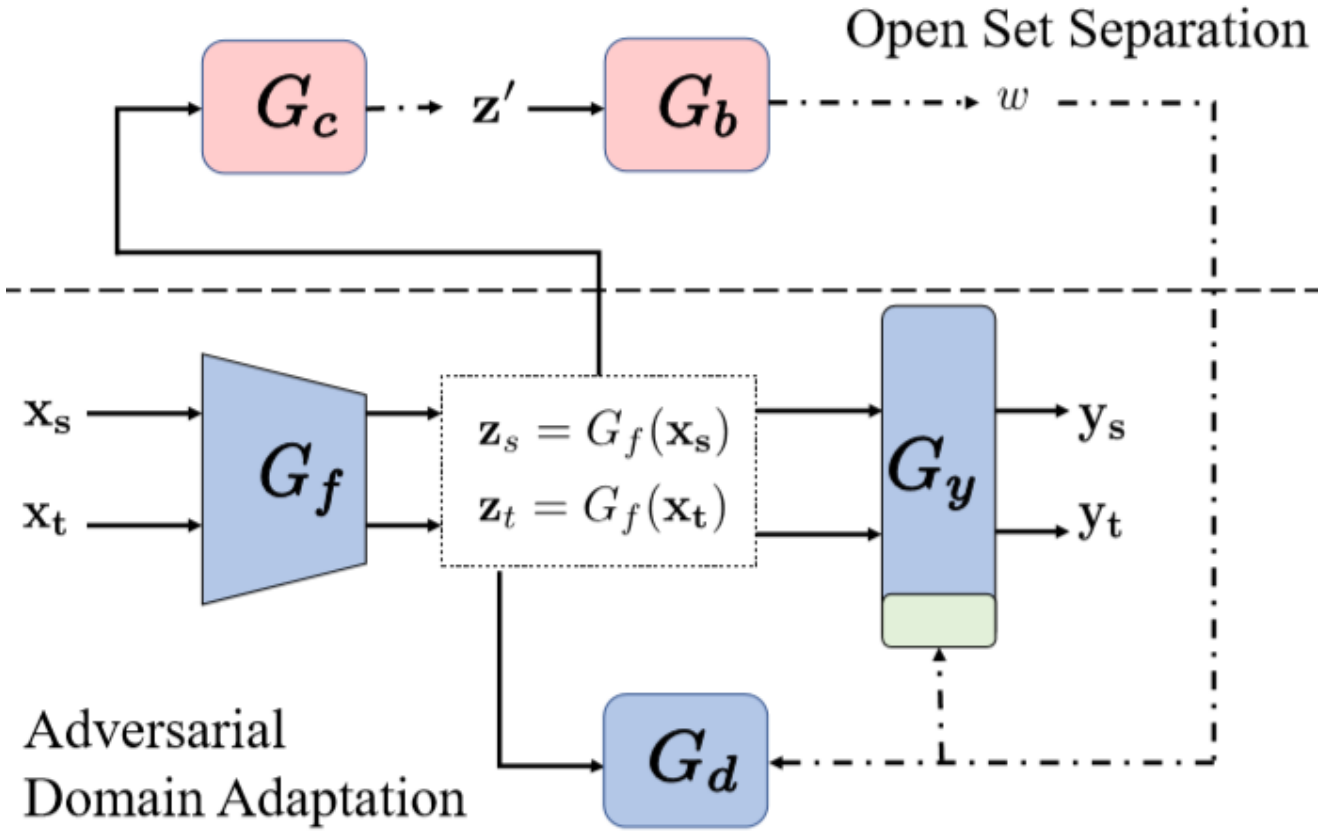
Fine Separation



Adversarial Domain Adaptation



The architecture of STA



G_c : Multi-Binary Classifier

G_b : Binary Classifier

G_f : Feature Extractor

G_y: Multi Classifier with $C_s + 1$ classes

G_d: Domain Discriminator

Separate to adapt

- **Loss function of multi-binary classifier:** G_c

Coarse Separation

$$L_S = \sum_{c=1}^{|C_s|} \frac{1}{n_s} \sum_{i=1}^{n_s} L_{bce}\left(G_c\left(G_f\left(\mathbf{x}_i^s\right)\right), I\left(y_i^s, c\right)\right)$$

