Discriminative Partial Domain Adversarial Network

Jian Hu[1], Hongya Tuo[1⋆], Chao Wang[2], Lingfeng Qiao[1], Haowen Zhong[1]
Junchi Yan[3][0000−0001−9639−7679]⋆⋆, Zhongliang Jing[1], Henry Leung[4]

[1] School of Aeronautics and Astronautics, Shanghai Jiao Tong University
[2] Alibaba Group
[3] Department of Computer Science and Engineering, Shanghai Jiao Tong University
[4] Department of Electrical and Computer Engineering, University of Calgary, Canada
{jianhu18,tuohy,qiaolf927,zhonghaowen, yanjunchi,zljing}@sjtu.edu.cn
xiaoxuan.wc@alibaba-inc.com,leungh@ucalgary.ca

Abstract. Domain adaptation (DA) has been a fundamental building block for Transfer Learning (TL) which assumes that source and target domain share the same label space. A more general and realistic setting is that the label space of target domain is a subset of the source domain, as termed by Partial domain adaptation (PDA). Previous methods typically match the whole source domain to target domain, which causes negative transfer due to the source-negative classes in source domain that does not exist in target domain. In this paper, a novel Discriminative Partial Domain Adversarial Network (DPDAN) is developed. We first propose to use hard binary weighting to differentiate the source-positive and source-negative samples in the source domain. The source-positive samples are those with labels shared by two domains, while the rest in the source domain are treated as source-negative samples. Based on the above binary relabeling strategy, our algorithm maximizes the distribution divergence between source-negative samples and all the others (source-positive and target samples), meanwhile minimizes domain shift between source-positive samples and target domain to obtain discriminative domain-invariant features. We empirically verify DPDAN can effectively reduce the negative transfer caused by source-negative classes, and also theoretically show it decreases negative transfer caused by domain shift. Experiments on four benchmark domain adaptation datasets show DPDAN consistently outperforms state-of-the-art methods.

Keywords: partial domain adaptation; adversarial learning; discriminative learning

⋆ Correspondence author is Hongya Tuo, this work is supported by National Natural Science Foundation of China(Grant No.61673262 and 61175028) and Shanghai key project of basic research(Grant No.16JC1401100)
⋆⋆ Partial work was done when Junchi Yan was with Tencent AI Lab as visiting scholar.

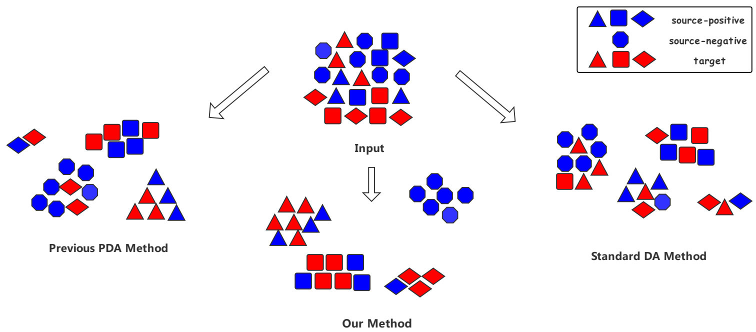


Fig. 1. The main difference of our method against the previous standard domain adaption and partial domain adaption mathods. The blue samples are from source domain and the red ones are from target domain. In standard DA method, source-negative classes can confuse discriminator, leading to performance degeneration. Previous PDA methods can select out most negative classes, but some are still hard to distinguish. Our method not only utilizes hard binary weight narrows the distribution divergence between source-positive and target samples, but also widens the distance between source-negative samples and others to reduce negative transfer caused by domain shift.

1 Introduction

Deep neural networks have show the excellence in many fields [7, 13, 20] such as computer vision, natural language processing. However, all these applications rely on a huge amount of labeled data. In practice, there may not be enough labeled data for training from scratch. Transfer learning has been considered as one of the most representative methods to deal with this problem [10].

One critical issue in transfer learning algorithms is domain shift in data distribution [22]. Domain adaptation (DA) is a traditional method for transfer learning to learn domain-invariant features to close the gap between domains. In this way, source classifier can be utilized to classify target samples without labels. Recent researches have shown that deep networks can learn more transferable features to bring the gap among different domains [31].

The basic assumption of standard DA is that source and target domain share the same label space [19, 30], which often does not hold in practice. One more general condition is that the label space of target domain is a subset of source domain. Hence, present standard domain adaptation methods are not available. Recently, a practical scenario named as partial domain adaptation (PDA) has been proposed [1, 2, 32]. It assumes the source label space contains target label space. PDA aims to transfer knowledge from a large domain with sufficient labels to a target domain with few labels. For example, the large-scale labeled dataset like ImageNet-1K [21] can be seen as a source domain and the related real-world dataset as target domain. We define positive classes as those shared by both source and target labels, and negative classes as only belonging to source space.

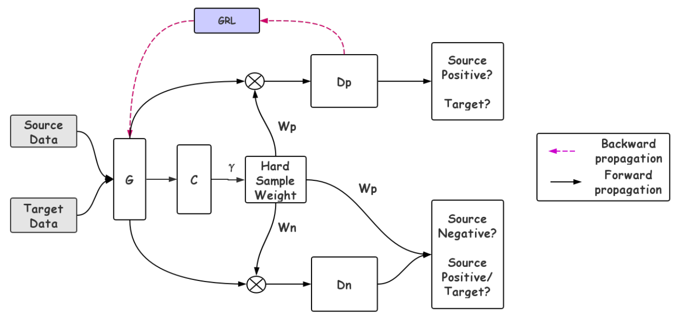


Fig. 2. Architecture of our method, G is the feature extractor, and C is not only the source classifier but also the weight discriminator to obtain the source class importance weights. The outputs of C when it plays the role of weight discriminator is γ. γ evaluates the possibility of the source class belonging to positive classes. We further transfer γ to w_p and w_n, which represents the possibility of source samples belonging to C_{sp} and C_{sn} respectively. D_p tries to narrow p_{sp} and p_t, and D_n tries to widen p_{sn} and others.

