Separate to Adapt: Open Set Domain Adaptation via Progressive Separation_论文笔记

CVPR 2019

Blog

Abstract

- target domain中包含source domain中没有的类别，称之为**Open Set Domain Adaptation (OSDA)**
- 之前解决OSDA的方法，没有考虑目标域的**开放性**（openness,开放性是通过所有目标类中未知类的比例来衡量的）
- 目前的工作是将**整个target domain 和 source domain对齐**，而不排除未知类样本，这可能会由于**未知类与已知类之间的不匹配而引起负迁移。**
- 本文提出了**Separate to Adapt (STA)**，一个端到端的方法来解决OSDA问题
- 该方法采用**从粗到细的加权机制**来逐步分离未知和已知类别的样本，同时权衡其在特征分布对齐上的重要性。
- 该方法openness-robust，它可以适应目标域中的各种开放性。

Introduction

- target domain 和 source domain 有完全相同的 label space，称之为 Closed Set Domain Adaptation
- 问题设置：target domain包含source domain中的所有的类，此外还有一些source domain中没有的类（source classes 是 target classes的一个子集）

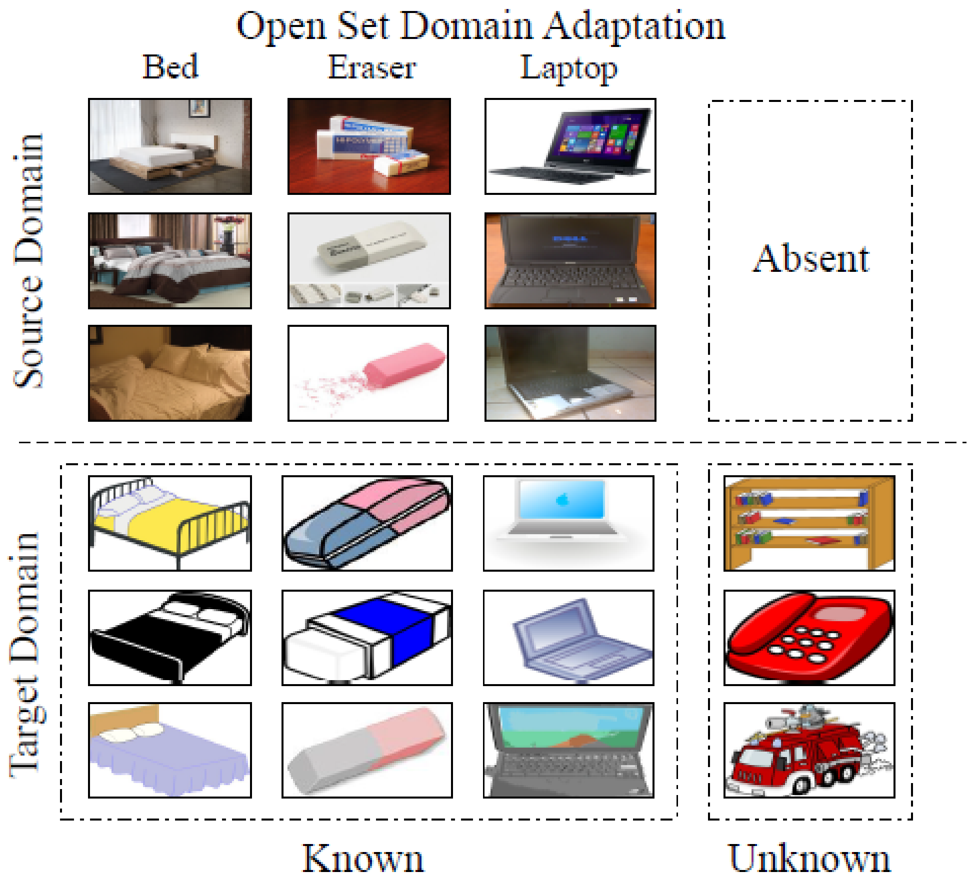


Figure 1. The open set domain adaptation problem, where the target domain contains “unknown” classes absent in the source domain.