- **Loss function of fine-grained binary classifier:** G_b

Fine Separation

$$L_b = \frac{1}{|\mathbf{X}'|} \sum_{\mathbf{x}_j \in \mathbf{X}'} L_{bce}\left(G_b(G_f\left(\mathbf{x}_j\right)), d_j\right)$$

Where $\mathbf{X}'$ to denote the set of filtered samples by the multi-binary classifier, and d_j to indicate whether a target sample is labeled as known $(d_j = 0)$ or unknown $(d_j = 1)$

Weighted Adaptation

- **Loss function of classifier G_y for $|\mathcal{C}_s|$ known classes:**

$$L_{cls}^{s} = \frac{1}{n_s} \sum_{\mathbf{x}_i \in \mathcal{D}_s} L_y \left(G_y^{1:|C_s|}(G_f(\mathbf{x}_i)), y_i \right)$$

- **Weighted loss function of domain discriminator :G_d**

Instance-level weight

$$w_j = G_b\left(G_f(\mathbf{x}_j)\right)$$

$$L_d = \frac{1}{n_s} \sum_{\mathbf{x}_i \in \mathcal{D}_s} L_{bce}\left(G_d(G_f(\mathbf{x}_i)), d_i\right) + \frac{1}{\sum_{\mathbf{x}_j \in \mathcal{D}_t}(1 - w_j)} \sum_{\mathbf{x}_j \in \mathcal{D}_t} (1 - w_j) L_{bce}\left(G_d(G_f(\mathbf{x}_j)), d_j\right)$$

Normalization

Where a larger w_j implies a higher probability to be from the unknown class.

Weighted Adaptation

- **Weighted loss function of classifier G_y for "unknown" class:**

$$L_{cls}^{t} = \frac{1}{\sum_{\mathbf{x}_j \in \mathcal{D}_t} w_j} \sum_{\mathbf{x}_j \in \mathcal{D}_t} w_j L_y \left(G_y^{|\mathcal{C}_s|+1}(G_f(\mathbf{x}_j)), l_{uk} \right)$$

Where l_{uk} is the unknown class.

- **Entropy minimization weighted loss:**

$$H(\boldsymbol{p}) = -\sum_k p_k \log p_k$$

$$L_e = \frac{1}{\sum_{\mathbf{x}_j \in \mathcal{D}_t} (1 - w_j)} \sum_{\mathbf{x}_j \in \mathcal{D}_t} (1 - w_j) H \left(G_y^{1:|\mathcal{C}_s|}(G_f(\mathbf{x}_j)) \right)$$

Where H is entropy loss. Incorporating the entropy minimization loss on the known classes of the target domain enforces the decision boundary to pass through low-density area in the target domain.

Training Procedure

- **Step 1:** Implementing coarse-to-fine separation
  - Training the feature extractor G_f and the classifier G_y to classify source samples.
  - Training the multi-binary classifier G_c by one-vs-rest way for each source class
  - Training the binary classifier G_b by using the target samples with high/low similarities to the source domain

$$(\hat{\theta}_f, \hat{\theta}_y, \hat{\theta}_b, \hat{\theta}_c) = \underset{\theta_f, \theta_y, \theta_b, \theta_c}{\arg\min} \; L^s_{cls} + L_s + L_b$$

- **Step 2**: Implementing adversarial domain adaptation
  - Aligning the feature distributions of known classes in the target domain with the source domain
  - Training G_y for the extra class with data from unknown class
  - Keeping training G_y with source samples to preserve the knowledge from the known classes

$$(\hat{\theta}_y, \hat{\theta}_d) = \underset{\theta_y, \theta_d}{\arg\min} \; L^s_{cls} + L^t_{cls} + L_d + \lambda L_e \qquad (\hat{\theta}_f) = \underset{\theta_f}{\arg\min} \; L^s_{cls} + L^t_{cls} - L_d + \lambda L_e$$