In PDA, one challenge is that we do not know which part of source label space is shared with target space, because negative classes are unavailable during training. Intuitively, when target label space contains only a subset of source classes, it is impossible to reduce the domain shift by comparing source and target distributions. Methods are devised [1,2,32] to reduce label space's mismatch by weighting each sample into the domain adversarial network. However, these approaches all use possibility based soft-weights to select positive classes. In the ideal situation, the weights of positive classes are expected to be 1, otherwise 0. In practice, even if the weights of positive classes are obviously higher than negative, few of them can achieve the expected values. Moreover, previous methods mainly focus on narrowing the distance between source positive and target samples, but ignore zooming out the distance between source negative and other samples. These two phenomenons lead to severe negative transfer.

To solve the deficiency caused by soft-weight, it is hoped not only to extract the common features from source-positive and target domain, but also avoid misjudging source positive and negative classes. Based on related analysis from PADA [2], since the source-negative label space and target label space are disjoint, the target data should be dissimilar to the source data in the negative label space. Hence, the probabilities of assigning the target data to the source negative classes should be sufficiently small. In another word, the weights from source positive domain are much higher than these from source negative ones. Therefore, we put forward the hard binary weights to adaptively divide source domain samples into positive and negative classes, weights of the positive classes are 1, otherwise are 0. In this way, our model can not only distinguish the positive class from the negative class in the source domain, but also give the positive class enough weight to eliminate negative transfer. Furthermore, we propose discriminative PDA Net to widen the distance between source negative classes and others, further reducing the influence of negative classes in source domain.

As illustrated in Fig. 1, we have presented a Discriminative Partial Domain Adversarial Network for PDA. Our contributions are three-folds.

For partial domain adaption, we show the benefit to incorporate the source-negative samples absent in the target domain, which has been rarely considered before. Specifically, we are the first to propose maximizing the distribution divergence between source-negative samples and the others (source-positive and target samples), to reduce negative transfer caused by domain shift.

To promote this maximization, we propose to use binary hard labels to distinguish source-negative and source-positive samples, in contrast to the soft-weighting used in previous works. The binary label strategy is also used to effectively narrow the distance between source-positive and target samples, leading to our main approach for jointly discriminative learning for partial domain adaption via adversarial networks. Furthermore, soft weights are still retained on the source classifier to ensure that the misclassified source samples can be corrected.

We theoretically prove that the proposed DPDAN is guaranteed to simultaneously achieve probability distribution alignment and prevent negative transfer. Extensive experimental results show the competitive performance of our approach on public datasets and the efficacy of the proposed components.

2 Related Works

Domain Adaptation DA plays an important role in transfer learning. It tries to narrow the gap between source and target domain by learning domain-invariant features. DA frees target domain from expensive label cost.

Deep neural networks ensure knowledge transfer by learning high-level domain-invariant features. But distribution discrepancy across different domains cannot be eliminated completely by utilizing deep neural networks alone. Some DA methods utilize high-level statistical features [4, 15, 28, 33] to match domains. Some help feature extractor to learn domain-invariant features in a min-max game by adding a discriminator [5, 17, 25, 26].

Partial Domain Adaptation PDA is a generalization of standard domain adaptation. PDA assumes target label space is a subset of source label space. Three approaches have been proposed to deal with it. Selective Adversarial Network (SAN) [1] utilizes weight evaluators for each class to select out negative classes in source samples, then the weighted source samples are adopted in multiple adversarial network. Partial Adversarial Domain Adaptation (PADA) [2] simplifies the weight evaluator to a universal one to evaluate negative class weights, meanwhile, the weights are applied to source classifier. Importance Weighted Adversarial Nets (IWAN) [32] appends a positive-class discriminator to assess the positive-sample weight for each sample, and weighted source and target samples are used for adversarial learning. Moreover, there are also some related DA problem setup like [12, 23] focus on universial and openset DA problems.

The above PDA works focus on reducing negative transfer caused by negative classes. IWAN tries to select out positive label space on the sample level, while PADA and SAN try to find out positive label space on the class level. These methods only minimize the shift of positive classes between domains, without considering maximizing the distribution divergence between the negative and target samples. Moreover, they employ possibility based soft-weights to evaluate the transferability. As such, even if the weights of positive classes are higher than the negative, few of them can achieve desired values. This phenomenon can still cause negative transfer.

This paper proposes a Discriminative Partial Domain Adversarial Network (DPDAN), which not only narrows the distribution divergence between source-positive samples and target ones, but also widens the distribution divergence between source-negative samples and the others including source-positive and target ones. Meanwhile, we apply the hard positive-sample label to eliminate negative transfer caused by source-negative classes.

3 Discriminative Partial Domain Adversarial Network

Similar to standard domain adaptation, in partial domain adaptation we are also provided with a *source* domain $D_s = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^{n_s}$ of n_s labeled examples associated with $|C_s|$ classes and a *target* domain $\mathcal{D}_t = \{\mathbf{x}_i^t\}_{i=1}^{n_t}$ of n_t unlabeled examples associated with $|\mathcal{C}_t|$ classes. Different from DA, in PDA we have $|C_s| > |\mathcal{C}_t|$. We further separate label space C_s into source-positive C_{sp} and source-negative C_{sn}, here $C_{sp} = \mathcal{C}_t$. $\mathcal{D}_{sp}$ is source-positive domain with $|C_{sp}|$ classes and D_{sn} is source-negative domain with $|C_{sn}|$ classes. D_s and D_t are sampled from distributions $p_s(x)$ and $p_t(x)$ respectively. In DA we have $p_s(x) \neq p_t(x)$. In PDA we have $p_{sp}(x) \neq p_t(x)$, where p_{sp} denotes the distribution of D_{sp}. Likewise, p_{sn} denotes the distribution of D_{sn}.

In summary, we should tackle two challenges to enable PDA: **(1)** Mitigate negative transfer by filtering out unrelated source labeled data belonging to source-negative domain $\mathcal{D}_{sn}$. **(2)** Promote positive transfer by maximally matching data distributions $p_{sp}(x)$ and $p_t(x)$ in source-positive domain $\mathcal{D}_{sp}$ and target domain $\mathcal{D}_t$. These two interleaving challenges should be dealt with jointly through decreasing negative influence of $\mathcal{D}_{sn}$ and meanwhile enabling effective domain adaptation between $\mathcal{D}_{sp}$ and $\mathcal{D}_t$. Although source-negative samples cannot be accurately separated from source domain, the distribution p_{sn} of $\mathcal{D}_{sn}$ will be roughly learned from semantic distribution measurement between source-positive and source-negative samples. This is a more practical method than finding out the negative data directly.

