Partial Adversarial Domain Adaptation

$Zhangjie\ Cao, et\ al.$, ECCV 2018

2019/03/19, KangYeol Kim

DAVIAN
Data and Visual Analytics Lab

Domain Adaptation(DA)

- **We cannot get massive data in every DOMAIN!!**
    - Lack of domain knowledge for labeling (e.g. Biomedical)
    - Taking so much time to get sufficient data (e.g. Semantic segmentation)
    - Specific domain where data is rare

- Thus, we want to **borrow knowledge** from the domain where massive data are available
    - Multi-task Learning with available data
    - **Transfer Learning** (e.g. pretrained model with ImageNet data)

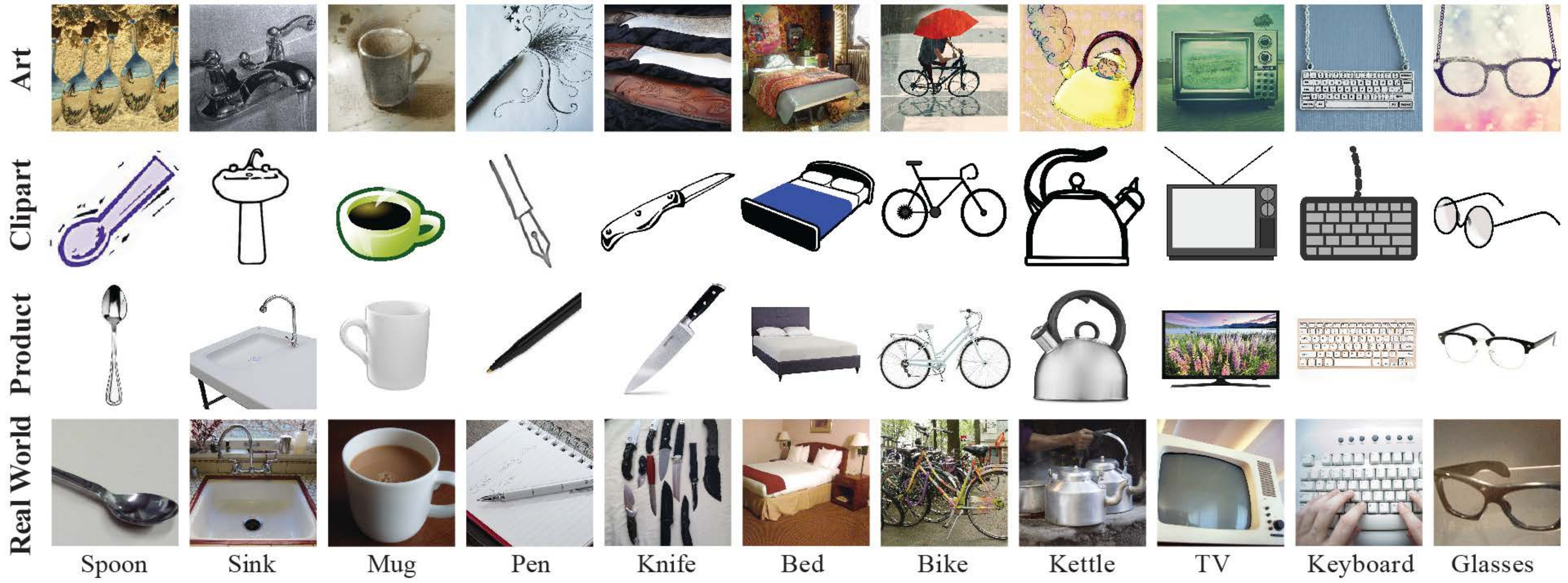
→ Domain Adaptation is a branch of "Transfer learning"

Domain Adaptation(DA)

Domain Adaptation's objectives

- From source domain(Big data) to target domain(Small data)
- Build a well-performed model in target domain with (Source + Target) data
- Source data have label etc. information
    - Target data have condition : Supervised DA / Semi-supervised DA / Unsupervised DA

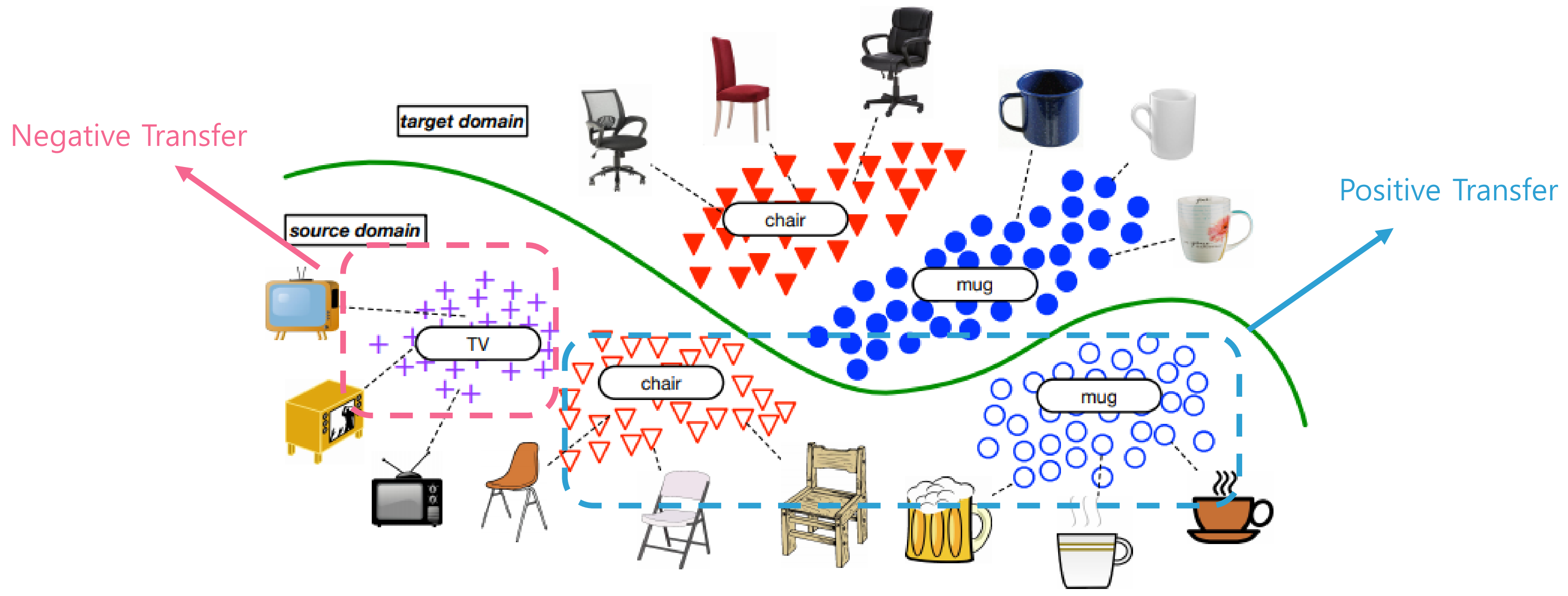
By Domain Adaptation,
we want to align the **feature distribution of source domain and that of target domain**