Experiment: Comparison 1

南京航空航天大学 Nanjing University of Aeronautics and Astronautics

Evaluation Metrics:

- **OS**: normalized accuracy for all the classes including the unknown as one class
- **OS***: normalized accuracy only on known classes
- **ALL:** the accuracy of all instances (without averaging accuracy over the classes)
- **UNK:** the accuracy of unknown samples

Table 1. Classification accuracy (%) of open set domain adaptation tasks on Digits (LeNet)

Method	Digits															
	SVHN → MNIST				USPS → MNIST				MNIST → USPS				Avg			
	OS	OS*	ALL	UNK	OS	OS*	ALL	UNK	OS	OS*	ALL	UNK	OS	OS*	ALL	UNK
OSVM [13]	54.3	63.1	37.4	10.5	43.1	32.3	63.5	97.5	79.8	77.9	84.2	**89.0**	59.1	57.7	61.7	65.7
MMD+OSVM	55.9	64.7	39.1	12.2	62.8	58.9	69.5	82.1	80.0	79.8	81.3	81.0	68.0	68.8	66.3	58.4
DANN+OSVM	62.9	75.3	39.2	0.70	84.4	92.4	72.9	0.90	33.8	40.5	21.4	44.3	60.4	69.4	44.5	15.3
ATI-λ	67.6	66.5	69.8	73.0	82.4	81.5	84.0	86.7	86.8	89.6	82.8	73.0	78.9	79.2	78.9	77.6
OSBP	63.0	59.1	71.0	82.3	**92.3**	91.2	**94.4**	**97.6**	92.1	**94.9**	88.1	78.0	82.4	81.7	84.5	85.9
STA	**76.9**	**75.4**	**80.0**	**84.4**	92.2	**91.3**	93.9	96.5	**93.0**	**94.9**	**90.3**	83.5	**87.3**	**87.2**	**88.1**	**88.1**

Experiment: Comparison 2

Table 2. Classification Accuracy (%) of open set domain adaptation tasks on Office-31 (ResNet-50)

Method	A → W		A → D		D → W		W → D		D → A		W → A		Avg	
	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*
ResNet [9]	82.5±1.2	82.7±0.9	85.2±0.3	85.5±0.9	94.1±0.3	94.3±0.7	96.6±0.2	97.0±0.4	71.6±1.0	71.5±1.1	75.5±1.0	75.2±1.6	84.2	84.4
RTN [19]	85.6±1.2	88.1±1.0	89.5±1.4	90.1±1.6	94.8±0.3	96.2±0.7	97.1±0.2	98.7±0.9	72.3±0.9	72.8±1.5	73.5±0.6	73.9±1.4	85.4	86.8
DANN [4]	85.3±0.7	87.7±1.1	86.5±0.6	87.7±0.6	**97.5**±0.2	**98.3**±0.5	**99.5**±0.1	**100.0**±.0	75.7±1.6	76.2±0.9	74.9±1.2	75.6±0.8	86.6	87.6
OpenMax [2]	87.4±0.5	87.5±0.3	87.1±0.9	88.4±0.9	96.1±0.4	96.2±0.3	98.4±0.3	98.5±0.3	83.4±1.0	82.1±0.6	82.8±0.9	82.8±0.6	89.0	89.3
ATI-λ [25]	87.4±1.5	88.9±1.4	84.3±1.2	86.6±1.1	93.6±1.0	95.3±1.0	96.5±0.9	98.7±0.8	78.0±1.8	79.6±1.5	80.4±1.4	81.4±1.2	86.7	88.4
OSBP [30]	86.5±2.0	87.6±2.1	88.6±1.4	89.2±1.3	97.0±1.0	96.5±0.4	97.9±0.9	98.7±0.6	88.9±2.5	90.6±2.3	85.8±2.5	84.9±1.3	90.8	91.3
STA	**89.5**±0.6	**92.1**±0.5	**93.7**±1.5	**96.1**± 0.4	**97.5**±0.2	96.5±0.5	**99.5**±0.2	99.6±0.1	**89.1**±0.5	**93.5**±0.8	**87.9**±0.9	**87.4**±0.6	**92.9**	**94.1**