The core idea of DPDAN is to decompose source domain distribution p_s into two parts from the perspective of probability distribution: source-positive domain distribution p_{sp} and source-negative domain distribution p_{sn}. The well-separated p_{sp} and p_{sn} can further facilitate the effective transfer by minimizing divergence between p_{sp} and p_t and maximizing differences between p_{sn} and others including p_{sp} and p_t. This can avoid negative transfer to a greater extent.

As shown in Fig. 2, G is the feature extractor, and C is not only the source classifier but also the weight discriminator to obtain the source class weights γ to evaluate the possibility of the source class belonging to positive classes. Meanwhile, inspired by OTSU-methods [18], we transfer γ to w_p and w_n, which represents the possibility of source samples belonging to C_{sp} and C_{sn} respectively. In this way, we can promote the weights of source-positive classes to 1 and that of source-negative ones to 0. D_p tries to narrow the distance between p_{sp} and p_t under aid of w_p, and D_n aims to widen the distance between p_{sn} and others by means of w_p and w_n.

3.1 Discriminative Partial Domain Adversarial Framework

Domain adaptation network is proposed to match the feature distributions cross domains. The basic framework of domain adaptation is domain adversarial neural network (DaNN) [5]. DaNN plays a min-max game. The first player is a feature extractor G that tries to extract common feature from both domains, the second player refers to a domain discriminator D distinguishing which domain the feature comes from. The framework follows the objective given by:

$$\min_G \max_D \mathcal{L}(D,G) = \mathbb{E}_{x\sim p_s}\left[\log\left(D\left(G\left(x\right)\right)\right)\right] + \mathbb{E}_{x\sim p_t}\left[\log\left(1-D\left(G\left(x\right)\right)\right)\right] \quad (1)$$

However, if we apply the DaNN framework to PDA directly, the mismatch between source and target label space can cause performance degeneration. In the previous methods, weights are added to domain discriminators to extract common features from source positive and target domains. Nevertheless, these methods mainly focus on narrowing the distance between p_{sp} and p_t, but ignore widen the distance between p_{sn} and p_t.

We also propose to introduce w_p and w_n to deal with PDA issue, w_n is the negative sample binary weight and w_p is the positive sample binary weight. How to get w_p and w_n will be illustrated in the Section 3.2. As such, the DaNN framework can be written:

$$\begin{aligned}
\min_G \max_{D_p} \mathcal{L}(D_p,G) =& \mathbb{E}_{x\sim p_s}\left[w_p \log\left(D_p\left(G(x)\right)\right)\right] + \mathbb{E}_{x\sim p_t}\left[\log\left(1-D_p\left(G(x)\right)\right)\right] \\
\min_G \max_{D_n} \mathcal{L}(D_p,G) =& \mathbb{E}_{x\sim p_s}\left[w_n \log\left(D_p\left(G(x)\right)\right)\right] + \mathbb{E}_{x\sim p_t}\left[w_n \log\left(1-D_n\left(G(x)\right)\right)\right] \\
&+ \mathbb{E}_{x\sim p_t}\left[\log\left(1-D_n\left(G\left(x\right)\right)\right)\right] \quad (2)
\end{aligned}$$

Here, we apply the positive sample binary weight w_p and negative sample binary weight w_n to positive domain discriminator D_p and negative domain discriminator D_n. In this way, p_s can be split into p_{sp} and p_{sn}. Our framework tries to narrow the distance between source positive and target samples, and further widens the distance between source negative and above ones. Our framework includes two domain discriminators sharing the same feature extracting layers.

3.2 Hard Binary Weights

Only when the weights of the positive classes are 1 and the weights of negative ones are 0, the negative transfer caused by negative label space in source domain can be eliminated. Nevertheless, to avoid misclassification of source positive and negative classes, previous approaches utilize possibility weight to evaluate the importance of classes in source domain. It can cause weights of positive samples are far from 1 while the weights of negative ones are far from 0, which can still cause negative transfer.

However, for each sample in source domain, the softmax output of the source classifier gives a probability distribution over source label space C_s. This distribution describes the probability of source samples belonging to each of the $|C_s|$ classes. A basic assumption is that p_{sp} and p_t are much similar than p_{sn} and p_t [2]. Hence, for target samples, if they are assigned to source negative classes, the possibility will be very small.In another word, the weights of positive classes are much higher than the weights of negative ones.

Similar to [2], we define γ as mean of predict labels over target data.

$$\gamma = \frac{1}{n_t} \sum_{i=0}^{|n_t|} softmax\left(G(x_i^t)\right) \tag{3}$$

We further normalize γ by dividing its largest element. γ is a $|C_s|$-dimension vector. The jth element γ_j indicates the contribution of the jth class. For example, In Office-31 dataset, γ is a 31-dimension vector, the 3rd element in this vector represents the possibility of the third source class belonging to C_{sp}. If the j-th class comes from C_{sn}, γ_j should be close to 0, otherwise close to 1. Hence, we build hard binary weights utilizing threshold to distinguish C_{sp} and C_{sn}.