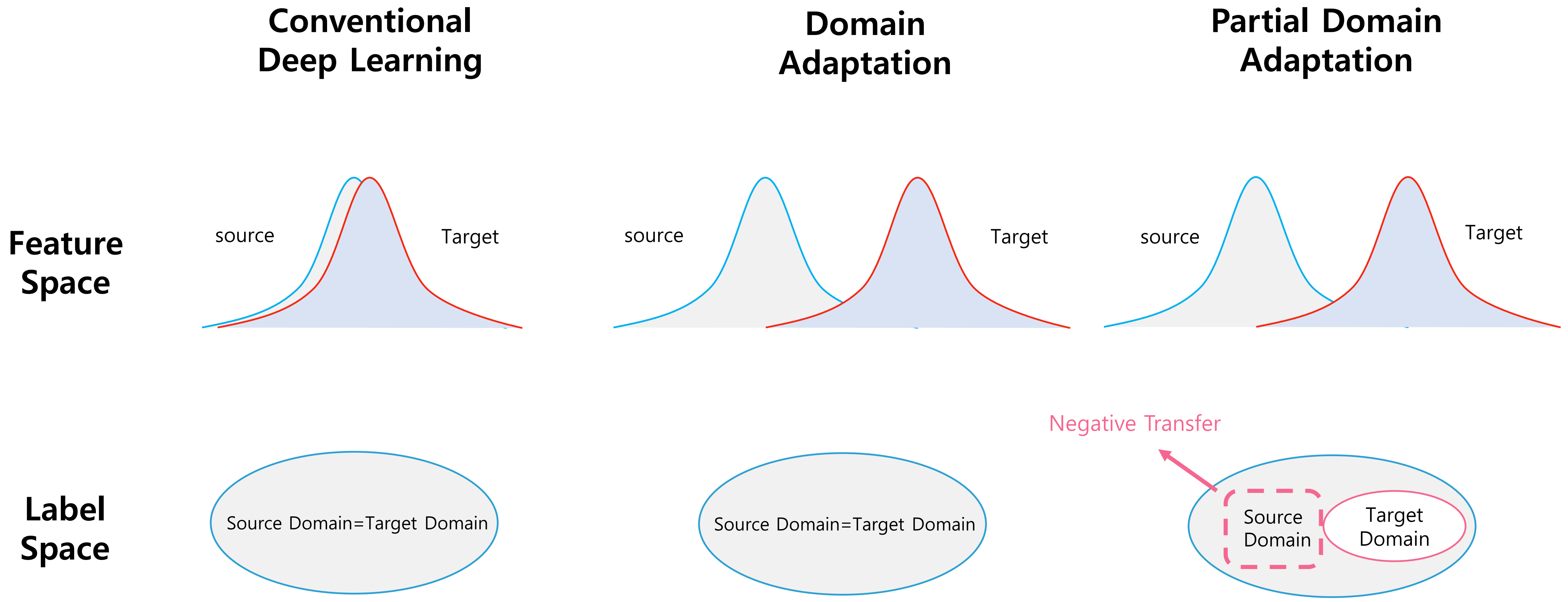
Introduction (1/3)

Problem Definition

- This paper want to solve unsupervised domain adaptation
- DA assumes that **domains share identical label space** but follow different distributions. However, most case, **label space of source domain is not identical to that of target domain**
- As a result, outlier class in source domain (train)can impair model's goal of feature distribution alignment on target domain data(test data) which is called **"Negative transfer"**

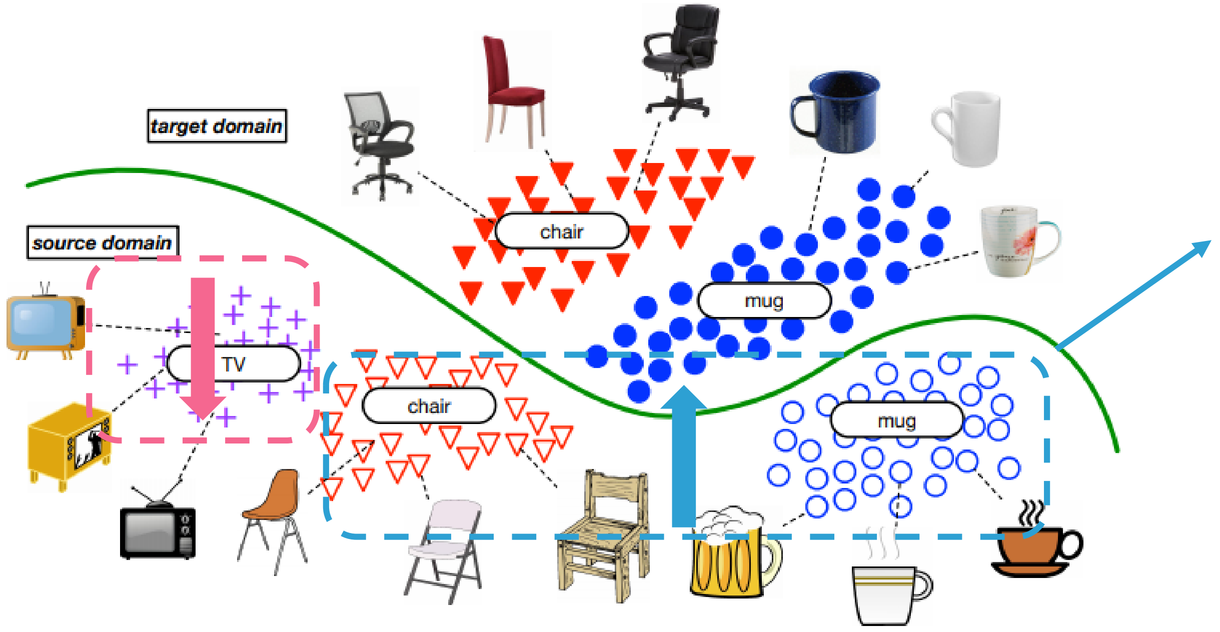


Introduction (2/3)

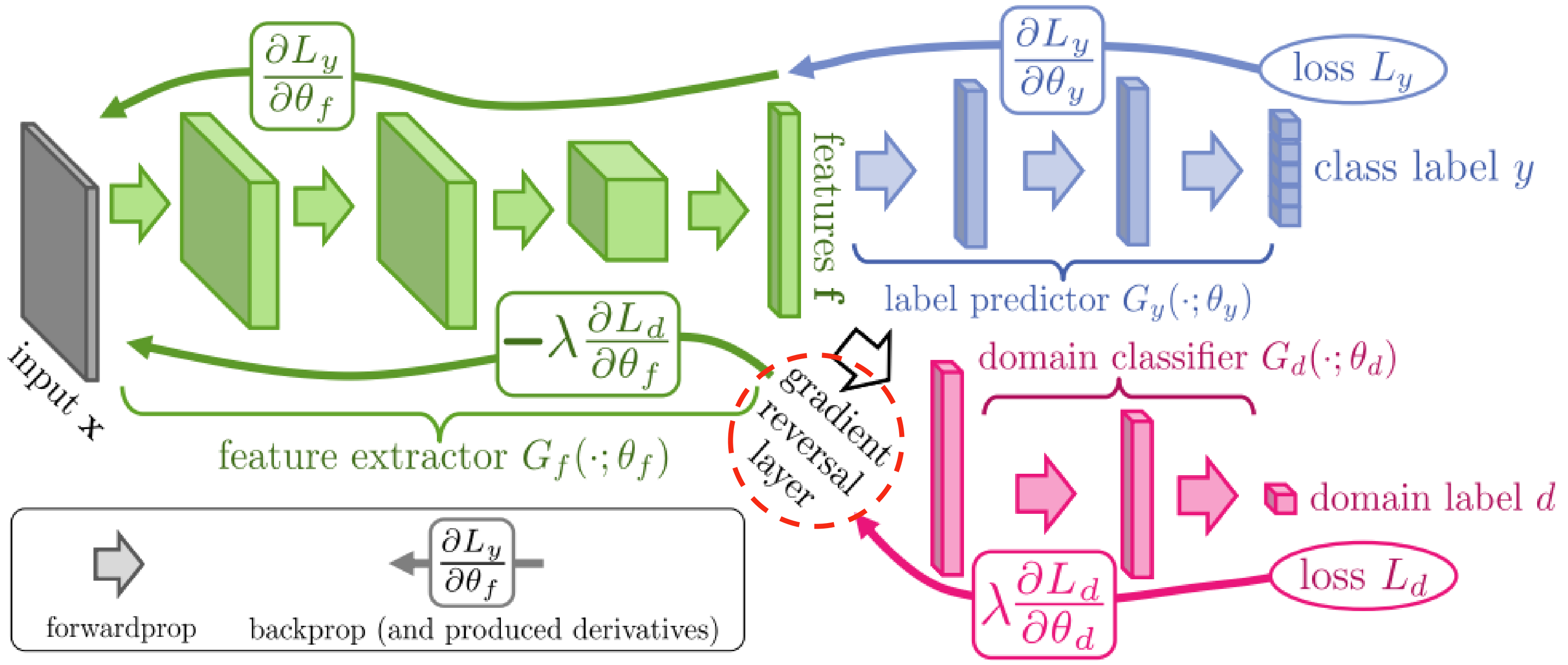


Introduction (3/3)