- 目标是，将target domain中正确地**将已知类的数据分类**，并且将所有**未知类的数据reject为“未知”**
- 未知类不仅和source domain中的类存在domain gap，还存在semantic gap

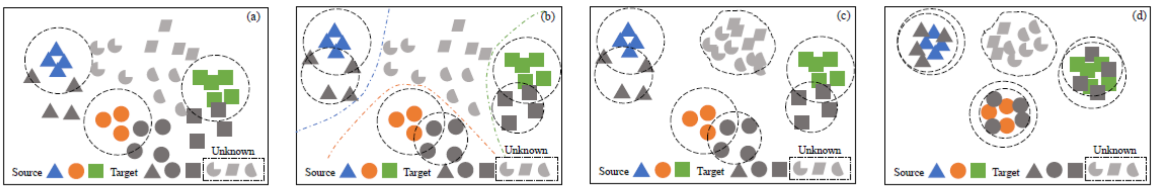


Figure 2. An overview of the proposed Separate to Adapt (STA) approach to open set domain adaptation. Gray shapes are data of target domain and shapes in color are data of source domain. Different kinds of shapes indicate different classes. (a) An example of open set domain adaptation problem, where all source classes are in the target classes and target have unknown classes. (b) The situation after training the multi-binary classifier $G_c|_{c=1}^{|\mathcal{C}_s|}$ for deriving coarse weights to distinguish the unknown classes from known classes in the target domain. Dashed curve in different colors indicates the decision boundary for each binary classifier G_c for the c-th class. It forces the target data of unknown classes to move away from source data. (c) The situation after training the fine-grained binary classifier G_b for deriving more accurate weights. Target data in shared classes and in unknown classes are deviated far away. (d) The situation after the final distribution alignment, where target data in the shared classes are close to their source domain counterparts. *Best viewed in color.*

- 提出 a **progressive separation mechanism** consisting of a coarse-to-fine separation pipeline
  - **用source data训练一个multi-binary分类器**
  - **选择具有极高和极低相似性的数据作为已知和未知类的数据**，并用它们训练一个**二分类器**以对所有目标样本进行精细分离
  - 迭代以上两步，并使用**instance-level权重**来拒绝未知类的样本

Method

Open Set Domain Adaptation

- source domain $\mathcal{D}_s = \left\{\left(\mathbf{x}_i^s, y_i^s\right)\right\}_{i=1}^{n_s}$,其中类别为$\mathcal{C}_s$
- target domain $\mathcal{D}_t = \left\{\mathbf{x}_j^t\right\}_{j=1}^{n_t}$,其中类别为$\mathcal{C}_t$, $\mathcal{C}_s \subset \mathcal{C}_t$
- $\mathcal{C}_{t\backslash s}$ 表示target中的未知类
- 源域分布为p, 目标域分布为q，$p \neq q\mathcal{C}_s$
- 定义openness，$\mathbb{O} = 1 - \frac{|\mathcal{C}_a|}{|\mathcal{C}_t|}$

Separate to Adapt

- 对于OSDA问题，如果直接对两个域进行对齐，会导致负迁移的问题，因此首先将target domain中已知类和未知类分离，然后仅对已知类的样本进行feature adaptation