We define w_p as hard positive binary weight. Inspired by [18], we obtain w_p from γ by automatically selecting an adequate threshold t. The probabilities of class occurrence α_{sn},α_{sp} and the class mean weights β_{sn}, β_{sp} are defined by:

$$\alpha_{sn} = \frac{|C_{sn}|}{|C_s|}, \quad \alpha_{sp} = \frac{|C_{sp}|}{|C_s|}, \quad \beta_{sn} = \frac{\sum_{0<=\gamma_j<t} \gamma_j}{|C_{sn}|}, \quad \beta_{sp} = \frac{\sum_{t<=\gamma_j<=1} \gamma_j}{|C_{sp}|} \tag{4}$$

Whatever choice of t is, it can be confirmed that:

$$\beta_{total} = \beta_{sn} * \alpha_{sn} + \beta_{sp} * \alpha_{sp} \tag{5}$$

We utilize between-cluster variance σ^2 to measure discrepancy of C_{sn} and C_{sp}.

$$\sigma^2 = (\beta_{sn} - \beta_{total})^2 * \alpha_{sn} + (\beta_{sp} - \beta_{total})^2 * \alpha_{sp} \tag{6}$$

The greater σ^2 is, the greater the difference between C_{sn} and C_{sp} is. If some C_{sn} or C_{sp} are misclassified, the difference will be reduced. Therefore, the threshold is set t by the largest variance between-cluster σ^2 refers to the least probability of misclassification, thus:

$$t = \arg\max(\sigma^2) \tag{7}$$

The hard positive binary weight w_p is given by $w_p = 1$ if $gamma_j \geq t$, and 0 otherwise. where w_p represents whether the jth class is from C_{sp} or C_{sn}. If the jth class belongs to C_{sp}, $w_p = 1$. If the jth class belongs to C_{sn}, $w_p = 0$. Accordingly, due to negative binary weight $w_n = 1 - w_p$, if the jth class belongs to C_{sn}, $w_n = 1$. If the jth class belongs to C_{sp}, $w_n = 0$.

3.3 Positive Partial Domain Adaptation

To extract common features from source positive and target domains, we utilize the hard binary weight w_p to select out feature from source positive and target domains. The objective is named as GAN_p, which takes the form of the standard GAN with the value function as follows:

$$GAN_p(G, D_p) = \mathbb{E}_{\mathbf{x}\sim p_s(\mathbf{x})}\left[w_p \log\left(D_p(\mathbf{x})\right)\right] + \mathbb{E}_{\mathbf{x}\sim p_t(\mathbf{x})}\left[\log\left(1 - D_p\left(G(\mathbf{x})\right)\right)\right] \quad (8)$$

In Eq.8, the positive domain adaptation label is as same as w_p. 0 represents input sample belonging to D_s while 1 represents input samples belonging to $\mathcal{D}_t$.

3.4 Negative Partial Domain Adaptation

To further reduce negative transfer, we also focus on how to widen the distance between source negative and other samples. We define this as GAN_n. We notice that if the sample belongs to D_{sp} or D_t, the corresponding w_n is 0, otherwise, w_n is 1. Hence, we set w_n as domain label for negative partial domain adaptation. If the sample is a part of D_n, the corresponding negative domain label is 1, otherwise 0. Meanwhile, in this part, we focus on the negative samples, the weight of negative samples should be w_n, while the weight of positive samples should be w_p, $w_n = 1 - w_p$. The value function of GAN_n is given by:

$$\begin{aligned} GAN_n(G, D_n) =& \mathbb{E}_{\mathbf{x}\sim p_s(\mathbf{x})}\left[w_n \log\left(D_n(G(\mathbf{x}))\right)\right] + \mathbb{E}_{\mathbf{x}\sim p_s(\mathbf{x})}\left[w_p \log\left(1 - D_n(G(\mathbf{x}))\right)\right] \\ &+ \mathbb{E}_{\mathbf{x}\sim p_t(\mathbf{x})}\left[\log\left(1 - D_n\left(G(\mathbf{x})\right)\right)\right] \end{aligned} \quad (9)$$

In Eq. 9, w_n also represents whether the sample belonging to D_{sn}, if the samples belong to D_{sp} or D_t, the negative domain label is 0, otherwise, it is 1.

In contrast to the 'zero-sum' loss, the optimization of GAN_n is given by two steps. First, the optimal D_n^* is obtained by maximizing Eq. 10.

$$\begin{aligned} D_n^* =& \arg\max_{D_n} \mathbb{E}_{\mathbf{x}\sim p_s(\mathbf{x})}\left[w_n \log\left(D_n(G(\mathbf{x}))\right)\right] + \mathbb{E}_{\mathbf{x}\sim p_s(\mathbf{x})}\left[w_p \log\left(1 - D_n(G(\mathbf{x}))\right)\right] \\ &+ \mathbb{E}_{\mathbf{x}\sim p_t(\mathbf{x})}\left[\log\left(1 - D_n\left(G(\mathbf{x})\right)\right)\right] \end{aligned} \quad (10)$$

Then for widen the distance between p_{sn} and others, G^* is optimized by plugging D_n^* into Eq. 9 and minimizing $-GAN_n(G, D_n^*)$:

$$G^* = \arg\min_G -GAN_n(G, D_n^*) \quad (11)$$

Equations 8-11 suggest when facing both D_p and D_n, G struggles to make the induced p_{sn} stay away from p_t, and forces p_{sp} to close with p_t. Minimizing Eq.11 helps D_n to separate source-positive and target samples from source-negative samples rather than confusing D_n. This crucial effect will eventually push p_t away from p_{sn} but towards p_{sp}. So the well-separated p_{sp} and p_{sn} can further facilitate the effective transfer of knowledge by minimizing divergence of p_{sp} and p_t and maximizing differences between p_{sn} and the combination of p_t and p_{sp}. This will avoid negative transfer to a greater extent.

3.5 Discriminative Partial Domain Adversarial Network

In DPDAN, positive discriminator D_p narrows the distance between p_{sp} and p_t, and negative discriminator D_n widens the distance between p_{sn} and the combination of p_{sp} and p_t. Thus, we get two well-separated distributions p_{sp} and p_{sn} for discriminators D_p and D_n. The overall objective can be written as:

$$\min_{G,D_n}\max_{D_p}\mathcal{L}(G,D_p,D_n)=\mathbb{E}_{\mathbf{x}\sim p_s(\mathbf{x})}\left[\gamma C\left(G(\mathbf{x}),y\right)\right]+\lambda_p GAN_p(G,D_p)-\lambda_n GAN_n(G,D_n) \tag{12}$$

where λ_p and λ_n control the trade-off between D_p and D_n respectively. C is the source classifier.γ is soft weight. At the beginning of the training, w_p is set as 1 for all source classes. After every 500 iterations, w_p will be updated based on the transferability between source and target classes. D_p, D_n and G plays the min-max game, and only D_p inserts a gradient reversal layer (GRL) [5] to multiply the gradient by -1 for the feature extractor to learn G and D_p simultaneously. All these modules are trained together.