- PADA **aligns the feature distributions** of the source and target data in the **shared label space**
- **PADA identifies the irrelevant source data** belonging to the outlier source classes and **down-weighs their importance automatically**



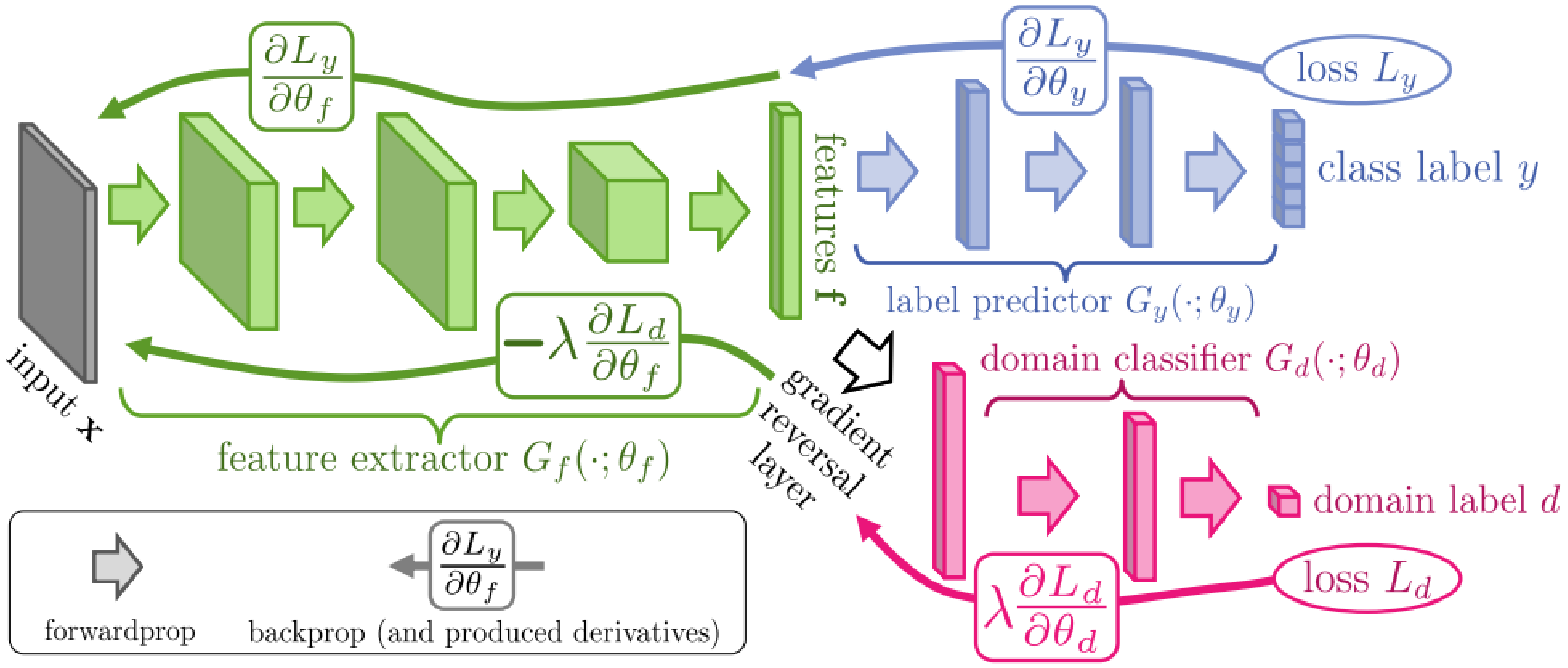
Domain Adversarial NN


$\frac{\partial L_y}{\partial \theta_f}$
$\frac{\partial L_y}{\partial \theta_y}$
loss L_y
features $\mathbf{f}$
class label y
label predictor $G_y(\cdot;\theta_y)$
$-\lambda\frac{\partial L_d}{\partial \theta_f}$
input $\mathbf{x}$
gradient reversal layer
domain classifier $G_d(\cdot;\theta_d)$
domain label d
feature extractor $G_f(\cdot;\theta_f)$
$\lambda\frac{\partial L_d}{\partial \theta_d}$
loss L_d
forwardprop
backprop (and produced derivatives)

Intuition on mechanism:

- Features should **be able to predict label**
- Feature should not be able to discriminate domains

Domain Adversarial NN



$\frac{\partial L_y}{\partial \theta_f}$
$\frac{\partial L_y}{\partial \theta_y}$
loss L_y
class label y
features $\mathbf{f}$
label predictor $G_y(\cdot;\theta_y)$
$-\lambda \frac{\partial L_d}{\partial \theta_f}$
input $\mathbf{x}$
feature extractor $G_f(\cdot;\theta_f)$
gradient reversal layer
domain classifier $G_d(\cdot;\theta_d)$
domain label d
$\lambda \frac{\partial L_d}{\partial \theta_d}$
loss L_d
forwardprop
backprop (and produced derivatives)

Discriminativeness + **Domain-invariance(@GRL)**

$$C_0(\theta_f, \theta_y, \theta_d) = \frac{1}{n_s} \sum_{\mathbf{x}_i \in \mathcal{D}_s} L_y\left(G_y\left(G_f\left(\mathbf{x}_i\right)\right), y_i\right) - \frac{\lambda}{n_s + n_t} \sum_{\mathbf{x}_i \in \mathcal{D}_s \cup \mathcal{D}_t} L_d\left(G_d\left(G_f\left(\mathbf{x}_i\right)\right), d_i\right)$$

$$\left(\boxed{\hat{\theta}_f}, \hat{\theta}_y\right) = \arg\min_{\theta_f, \theta_y} C_0\left(\theta_f, \theta_y, \theta_d\right),$$

$$\left(\hat{\theta}_d\right) = \arg\max_{\theta_d} C_0\left(\theta_f, \theta_y, \theta_d\right).$$

Partial Adversarial DA

(1) How to down-weigh outlier classes in source domain?