Table 3. Classification accuracy OS (%) of open set domain adaptation tasks on Office-Home (ResNet-50)

Method	Ar → Cl	Pr → Cl	Rw → Cl	Ar → Pr	Cl → Pr	Rw → Pr	Cl → Ar	Pr → Ar	Rw → Ar	Ar → Rw	Cl → Rw	Pr → Rw	Avg
ResNet [9]	53.4±0.4	52.7±0.6	51.9±0.5	69.3±0.7	61.8±0.5	74.1±0.4	61.4±0.6	64.0±0.3	70.0±0.3	78.7±0.6	71.0±0.6	74.9±0.9	65.3
ATI-λ [25]	55.2±1.2	52.6±1.6	53.5±1.4	69.1±1.1	63.5±1.5	74.1±1.5	61.7±1.2	64.5±0.9	70.7±0.5	79.2±0.7	72.9±0.7	75.8±1.6	66.1
DANN [5]	54.6±0.7	49.7±1.6	51.9±1.4	69.5±1.1	63.5±1.0	72.9±0.8	61.9±1.2	63.3±1.0	71.3±1.0	80.2±0.8	71.7±0.4	74.2±0.4	65.4
OSBP [30]	56.7±1.9	51.5±2.1	49.2±2.4	67.5±1.5	65.5±1.5	74.0±1.5	62.5±2.0	64.8±1.1	69.3±1.1	80.6±0.9	74.7±2.2	71.5±1.9	65.7
OpenMax [2]	56.5±0.4	52.9±0.7	53.7±0.4	69.1±0.3	64.8±0.4	74.5±0.6	**64.1**±0.9	64.0±0.8	71.2±0.8	80.3±0.8	73.0±0.5	76.9±0.3	66.7
STA	**58.1**±0.6	**53.1**±0.9	**54.4**±1.0	**71.6**±1.2	**69.3**±1.0	**81.9**±0.5	63.4±0.5	**65.2**±0.8	**74.9**±1.0	**85.0**±0.2	**75.8**±0.4	**80.8**±0.3	**69.5**

Experiment: Comparison 3

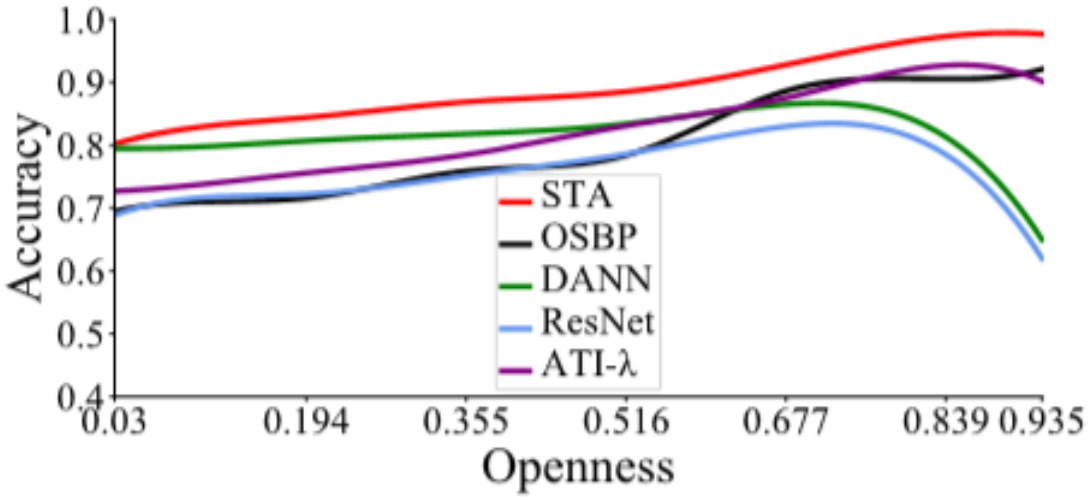
Variants of STA:

- STA (w/o) w: without weighting target samples in domain adversarial learning
- STA (w/o) c: without multi-binary classifier and replaces it with a softmax classifier
- STA (w/o) b: without binary classifier
- STA (w/o) j: without alternation between the two steps

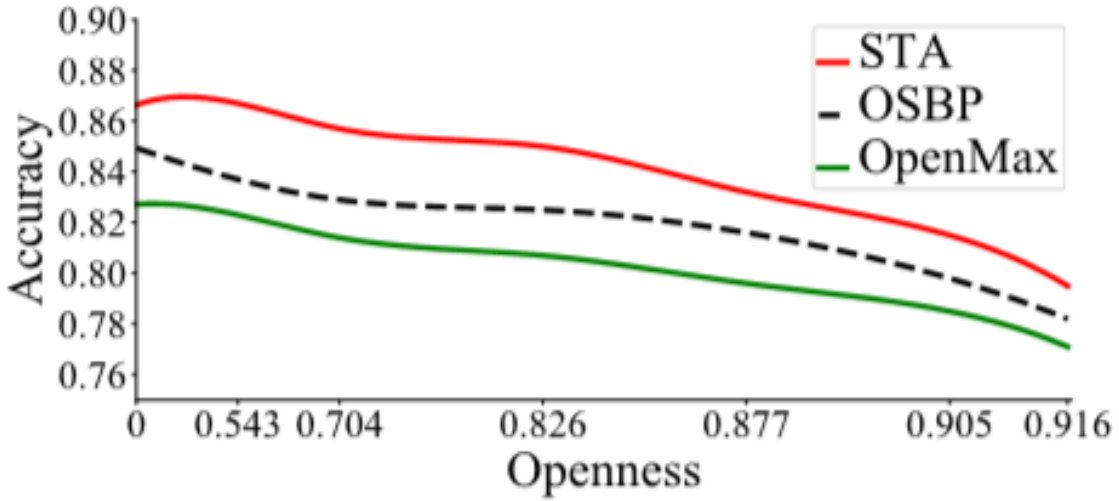
Table 4. Classification accuracy (%) of STA and its three variants on Office-31 (ResNet-50)

Method	A → W		A → D		D → W		W → D		D → A		W → A		Avg	
	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*	OS	OS*
STA w/o w	87.5±1.4	91.4±1.1	83.0±1.2	89.6±1.2	96.2±0.9	97.3±0.4	98.1±0.7	**100.0**±.0	80.3±1.5	79.3±1.5	71.2±1.2	74.3±1.2	86.1	88.7
STA w/o c	**90.4**±1.7	90.6±1.7	91.5±1.4	91.3±1.4	95.9±1.0	96.7±1.1	98.8±0.6	98.7±0.5	87.4±1.5	87.8±1.5	84.6±1.7	85.2±1.7	91.5	91.8
STA w/o b	85.0±1.5	89.0±1.5	90.6±1.2	91.5±1.3	94.8±1.9	**97.6**±0.8	96.2±0.6	98.2±0.5	77.7±2.2	82.5±2.4	78.9±2.6	83.6±3.5	87.2	90.4
STA w/o j	89.0±1.3	**92.8**±1.2	**94.8**±1.5	95.9±1.0	96.4±0.6	96.2±0.3	98.8±0.7	99.4±0.2	**89.7**±1.4	**93.6**±1.4	85.1±1.1	86.7±1.1	92.5	93.9
STA	89.5±0.6	92.1±0.5	93.7±1.5	**96.1**± 0.4	**97.5**±0.2	96.5±0.5	**99.5**±0.2	99.6±0.1	89.1±0.5	93.5±0.8	**87.9**±0.9	**87.4**±0.6	**92.9**	**94.1**