In fact, since the weight applied to the source domain classifier is soft weight γ, even if some source classes are misclassified, features from misjudged samples are still being learned by C. D_p constantly narrows the gap between the source and target domain, these misclassified samples at early stage will be easy to distinguish in the process of training, correcting misclassified source samples.

3.6 Theoretical Analysis

Theorem 1. *At the Nash equilibrium point of Eq.12, the minimal JSD between source-positive distributions p_{sp} and the target data distributions p_t is achieved, i.e. $p_{sp}=p_t$. Meanwhile, the JSD between source-negative distribution p_{sn} and others is maximized as much as possible.*

Given fixed generators G, the optimal discriminators D_p and D_n for the objective in Eq.12 have the following forms:

$$D_p^*=\frac{w_p p_s(x)}{w_p p_s(x)+p_t(x)},\quad D_n^*=\frac{w_n p_s(x)}{p_s(x)+p_t(x)} \tag{13}$$

Proof. In GAN_p, we try to minimize the JSD between $p_s(x)w_p$ and $p_t(x)$. Similar to GAN network, given x, one obtains the optimal D_p^* is by maximizing:

$$f(D_p^*)=p_s(x)w_p\log D_p(x)+p_t(x)\log(1-D_p(x)) \tag{14}$$

The derivative of $f(D_p)$ is $\frac{df(D_p)}{dD_p}=\frac{p_s(x)w_p}{D_p(x)}-\frac{p_t(x)}{1-D_p(x)}$. Hence, $D_p^*=\frac{p_s(x)w_p}{p_s(x)w_p+p_t(x)}$. When it comes to $GAN_n(G,D_n)$, we try to maximize the JSD between $p_s(x)w_n$ and $p_s(x)w_p+p_t(x)$. Similar to GAN_p, we conclude that the maximum D_p and D_n can be achieved at Eq.13. Substitute optimal D_p^* and D_n^* into Eq.12, then

$\mathcal{L}(G, D_p, D_n)$ becomes:

$$\begin{aligned}
&\lambda_p \left\{ \mathbb{E}_{\mathbf{x}\sim p_s(\mathbf{x})} \left[w_p \log \left(\frac{w_p p_s(x)}{w_p p_s(x) + p_t(x)} \right) \right] + \mathbb{E}_{\mathbf{x}\sim p_t(\mathbf{x})} \left[\log \left(1 - \frac{w_p p_s(x)}{w_p p_s(x) + p_t(x)} \right) \right] \right\} \\
&- \lambda_n \left\{ \mathbb{E}_{\mathbf{x}\sim p_{sp}(\mathbf{x})} \left[w_p \log \left(\frac{w_n p_s(x)}{p_s(x) + p_t(x)} \right) \right] + \mathbb{E}_{\mathbf{x}\sim p_{sn}(\mathbf{x})} \left[w_n \log \left(\frac{w_n p_s(x)}{p_s(x) + p_t(x)} \right) \right] \right. \\
&\qquad \left. + \mathbb{E}_{\mathbf{x}\sim p_t(\mathbf{x})} \left[\log \left(1 - \frac{w_n p_s(x)}{p_s(x) + p_t(x)} \right) \right] \right\} \\
=&\lambda_p \left(2JSD\left(w_p p_s | p_t \right) - \log 4 \right) - \lambda_n \left(2JSD\left(w_n p_s | p_t + w_p p_s \right) - \log 4 \right) \qquad (15)
\end{aligned}$$

which peaks its minimum if $p_s w_p = p_t$ and $p_s w_n \neq p_s w_p + p_t$. The detail of the proof is provided in our supplementary material.

The proof reveals that approaching to Nash equilibrium is equivalent to jointly minimizing $JSD\left(w_p p_s | p_t\right)$ and maximizing $JSD\left(w_n p_s | p_t + w_p p_s\right)$. Thus, DPDAN tries to captures p_{sn} and p_{sp}. So, the proposed method can simultaneously achieve probability distribution alignment and prevent negative transfer.

4 Experiment

4.1 Datasets and Protocols

Office-31 [22] dataset is a standard DA dataset. It contains 31 categories decomposed by three different domains: Amazon (A), Webcam (W) and DSLR (D). We denote three domains with 31 categories as source domain A31, W31 and D31. 10 categories [6] shared by Office-31 and Caltech-256 [8] dataset are defined as target domain A10, W10 and D10 respectively. We evaluate the methods in 6 partial domain adaptation tasks.
Caltech-Office utilizes Caltech-256 dataset as source domain, while the 10 positive classes shared by Office-31 and Caltech-256 dataset as target domain. We denote source domain as C256 and target domain as A10, W10, D10, respectively. Moreover, these 10 classes are also used as source domain while the target domain is the first five classes in 10 classes as target domain C5, A5, W5 and D5. We evaluate our methods in 15 partial domain adaptation tasks.
Office-Home dataset [27] is a larger dataset with a higher domain distribution discrepancy. It includes four different domains with 65 categories: Artistic, Clip Art, Product images and Real-World. They are denoted as Ar, Cl, Pr and Rw. The target domain has 25 classes set as [2]. We carry out 6 partial domain adaptation tasks in this dataset.
VisDA2017 [29] is a challenging large-scale , which tries to narrow the synthetic-to-real domain gap across 12 categories. Under partial setting, we choose the first 6 categories(in alphabetic order) as target domain and conduct Symthetic12 $\rightarrow$ Real6 task as S$\rightarrow$R.

We compare our DPDAN with present state-of-the-art results domain adaptation [4, 9, 14] and partial domain adaptation methods [1, 2, 32].

Table 1. Accuracy of partial DA tasks on *Caltech-Office* (10 classes → 5 classes).