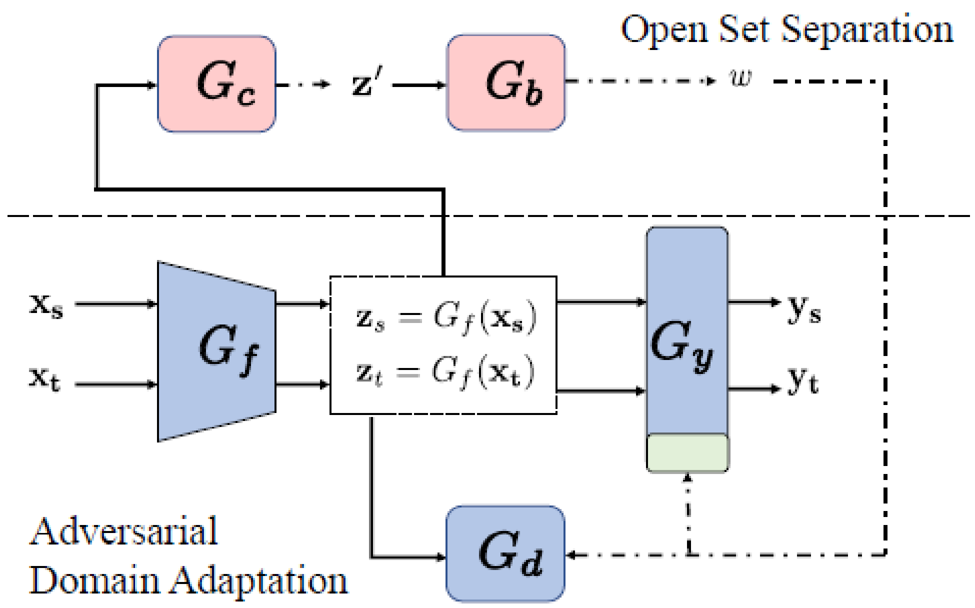


Figure 3. The proposed Separate to Adapt (STA) approach for open set domain adaptation, which is divided into two parts by the dashed line. The top part consists of a multi-binary classifier $G_c|_{c=1}^{|\mathcal{C}_s|}$ and a binary classifier G_b, which will generate the weights w for rejecting target samples in the unknown classes $\mathcal{C}_t \backslash \mathcal{C}_s$. The bottom part consists of feature extractor G_f, classifier G_y and domain discriminator G_d to perform adversarial domain adaptation between source and target data in the shared label space. $\mathbf{z}_s$ and $\mathbf{z}_t$ are the extracted deep features. $\hat{\mathbf{y}}_s$ and $\hat{\mathbf{y}}_t$ are the predicted labels. $\mathbf{z}'$ is the feature selected by G_c. The solid lines show the flow of tensors, and the dashdotted lines indicate the weighting mechanism.

- 虚线，上下两部分
  - $G_c|_{c=1}^{|\mathcal{C}_s|}$ 代表多二元分类器，G_b代表而分类器，用于生成未知类的权重w
  - G_f 表示特征提取模块，G_y 表示分类器，G_d 是域分类器

Progressive Separation

- G_c 的训练损失，用source data

$$L_s = \sum_{c=1}^{|\mathcal{C}_s|} \frac{1}{n_s} \sum_{i=1}^{n_s} L_{\text{bce}}\left(G_c\left(G_f\left(\mathbf{x}_i^s\right)\right), I\left(y_i^s, c\right)\right), \qquad (1)$$

where L_{bce} is the binary cross-entropy loss and $I\left(y_i^s, c\right) = 1$ if $y_i^s = c$ and $I\left(y_i^s, c\right) = 0$ otherwise. Each binary classifier

- G_c的output p_c可以看作是每个样本和当前类c的相似性
- 对于target 样本x_j^t，找到这个样本和哪个类最相似，计算相似性

$$s_j = \max_{c \in \mathcal{C}_s} G_c(G_f(\mathbf{x}_j^t)). \qquad (2)$$

filtering strategy one

- 接着，对所有的target 样本根据相似性排序，选择具有最高和最低相似性的样本来训练二分类器G_b

  存在疑问，最高和最低是只选了两个样本吗？还是选了最高和最低的分数，然后用这些分数对应的样本，去训练二分类器G_b, 感觉后者比较合理一点？
- 优点：
  - 由于只使用了相似性在极限值的数据，所以过滤相对粗糙但是拥有较高的可信度
  - 无需手动调整超参数，鲁棒性强

filtering strategy two

- 分数聚类，聚成 高 中 低 三类
- 使用高类中的均值s_h，作为阈值, $s_j >= s_h$的是已知类，使用低类中的均值s_l，作为阈值, $s_j <= s_l$的是未知类

  疑问，小于s_h 大于s_l 的，算啥？

  $\mathbf{X}'$ to denote the set of filtered samples by the multi-binary classifier, and d_j to indicate whether a target sample $\mathbf{x}_j \in \mathbf{X}'$ is labeled as known ($d_j = 0$) or unknown ($d_j = 1$), the fine-grained binary classifier G_c can be trained as follows,

$$L_b = \frac{1}{|\mathbf{X}'|} \sum_{\mathbf{x}_j \in \mathbf{X}'} L_{\text{bce}}\left(G_b\left(G_f\left(\mathbf{x}_j\right)\right), d_j\right). \qquad (3)$$

Weighted Adaptation

$$L_{\text{cls}}^{s}=\frac{1}{n_s}\sum_{\mathbf{x}_i\in\mathcal{D}_s}L_y\left(G_y^{1:|\mathcal{C}_s|}\left(G_f\left(\mathbf{x}_i\right)\right),y_i\right), \tag{4}$$

where L_y is cross-entropy loss, G_y is an *extended* classifier for $|\mathcal{C}_s|+1$ classes, i.e. the $|\mathcal{C}_s|$ known classes in the source domain plus the additional "unknown" class in the target domain. $G_y^{1:|\mathcal{C}_s|}$ denotes the probabilities corresponding to assigning each sample to the $|\mathcal{C}_s|$ known classes.