- The probabilities of assigning the target data to the source outlier classes should be small
- Averaging the label predictions $\hat{y}_i$ on all target data and scaling to 0-1
- Update of γ takes place at regular iteration intervals

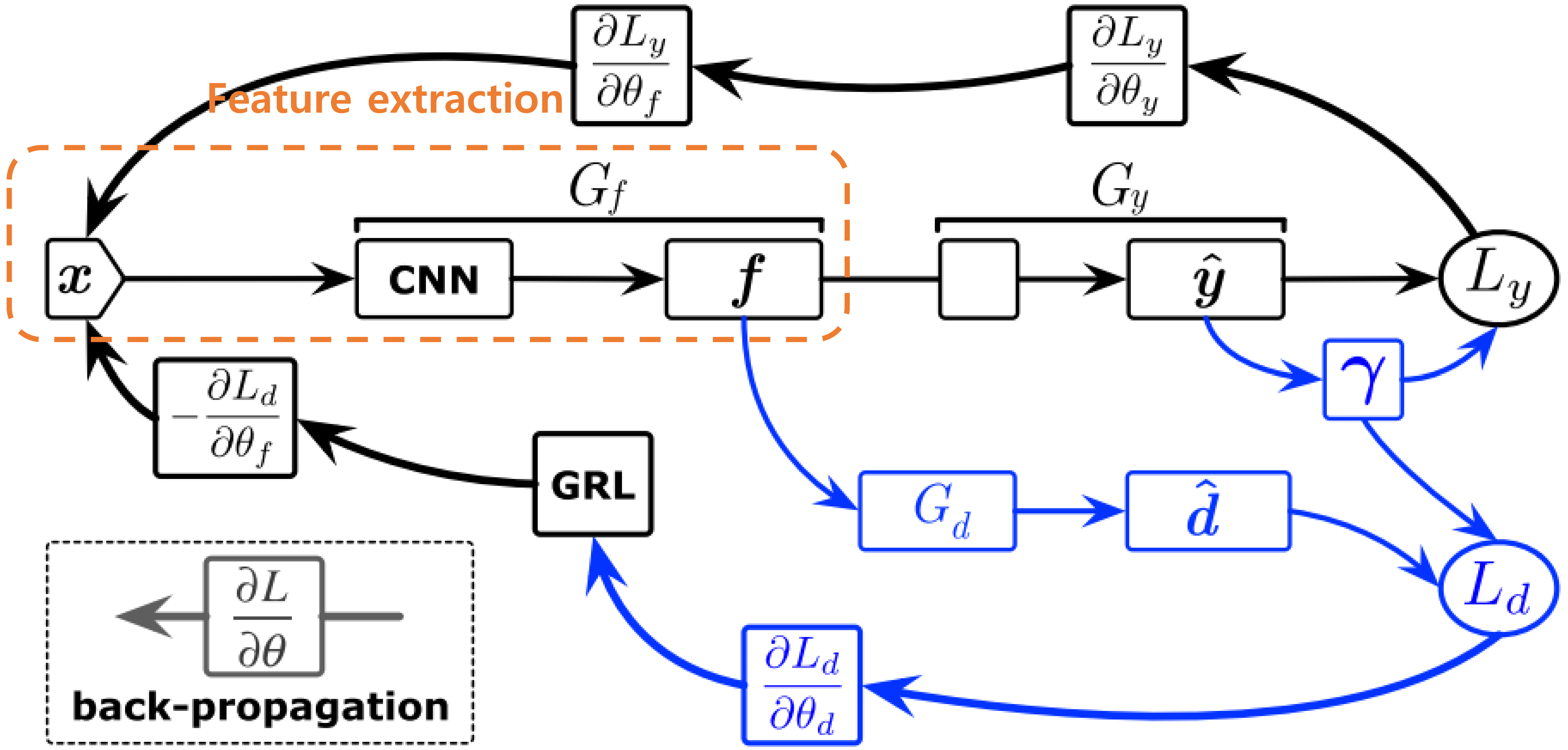
$$\gamma = \frac{1}{n_t} \sum_{i=1}^{n_t} \hat{\mathbf{y}}_i,$$

(2) Objective function

$$\begin{aligned} C\left(\theta_f, \theta_y, \theta_d\right) = & \frac{1}{n_s} \sum_{\mathbf{x}_i \in \mathcal{D}_s} \gamma_{y_i} L_y\left(G_y\left(G_f\left(\mathbf{x}_i\right)\right), y_i\right) \\ & - \frac{\lambda}{n_s} \sum_{\mathbf{x}_i \in \mathcal{D}_s} \gamma_{y_i} L_d\left(G_d\left(G_f\left(\mathbf{x}_i\right)\right), d_i\right) \\ & - \frac{\lambda}{n_t} \sum_{\mathbf{x}_i \in \mathcal{D}_t} L_d\left(G_d\left(G_f\left(\mathbf{x}_i\right)\right), d_i\right) \end{aligned}$$

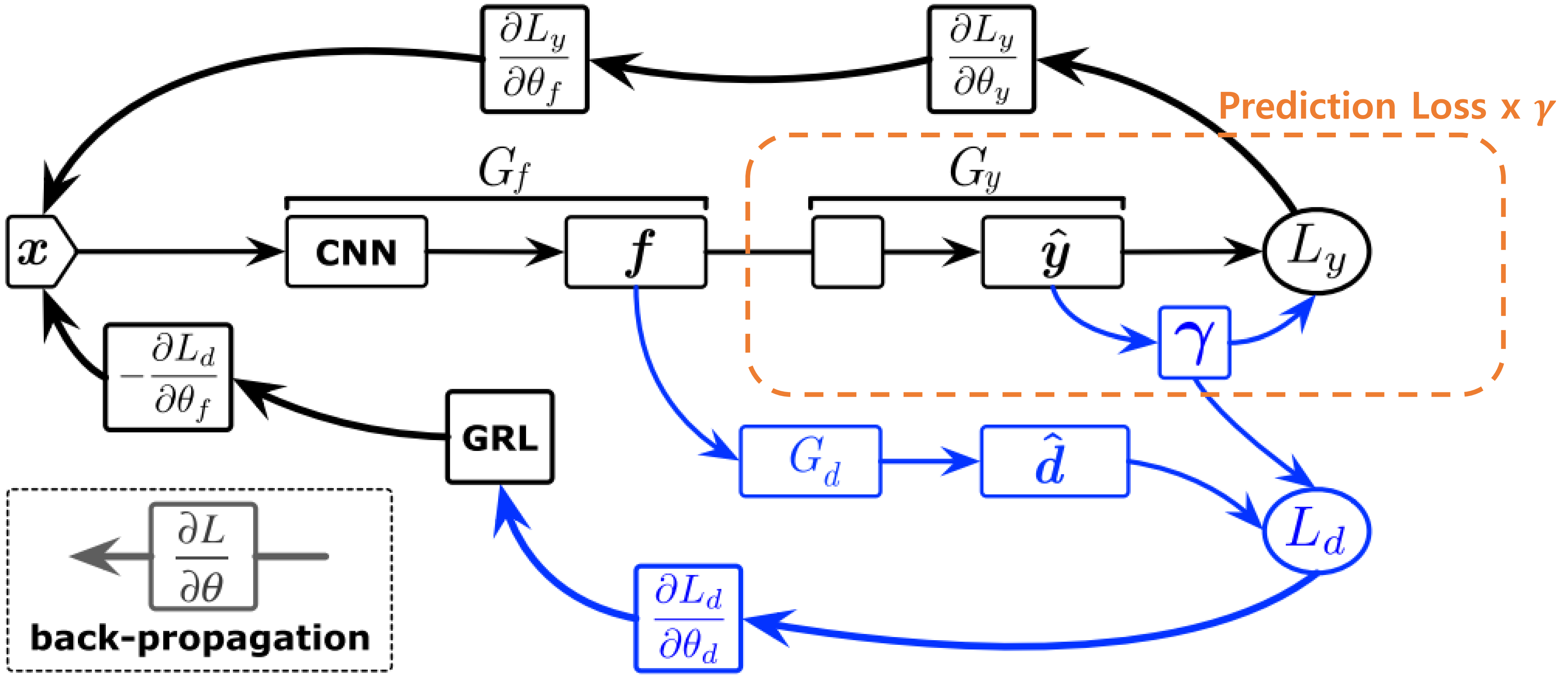
$$\begin{aligned} \left(\hat{\theta}_f, \hat{\theta}_y\right) &= \arg\min_{\theta_f, \theta_y} C\left(\theta_f, \theta_y, \theta_d\right), \\ \left(\hat{\theta}_d\right) &= \arg\max_{\theta_d} C\left(\theta_f, \theta_y, \theta_d\right). \end{aligned}$$