Method	Caltech-Office (10 classes → 5 classes)												
	C → A	C → W	C → D	A → C	A → W	A → D	W → C	W → A	W → D	D → C	D → A	D → W	Avg
AlexNet [11]	93.58	83.70	91.18	85.27	76.30	85.29	74.17	87.37	**100.00**	80.82	89.51	98.52	87.14
DaNN [5]	91.86	82.22	83.82	77.57	65.93	80.88	72.60	80.30	95.59	69.35	77.09	80.74	79.83
RTN [4]	91.86	93.33	80.88	80.99	69.63	70.59	59.08	74.73	**100.00**	59.08	70.02	91.11	78.44
ADDA [25]	93.15	94.07	97.06	85.27	87.41	89.71	86.82	92.08	**100.00**	89.90	93.79	98.52	92.31
IWAN [32]	94.22	**97.78**	98.53	89.90	87.41	88.24	90.24	95.29	**100.00**	91.61	94.43	98.52	93.85
PADA [2]	96.25	96.00	97.59	92.05	87.33	96.39	96.85	96.14	**100.00**	95.80	97.31	97.87	95.72
DPDAN	**96.28**	96.67	**100.00**	**97.15**	**91.33**	**100.00**	**97.11**	**97.93**	**100.00**	**96.59**	**97.32**	**100.00**	**97.53**

Table 2. Accuracy of *Office-Home* and *Caltech-Office* (256 classes → 10 classes).

Method	Office-Home							Method	Caltech-Office(256 classes → 10 classes)			
	Ar → Rw	Ar → Cl	Pr → Rw	Rw → Ar	Rw → Cl	Rw → Pr	Avg		C → W	C → A	C → D	Avg
ResNet [9]	75.87	46.33	74.88	67.40	48.18	74.17	64.47	AlexNet [11]	58.44	74.64	65.86	66.98
DaNN [5]	77.47	43.76	76.37	69.15	44.30	77.48	64.75	ResNet [9]	61.33	77.57	68.90	69.27
RTN [4]	78.58	49.31	75.32	63.18	43.57	80.50	65.58	DaNN [5]	54.57	72.86	57.96	61.80
IWAN [32]	78.12	53.94	81.28	76.46	56.75	**82.90**	71.58	RTN [16]	71.02	81.32	62.35	71.56
SAN [1]	74.60	44.42	80.07	72.18	50.21	78.66	66.69	SAN [14]	88.33	83.87	85.54	85.83
PADA [2]	78.74	51.95	78.79	73.73	56.6	77.09	69.48	PADA [1]	89.07	89.34	88.54	88.93
DPDAN	**79.04**	**59.40**	**81.79**	**76.77**	**58.67**	82.18	**72.98**	DPDAN	**89.96**	**90.17**	**92.06**	**90.73**

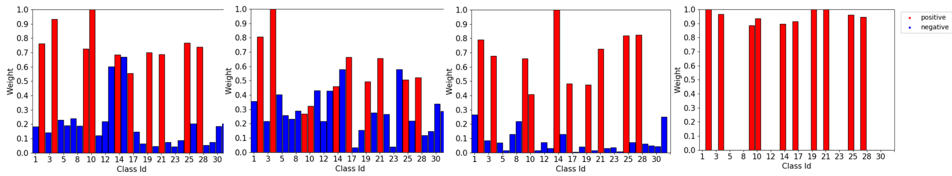
Table 3. Accuracy of tasks on *Office-31* and *VisDA2017*.

Method	Office-31							VisDA2017
	A → W	D → W	W → D	A → D	D → A	W → A	Avg	S → R
ResNet [9]	54.52	94.57	94.27	65.61	73.17	71.71	75.64	45.26
DAN [14]	46.44	53.56	58.60	42.68	65.66	65.34	55.38	47.60
DaNN [5]	41.35	46.78	38.85	41.36	41.34	44.68	42.39	51.01
ADDA [25]	43.65	46.68	40.12	43.66	42.76	45.95	43.77	50.06
RTN [4]	75.25	97.12	98.32	66.88	85.59	85.70	84.81	50.04
IWAN [32]	76.27	98.98	**100.00**	78.98	89.46	81.73	87.57	52.18
SAN [1]	81.82	98.64	**100.00**	81.28	80.58	83.09	87.27	52.06
PADA [2]	86.54	99.32	**100.00**	82.27	92.69	95.41	92.69	53.53
DPDAN	**96.27**	**100.00**	**100.00**	**96.82**	**96.35**	**95.62**	**97.51**	**65.26**

Table 4. Accuracy of DPDAN and its variants on *PDA* and *DA* setting.

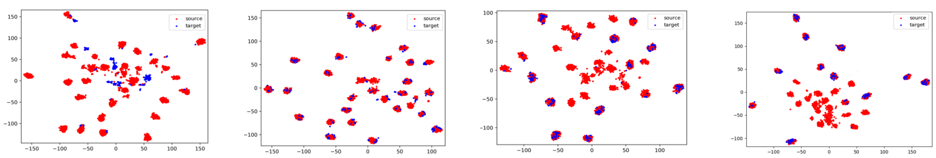
DPDAN	Office-31							VisDA2017
	A → W	D → W	W → D	A → D	D → A	W → A	Avg	S → R
w/o hard binary weight	86.53	**100.00**	**100.00**	80.06	92.03	95.41	92.34	53.66
w/o negative domain discriminator	91.19	**100.00**	**100.00**	95.12	95.02	95.51	96.14	60.12
vanilla	**96.27**	**100.00**	**100.00**	**96.82**	**96.35**	**95.62**	**97.51**	**65.26**
DPDAN	**Office-10 → Caltech-10**							VisDA2017
	A → W	D → W	W → D	A → D	D → A	W → A	Avg	S → R
RTN [4]	95.51	95.20	94.19	93.70	**99.22**	**100.00**	96.31	63.80
vanilla	**96.01**	**96.33**	**95.45**	**94.38**	98.35	99.06	**96.60**	**64.51**

We conduct ablation tests by evaluating two variants of DPDAN: **(1)** DPDAN w/o hard binary weight is the variant without hard binary weight, in which only the positive and negative discriminators play their roles. In this case, the hard binary weight is taken placed by soft-weight. **(2)** DPDAN w/o negative discriminator is the variant without negative classes discriminator, in which only the positive discriminator and hard binary weight play their roles.**(3)** DPDAN from **Office-10** classes to **Caltech-10** classes and **VisDA2017** in standard domain adaptation setting. In this setting, we compared our results to RTN[[4]. We



(a) ResNet (b) DaNN (c) PADA (d) DPDAN

Fig. 3. Histograms of class weights learned by ResNet, DaNN, PADA and DPDAN.