- 使用G_b的softmax输出作为instance-level权重，$w_j=G_b(G_f(x_j))$,w_j越大则属于未知类的概率越大

$$\begin{aligned}L_d=&\frac{1}{n_s}\sum_{\mathbf{x}_i\in\mathcal{D}_s}L_{\text{bce}}\left(G_d\left(G_f\left(\mathbf{x}_i\right)\right),d_i\right)\\&+\frac{1}{\sum_{\mathbf{x}_j\in\mathcal{D}_t}\left(1-w_j\right)}\sum_{\mathbf{x}_j\in\mathcal{D}_t}\left(1-w_j\right)L_{\text{bce}}\left(G_d\left(G_f\left(\mathbf{x}_j\right)\right),d_j\right).\end{aligned} \tag{5}$$

- 此外，还需要在**目标域中选取未知类的样本**，以**训练G_f获得额外的unknown类**

$$L_{\text{cls}}^{t}=\frac{1}{|\mathcal{C}_s|}\frac{1}{\sum_{\mathbf{x}_j\in\mathcal{D}_t}w_j}\sum_{\mathbf{x}_j\in\mathcal{D}_t}w_jL_y\left(G_y^{|\mathcal{C}_s|+1}\left(G_f\left(\mathbf{x}_j\right)\right),l_{\text{uk}}\right), \tag{6}$$

l_{uk} 是未知类，疑问，为什么这里也要加权呢?

- 对于已知类

$$L_e=\frac{1}{\sum_{\mathbf{x}_j\in\mathcal{D}_t}\left(1-w_j\right)}\sum_{\mathbf{x}_j\in\mathcal{D}_t}(1-w_j)H\left(G_y^{1:|\mathcal{C}_s|}\left(G_f\left(\mathbf{x}_j\right)\right)\right), \tag{7}$$

where H is the entropy loss and $H\left(\mathbf{p}\right)=-\sum_k p_k\log p_k$. It is noteworthy that we only aim to minimize the entropy of target samples estimated to be the known classes, so we use w_j as instance-level weight for the entropy minimization.

Training Procedure

- **known/unknown separation step**

  使用source data训练G_f G_y

  G_c G_b

$$(\hat{\theta}_f, \hat{\theta}_y, \hat{\theta}_b, \hat{\theta}_c|_{c=1}^{|\mathcal{C}_s|}) = \underset{\theta_f, \theta_y, \theta_b, \theta_c|_{c=1}^{|\mathcal{C}_s|}}{\arg\min} \quad L_{\text{cls}}^s + L_s + L_b. \quad (8)$$

- **weighted adversarial adaptation step**

  实现对抗性自适应，使**目标域中已知类的特征分布与源域保持一致**，并利用**未知类中的数据为额外类训练G_y**

$$(\hat{\theta}_y, \hat{\theta}_d) = \underset{\theta_y, \theta_d}{\arg\min}\, L_{\text{cls}}^s + L_{\text{cls}}^t + L_d + \lambda L_e, \qquad (9)$$

$$(\hat{\theta}_f) = \underset{\theta_f}{\arg\min}\, L_{\text{cls}}^s + L_{\text{cls}}^t - L_d + \lambda L_e, \qquad (10)$$

Experiment

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**

疑问，两个filtering strategy没有对比实验?

消融实验:

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**

1. w/o w（缺少对抗域训练的目标域样本的权重→对已知类和未知类的样本进行加权分离是必要的
2. w/o c（缺少多二元分类器中的softmax分类层）→多二元分类器可以产生更好的相似度，独立地度量目标样本与每个源类之间的关系
3. w/o b（缺少二元分类器G_b）→二元分类器可以根据多个二元分类器的结果来细化未知类和已知类样本之间的分离
4. w/o j（缺少Training Procedure中的两个steps的迭代）→联合分离和适应的有效性

遗留问题

- 关于 Adapting Object Detectors via Selective Cross-Domain Alignment 分组相关问题
  - 为什么要分组?