Partial Adversarial DA

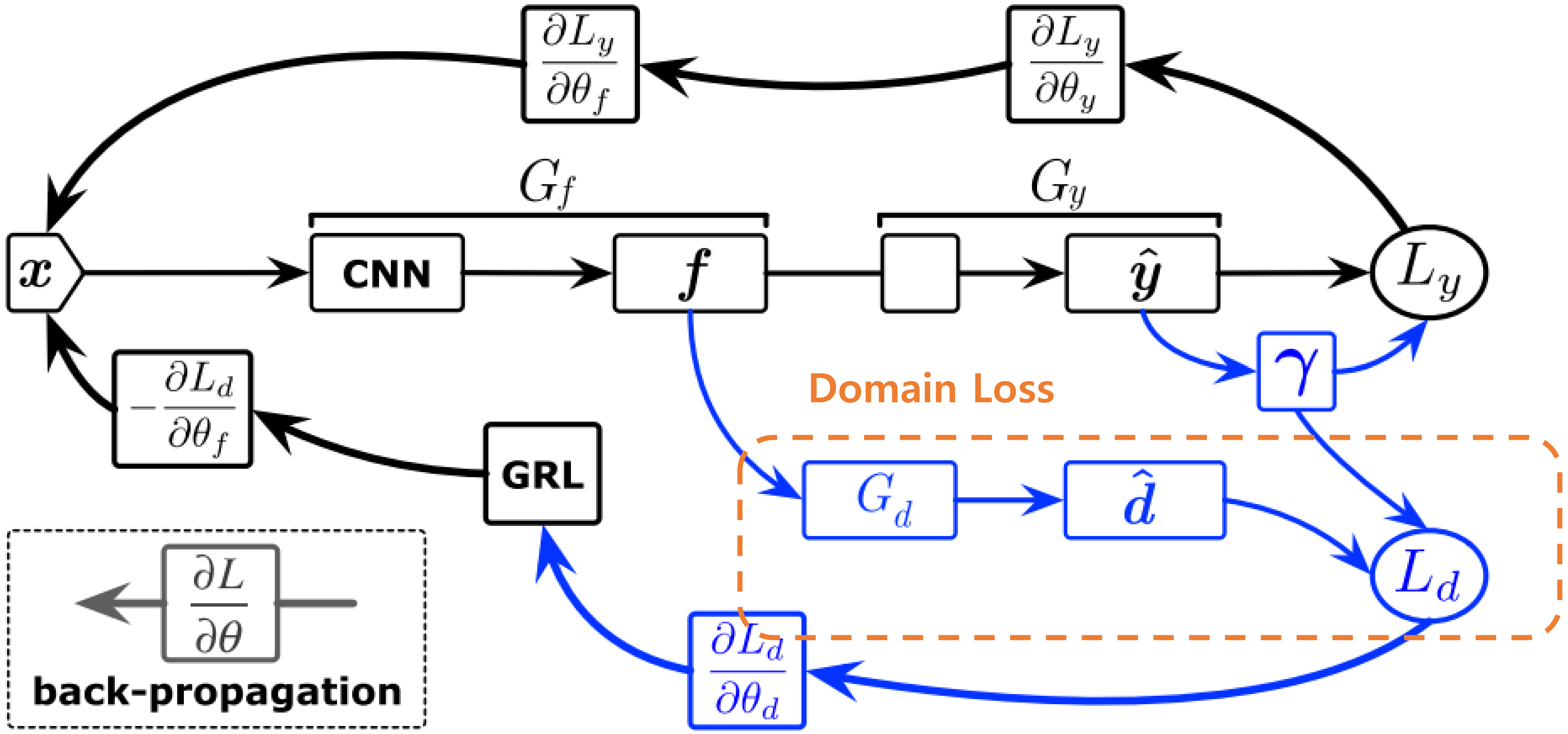


$\frac{\partial L_y}{\partial \theta_f}$
$\frac{\partial L_y}{\partial \theta_y}$
Feature extraction
G_f
G_y
x
CNN
f
$\hat{y}$
L_y
γ
$-\frac{\partial L_d}{\partial \theta_f}$
GRL
G_d
$\hat{d}$
L_d
$\frac{\partial L}{\partial \theta}$
back-propagation
$\frac{\partial L_d}{\partial \theta_d}$