(a) ResNet (b) DaNN (c) PADA (d) DPDAN

Fig. 4. Visualization of features learned by ResNet, DaNN, PADA and DPDAN.

implement all methods in PyTorch, and finetune ResNet-50 [9] and AlexNet [11] pre-trained on ImageNet. Our implementation is based on DaNN [5]. The classifier layers C is added before DaNN bottleneck. For DPDAN, we train C, D_p and D_n from scratch. The learning rate of these above layers are set as 10 times of other layers. We use mini-batch stochastic gradient descent (SGD) with momentum of 0.9 and the learning rate strategy implemented in DaNN. The learning rate is adjusted during SGD using $p = \frac{\eta_0}{(1+\alpha p)^\gamma}$, where p is the training progress changing from 0 to 1, while α and γ are optimized with importance-weighted cross-validation [24] on one task of a dataset and fixed for all the other tasks of this dataset. Moreover, at the beginning of training, w_p is set as 1 for each class belonging to source domain in case the influence of prior knowledge. Note C, w_p and w_n are updated each 500 iterations.

4.2 Experimental Results

Classification results using ResNet-50 on the twelve tasks of Office10-Caltech5 are shown in Table 1. Six tasks of Office-Home and three tasks of Caltech256-Office10 are shown in Table 2, and six tasks of Office-31 are shown in Table 3.

The results also imply some insightful observations. **(1)** ResNet overperforms other standard DA on most tasks. It shows that source-negative classes have negative impact on standard DA methods. **(2)** RTN utilizes the entropy minimization criterion to restrain negative transfer caused by source-negative classes, but the result is still not satisfied. **(3)** PDA approaches perform better on these tasks due to the negative class evaluation. **(4)** DPDAN outperforms all the others, proving that our proposed mechanism can decrease negative transfer by dividing source label space into positive and negative spaces. As such, DPDAN is more accurate than the previous standard and partial DA approaches.

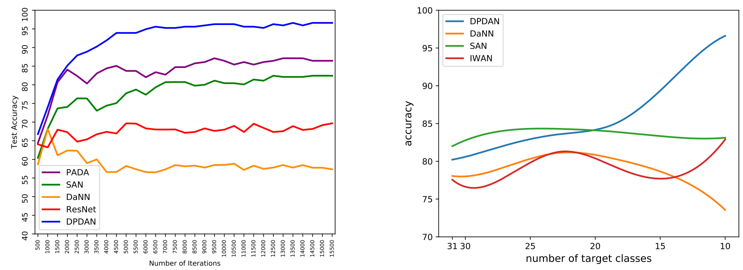
We perform some ablation experiments to inspect the effect of different modules in Table 4. The results indicate some interesting points. **(1)** DPDAN outperforms DPDAN w/o hard binary weight, proving hard positive binary weight plays an important role. **(2)** DPDAN outperforms DPDAN w/o negative domain discriminator, showing negative domain adversarial network can decrease negative transfer by maximizing the distance between source-negative samples and others. **(3)** Our experiment can also get better results on standard DA compared with RTN according to Table 4.**(4)** Different from most ablation experiments, the relationship between D_n and w_p is not independent but sequential. In the experiment of DPDAN w/o hard binary weight from Table 4, due to lack of hard binary weight, we can only use soft-weight as negative domain labels directly. Actually this ablation experiment is only a combination of PADA and D_n. Here, the gap between the negative domain labels of positive and negative classes will be very small, and D_n will be hard to work, so the promotion is tiny. However, comparing DPDAN with DPDAN w/o negative domain discriminator, after setting w_p as negative domain label, the gap between the negative domain labels of positive and negative classes can be more distinct. Then D_n can effectively zoom out the distance between source negative classes and others. The comparison between these two shows the obvious performance promotion of D_n.

4.3 Analysis and Discussion

Class Weight: Fig. 3 are the histograms of class weights learned by ResNet-50, DaNN, PADA and DPDAN on task A (31 classes) $\rightarrow$ W (10 classes). The red and blue bins represent positive and negative samples, respectively.

Fig. 3(a) implies ResNet-50 can select out most positive classes thanks to finetune. Fig. 3(b) shows DaNN can barely classify positive and negative classes resulting in negative transfer. From Fig. 3(c), we observe that PADA can classify positive and negative classes correctly, but most weights of the positive and negative samples cannot achieve 1 and 0, which can still cause negative transfer. From Fig. 3(d), we can see the weights of positive and negative classes are almost to expect values. DPDAN can select out positive and negative classes correctly, and nearly eliminates negative transfer.

Feature Visualization: We visualize the t-SNE embeddings [3] of the bottleneck layer learned by ResNet-50, on task A (31 classes) $\rightarrow$ W (10 classes) in Fig. 4. The red points are source samples while the blue are target ones. From Fig. 4, we have some intuitive observations. **(1)** Thanks to finetune, ResNet can cluster some target samples into the right classes, but the accuracy is still not satisfied. **(2)** DaNN can not select out negative classes and lead to negative transfer. **(3)** PADA can select out most negative classes but the boundary between positive and negative samples is still unclear. **(4)** DPDAN can select out positive and negative classes, and cluster negative samples together. Each cluster of positive classes is far from the others. In this way, positive samples are hard to be misclassified and the negative samples are easy to be selected out.



(a) Accuracy over iterations (b) Accuracy over target classes

Fig. 5. Target test accuracy.

Convergence Performance: In Fig. 5(a) we compare our results of target test accuracy with other methods on task A (31 classes) $\rightarrow$ W (10 classes). DPDAN can reach the highest accuracy rapidly, and the accuracy is still the most robust when the iteration increases. This observation also shows that our DPDAN can be trained more efficiently than previous PDA and DA methods.