    这里主要是为了找到我们感兴趣的区域，一般使用RPN出来的候选框，但是

        - 候选框大小不一样，而作者希望获得**固定大小的区域**，以便于进一步处理（例如后面的patch 生成等）
        - 第二个是利用K-means聚类，能刨除一些**噪声**，比如只有背景的框就不要了

    - 分组之后?
        - 分组之后，要确定每个区域的特征，进行feature reassignment
        - 对于第k个分组中，里面有m_k个proposal，每个proposal的RoI特征的维度是d, m_k是不确定的，所以选定了一个超参m
        - 对选出来的m个proposal，进行拼接（论文里倒也没过多描述，我理解是一个cat操作？）

Feature Reassignment. Given the selected regions, we derive the feature representations thereof by reassigning the RoI features according to the grouping results. Specifically, each region is associated with a subset of region proposals assigned to the corresponding K-means cluster. By stacking the corresponding RoI features, we can obtain a matrix $\Theta_k \in \mathbb{R}^{m_k \times d}$ to represent the k-th region, where m_k is the number of region proposals assigned to the k-th cluster, and d is the feature dimension.

This representation is inconvenient to work with, as the number m_k can vary. It is desirable to fix the number of features. For this purpose, we adopt a simple select-or-copy scheme. Given a pre-defined number m, if m_k is greater than m, we retain only the top-m features; if m_k is less than m, we simply make copies of the assigned features until we get enough. In this way, we can derive a fixed number of features $\hat{\Theta}_k \in \mathbb{R}^{m \times d}$ to represent each region.

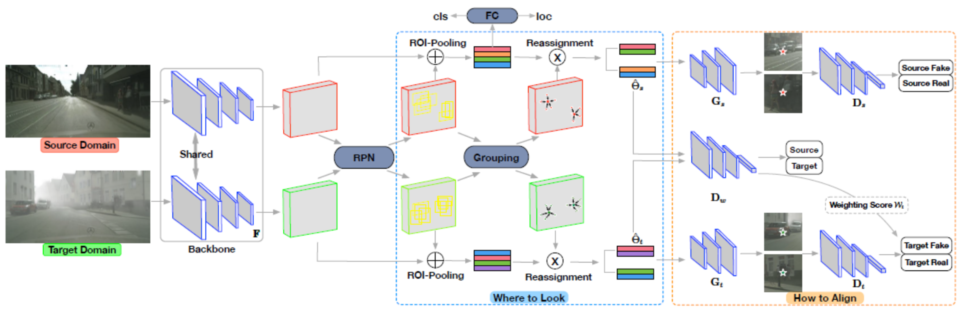
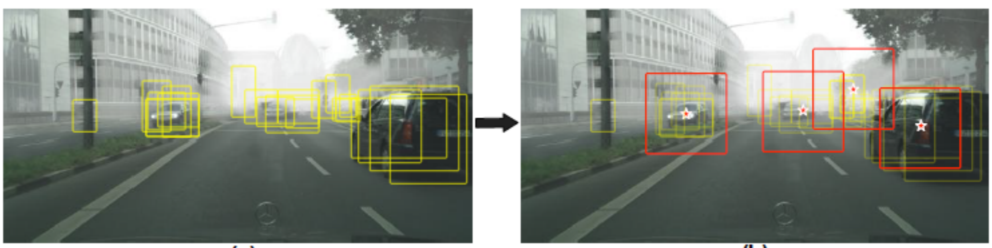


Figure 2. The pipeline of our framework. Two major components, *i.e.* "Where to Look" and "How to Align" are illustrated with two dashed rectangles. For the first component, an ROI-based grouping strategy is designed to mine the discriminative regions for two domains. We display the grouping procedure with cluster number = 2 (Note that ★ and ★ denote the centroids of clusters). For the second one, our model performs the adjusted region-level alignment using generators (G_s and G_t), discriminators (D_s and D_t) and weighting estimator (D_w). We use Faster R-CNN as the detection model ($\mathbb{F}$) which consists of the backbone, RPN and head part. (**Best viewed in color**)



(a) (b)