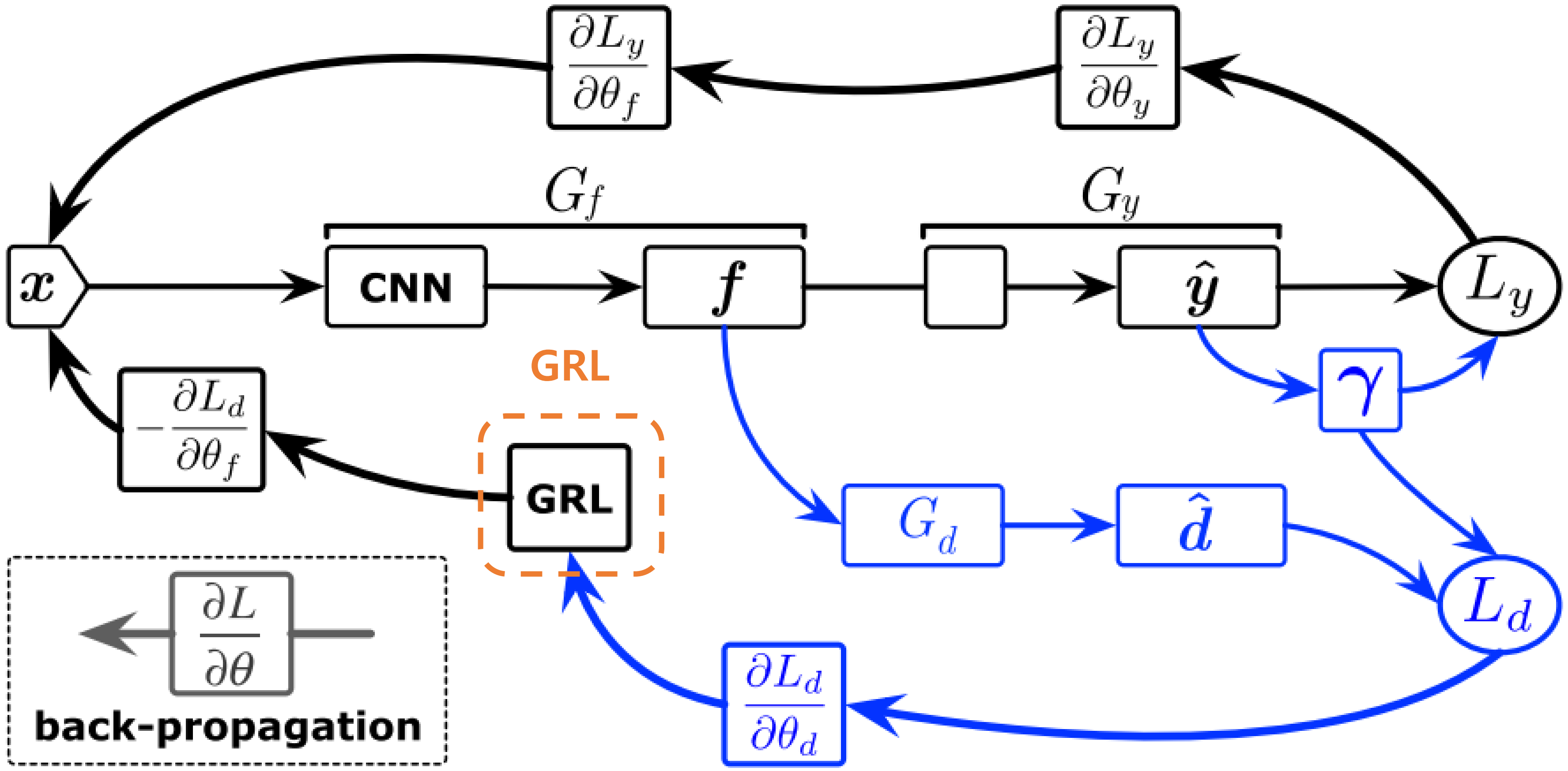
Partial Adversarial DA


$\frac{\partial L_y}{\partial \theta_f}$
$\frac{\partial L_y}{\partial \theta_y}$
Prediction Loss x γ
G_f
G_y
x
CNN
f
$\hat{y}$
L_y
γ
$-\frac{\partial L_d}{\partial \theta_f}$
GRL
G_d
$\hat{d}$
L_d
$\frac{\partial L}{\partial \theta}$
back-propagation
$\frac{\partial L_d}{\partial \theta_d}$

Partial Adversarial DA


$\frac{\partial L_y}{\partial \theta_f}$
$\frac{\partial L_y}{\partial \theta_y}$
G_f
G_y
x
CNN
f
$\hat{y}$
L_y
γ
Domain Loss
$-\frac{\partial L_d}{\partial \theta_f}$
GRL
G_d
$\hat{d}$
L_d
$\frac{\partial L}{\partial \theta}$
back-propagation
$\frac{\partial L_d}{\partial \theta_d}$

Partial Adversarial DA


$\frac{\partial L_y}{\partial \theta_f}$
$\frac{\partial L_y}{\partial \theta_y}$
G_f
G_y
x
CNN
f
$\hat{y}$
L_y
GRL
$-\frac{\partial L_d}{\partial \theta_f}$
GRL
γ
G_d
$\hat{d}$
L_d
$\frac{\partial L}{\partial \theta}$
back-propagation
$\frac{\partial L_d}{\partial \theta_d}$

Experiments

- **Office-31** –31 classes from 3 domains : $Amazon(A)$ / $Webcam(W)$ / $DSLR(D)$

In this paper, $A(31\ classes) \rightarrow\ W\ (10\ classes)$ …

- **Office-Home** – 65 classes from 4 domains : $Artistic(Ar)$ / $Clipart(CI)$ / $Product(\mathrm{Pr})$ / $Real(Rw)$

In this paper, $Ar(65\ classes) \rightarrow\ CI(25\ classes)$ …

- **ImageNet-Caltech** – 84 shared classes

In this paper, $ImageNet - 1K \rightarrow\ Caltech - 84, Caltech - 256 \rightarrow\ ImageNet - 84$

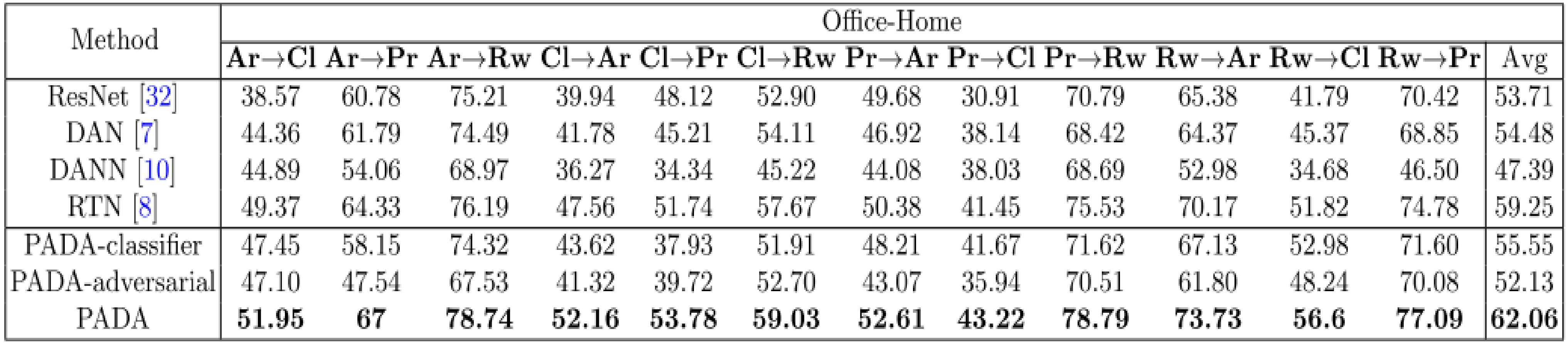
- **VisDA2017** – 12classes from 2 domains : $Real$ / $Synthetic$

In this paper, $Real - 12 \rightarrow Synthetic - 6, Synthetic - 12 \rightarrow Real - 6$

ResNet-50, adding bottleneck layer between the res5c and fc layers

Experiments

Table 1. Accuracy of partial domain adaptation tasks on *Office-Home* (ResNet-50)



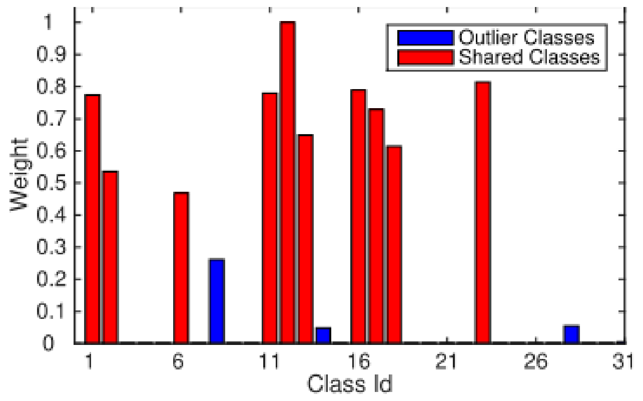
MMD
AD
MMD

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 [32]	38.57	60.78	75.21	39.94	48.12	52.90	49.68	30.91	70.79	65.38	41.79	70.42	53.71
DAN [7]	44.36	61.79	74.49	41.78	45.21	54.11	46.92	38.14	68.42	64.37	45.37	68.85	54.48
DANN [10]	44.89	54.06	68.97	36.27	34.34	45.22	44.08	38.03	68.69	52.98	34.68	46.50	47.39
RTN [8]	49.37	64.33	76.19	47.56	51.74	57.67	50.38	41.45	75.53	70.17	51.82	74.78	59.25
PADA-classifier	47.45	58.15	74.32	43.62	37.93	51.91	48.21	41.67	71.62	67.13	52.98	71.60	55.55
PADA-adversarial	47.10	47.54	67.53	41.32	39.72	52.70	43.07	35.94	70.51	61.80	48.24	70.08	52.13
PADA	**51.95**	**67**	**78.74**	**52.16**	**53.78**	**59.03**	**52.61**	**43.22**	**78.79**	**73.73**	**56.6**	**77.09**	**62.06**

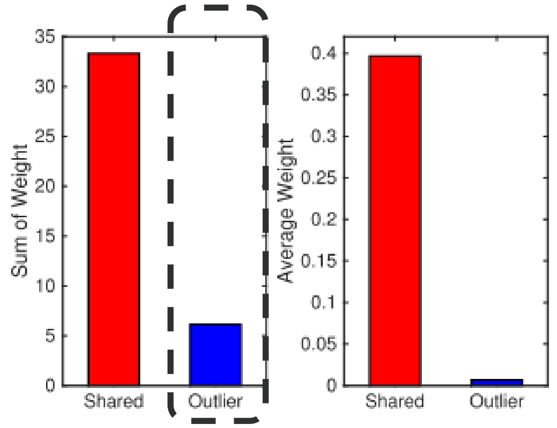
(1) All of baseline model suffer from **negative transfer**