Target Classes: In Fig. 5(b) we conduct plenty of experiments with a wide range of number of target classes on task A (31 classes) $\rightarrow$ W (10 classes). DaNN performance degenerates when the number of target classes decreases. The performance of SAN is stable and accuracy does not deteriorate with the number of target classes decreasing. IWAN performs better than DANN only when the label space does not overlap much and negative transfer is very serious. DPDAN outperforms most other methods when the label space totally overlaps. It performs better when the number of target number becomes less, which shows our approach can effectively select out negative classes and promote accuracy.

5 Conclusion

We propose a Discriminative Partial Domain Adversarial Network for partial domain adaptation. A hard binary weighting algorithm is proposed to decide which class belongs to positive ones, the weights of positive classes can be as high as possible while the weights of the negative are almost zero, which can eliminate negative transfer greatly. Our framework contains two domain discriminator and one feature extractor to identify positive and negative samples from source domain. The distribution divergence between source-negative samples and all the others is maximized to mitigate negative transfer, and simultaneously the domain shift between source-positive and target samples is narrowed to obtain more discriminative domain-invariant features. The proposed framework can not only be applied to partial domain adaptation, but also be utilized to standard domain adaptation.Our DPDAN outperforms PDA and DA methods, and achieves the state-of-the-art result, showing the effectiveness and robustness of the method.

References

1. Cao, Z., Long, M., Wang, J., Jordan, M.I.: Partial transfer learning with selective adversarial networks. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
2. Cao, Z., Ma, L., Long, M., Wang, J.: Partial adversarial domain adaptation. In The European Conference on Computer Vision (ECCV) (2018)
3. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell., T.: Decaf: A deep convolutional activation feature for generic visual recognition. In International Conference on Machine Learning (ICML), (2014)
4. G. Cai, Y. Wang, M.Z.L.H.: Unsupervised domain adaptation with residual transfer networks. In Advances in Neural Information Processing Systems (2016)
5. Ganin, Y., Ustinova, E., Ajakan, H.., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., Lempitsky, V.S.: Domain-adversarial training of neural networks. Journal of Machine Learning Research **17**, 59:1–59:35 (2016)
6. Gong, B., Shi, Y., Sha, F., Grauman., K.: Geodesic flow kernel for unsupervised domain adaptation. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2012)
7. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets (2014)
8. Griffin, G., Holub, A., Perona, P.: Caltech-256 object category dataset. Technical report (2007)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
10. Hoffman, J., Tzeng, E., Park, T., Zhu, J., P. Isola, K.S., Efros, A.A., Darrell, T.: Cycada: Cycleconsistent adversarial domain adaptation. In Proceedings of the 35th International Conference on Machine Learning (ICML) **2**, 1994–2003 (2018)
11. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems (NIPS) (2012)
12. L. P. Jain, W.J.S., Boult., T.E.: Multi-class open set recognition using probability of inclusion. In European Conference on Computer Vision (ECCV) (2014)
13. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation (2015), in IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
14. Long, M., Cao, Y., Wang, J., Jordan., M.I.: Learning transferable features with deep adaptation networks. In Proceedings of the 34th International Conference on Machine Learning, International Conference on Machine Learning (ICML) (2015)
15. Long, M., Cao, Z., Wang, J., I, J.M.: transferable features with deep adaptation networks. In Proceedings of the 34th International Conference on Machine Learning, International Conference on Machine Learning (ICML) (2015)
16. Long, M., Zhu, H., Wang, J., Jordan., M.I.: Deep transfer learning with joint adaptation networks. In Proceedings of the 34th International Conference on Machine Learning, International Conference on Machine Learning (ICML) (2017)
17. Luo, Z., Zou, Y., Hoffman, Y., Li, F.: Label efficient learning of transferable representations acrosss domains and tasks. In Advances in Neural Information Processing Systems (NIPS) (2017)
18. Otsu, N.: A threshold selection method from grey-level histograms. IEEE Transactions on Systems, Man, and Cybernetics **9**, 62–66 (1979)

19. Pan, Y., Yao, T., Li, Y., Wang, Y., Ngo, C., Mei, T.: Transferable prototypical networks for unsupervised domain adaptation. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
20. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn:towards real-time object detection with region proposal networks. In Advances in Neural Information Processing Systems **28**, 91–99 (2015)
21. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei., L.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision (IJCV) **115**, 211–252 (2015)
22. Saenko, K., Kulis, B., Fritz, M., Darrell, T.: Adapting visual category models to new domains. In European Conference on Computer Vision (ECCV) (2010)
23. Saito, K., K.D.S.S., Saenko, K.: Universal domain adaptation through self-supervision. arXiv preprint arXiv
24. Sugiyama, M., Krauledat, M., Muller., K.R.: Covariate shift adaptation by importance weighted cross validation. Journal of Machine Learning Research (JMLR) **8**, 985–1005 (2007)
25. Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
26. Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., Darrell., T.: Simultaneous deep transfer across domains and tasks. In IEEE International Conference on Computer Vision (ICCV) (2015)
27. Venkateswara, H., Eusebio, J., Chakraborty, S., Panchanathan, S.: Deep hashing network for unsupervised domain adaptation. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
28. Wang, C., Tuo, H., Wang, J., Qiao, L.: Discriminative transfer learning via local and global structure preservation. Signal, Image and Video Processing **13**(4), 753–760 (Jun 2019). https://doi.org/10.1007/s11760-018-1405-7, `https://doi.org/10.1007/s11760-018-1405-7`
29. Xingchao Peng, Ben Usman, N.K.J.H.D.W., Saenko, K.: The visual domain adaptation challenge. arXiv preprint arXiv (2017)
30. X.Wang, Li, L., Wei, W., Long, M., Wang, J.: Transferable attention for domain adaptation. the Association for the Advance of Artificial Intelligence (AAAI) (2019)
31. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In Advances in Neural Information Processing Systems (NIPS) (2014)
32. Zhang, J., Ding, Z., Li, W., Ogunbona, P.: Importance weighted adversarial nets for partial domain adaptation. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
33. Zhong, H., Tuo, H., Wang, C., Ren, X., Hu, J., Qiao, L.: Source-constraint adversarial domain adaptation pp. 2486–2490 (09 2019). https://doi.org/10.1109/ICIP.2019.8803282