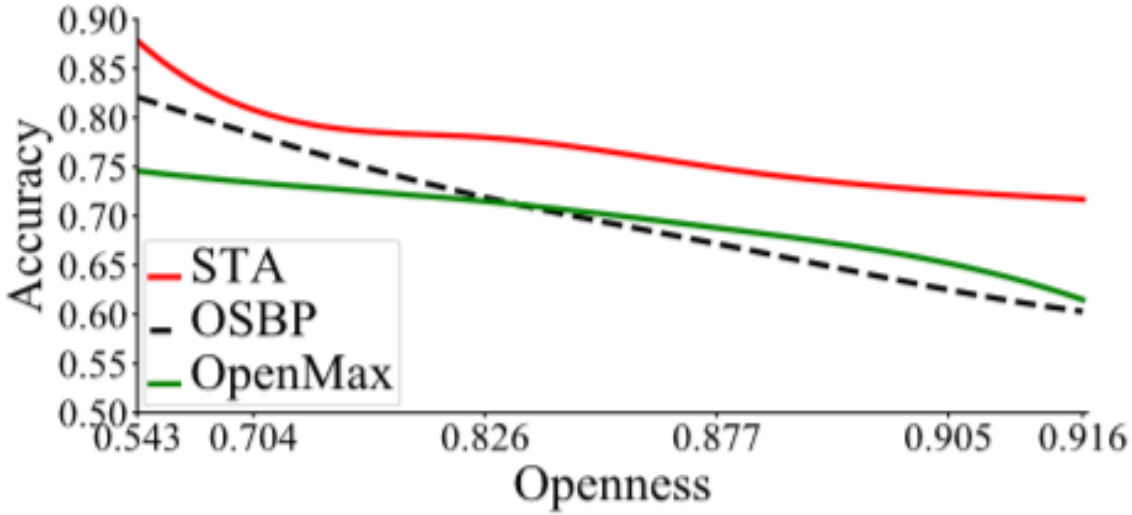
Experiment: Openness



(a) Office-31 OS



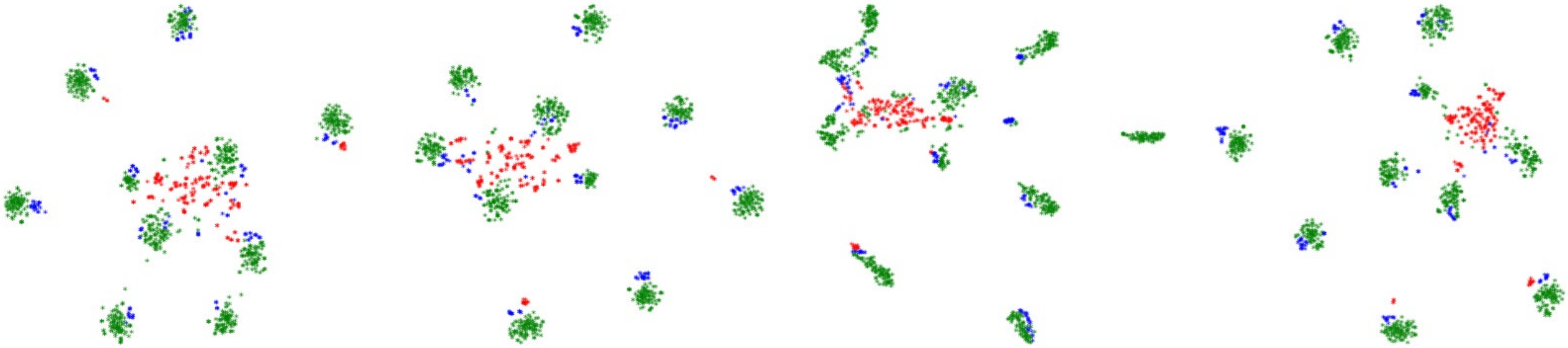
(b) Caltech-ImageNet Known



(c) Caltech-ImageNet Unknown

$$O = 1 - \frac{|\mathcal{C}_s|}{|\mathcal{C}_t|}$$

Experiment: Feature Visualization



(a) ResNet (b) DANN (c) OSBP (d) STA