(2) PADA outperforms PADA-classifier(PADA without γ in classifier part) / PADA-adversarial(PADA without γ in adversarial part)

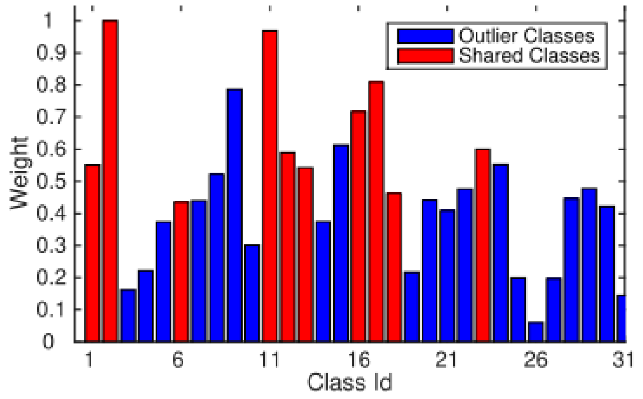
Empirical Analysis



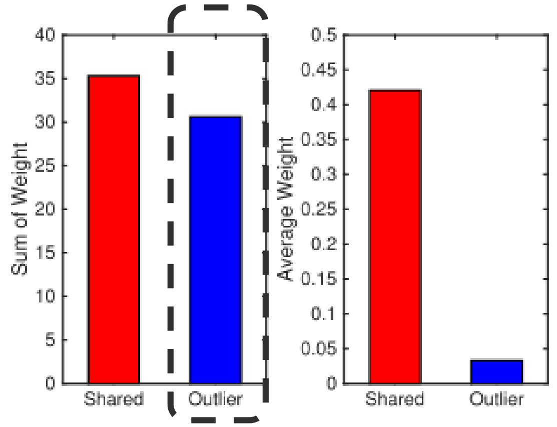
(a) PADA: **A→W**



(b) PADA: **ImageNet-1K→Caltech-84**

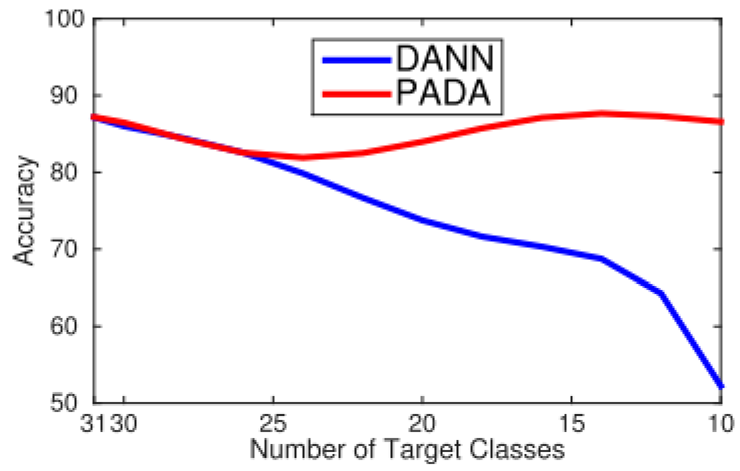


(c) DANN: **A→W**

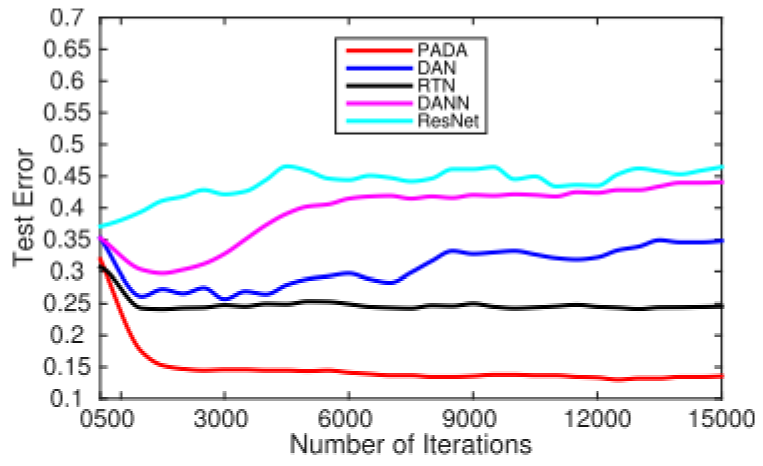


(d) DANN: **ImageNet-1K→Caltech-84**

Empirical Analysis



(a) Acc w.r.t #Target Classes



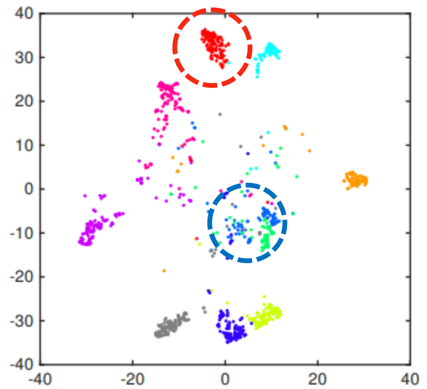
(b) Test Error

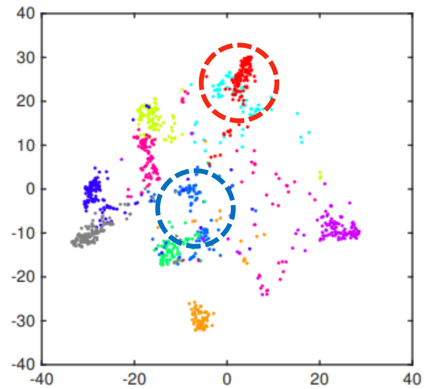
[# of Target Classes] The performance of DANN degrades quickly by negative transfer effect

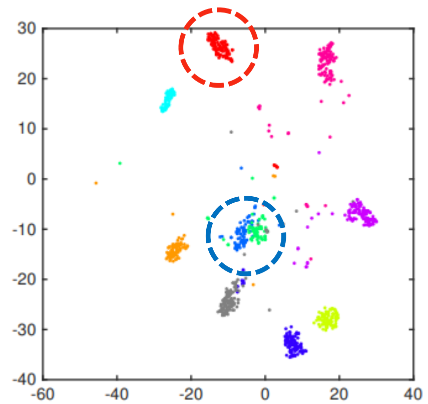
[Convergence] PADA converges fast and stably to the lowest test error,

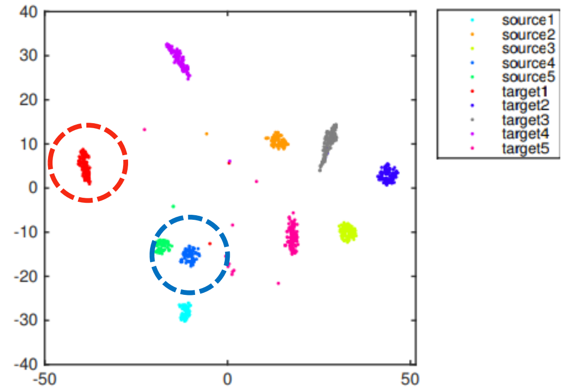
Empirical Analysis

Class info.(Not Shared − Shared)


(a) DAN

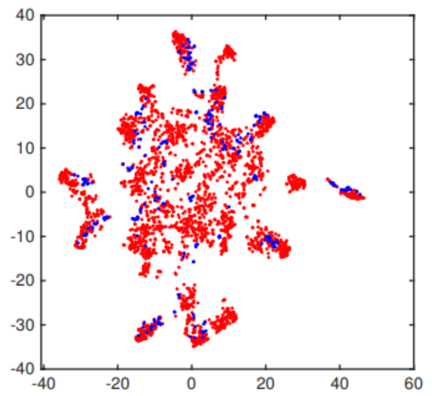

(b) DANN

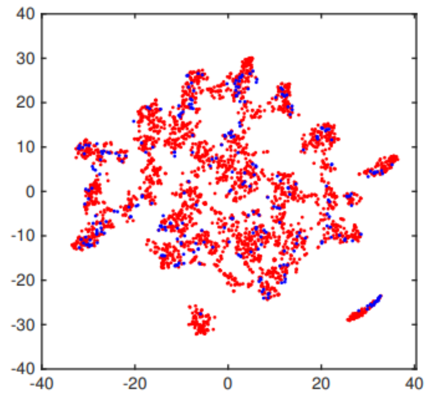

(c) RTN

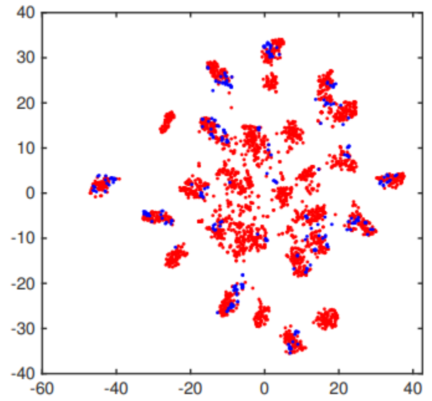


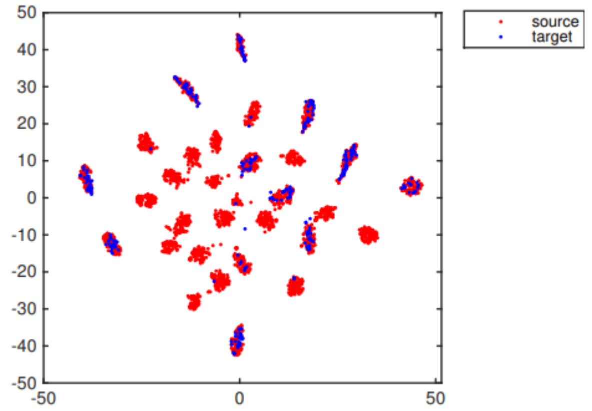
(d) PADA

Domain info.(31c)


(a) DAN

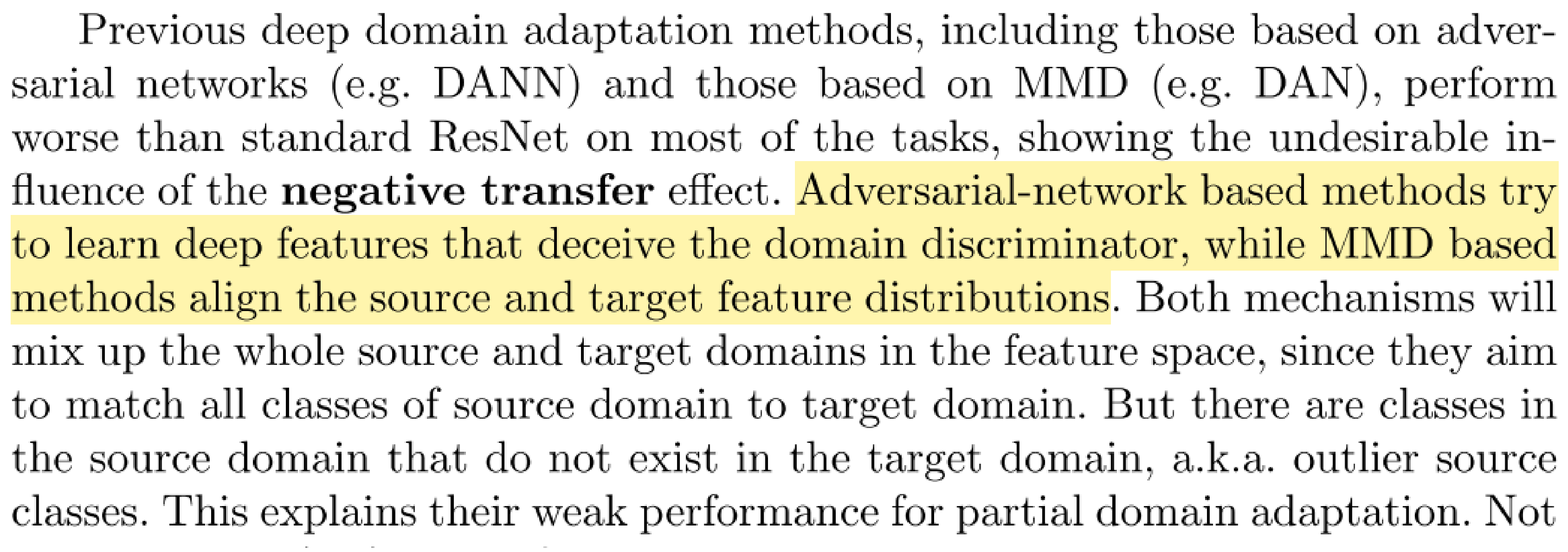

(b) DANN

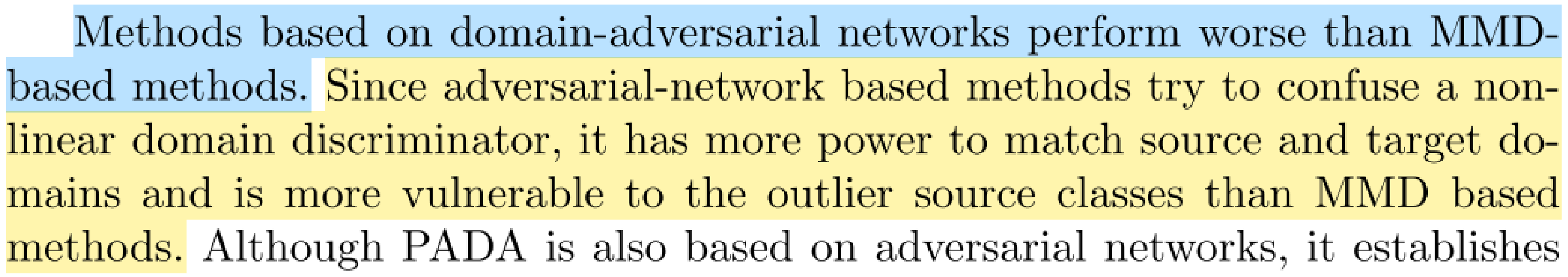

(c) RTN



(d) PADA

My Question


Previous deep domain adaptation methods, including those based on adversarial networks (e.g. DANN) and those based on MMD (e.g. DAN), perform worse than standard ResNet on most of the tasks, showing the undesirable influence of the **negative transfer** effect. Adversarial-network based methods try to learn deep features that deceive the domain discriminator, while MMD based methods align the source and target feature distributions. Both mechanisms will mix up the whole source and target domains in the feature space, since they aim to match all classes of source domain to target domain. But there are classes in the source domain that do not exist in the target domain, a.k.a. outlier source classes. This explains their weak performance for partial domain adaptation. Not


Methods based on domain-adversarial networks perform worse than MMD-based methods. Since adversarial-network based methods try to confuse a non-linear domain discriminator, it has more power to match source and target domains and is more vulnerable to the outlier source classes than MMD based methods. Although PADA is also based on adversarial networks, it establishes

Thank You