Small Is Beautiful: Compressing Deep Neural Networks for Partial Domain Adaptation

Yuzhe Ma, *Member, IEEE*, Xufeng Yao, *Graduate Student Member, IEEE*, Ran Chen, Ruiyu Li, *Member, IEEE*, Xiaoyong Shen, *Member, IEEE*, and Bei Yu, *Member, IEEE*

***Abstract*— Domain adaptation is a promising way to ease the costly data labeling process in the era of deep learning (DL). A practical situation is partial domain adaptation (PDA), where the label space of the target domain is a subset of that in the source domain. Although existing methods yield appealing performance in PDA tasks, it is highly presumable that computation overhead exists in deep PDA models since the target is only a subtask of the original problem. In this work, PDA and model compression are seamlessly integrated into a unified training process. The cross-domain distribution divergence is reduced by minimizing a soft-weighted maximum mean discrepancy (SWMMD), which is differentiable and functions as regularization during network training. We use gradient statistics to compress the overparameterized model to identify and prune redundant channels based on the corresponding scaling factors in batch normalization (BN) layers. The experimental results demonstrate that our method can achieve comparable classification performance to state-of-the-art methods on various PDA tasks, with a significant reduction in model size and computation overhead.**

***Index Terms*— Deep learning, neural network compression, transfer learning.**

I. Introduction

SUPERVISED deep learning (DL) has achieved significant success in various applications. A common restriction of conventional DL algorithms is the great demand for labeled data which is costly. Domain adaptation is a promising solution to tackle this problem, which leverages rich labeled data in the source domain to build a model for the target domain where very limited or even no labeled data are available. The core idea is to learn domain-invariant representations such that the cross-domain distribution inconsistency can be resolved. Thanks to the extraordinary capability of representation learning of deep models [1]–[3], recent studies show that the DL models can achieve compelling performance in domain adaptation [4]–[8].

Manuscript received 13 March 2021; revised 24 November 2021 and 27 April 2022; accepted 16 July 2022. Date of publication 8 August 2022; date of current version 1 March 2024. *(Corresponding author: Yuzhe Ma.)*

Yuzhe Ma is with the Microelectronics Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511400, China (e-mail: mayuzhe533@gmail.com).

Xufeng Yao, Ran Chen, and Bei Yu are with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong.

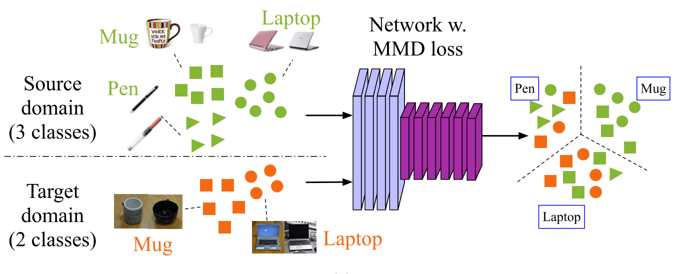
Ruiyu Li and Xiaoyong Shen are with SmartMore Corporation Limited, Hong Kong.

Digital Object Identifier 10.1109/TNNLS.2022.3194533

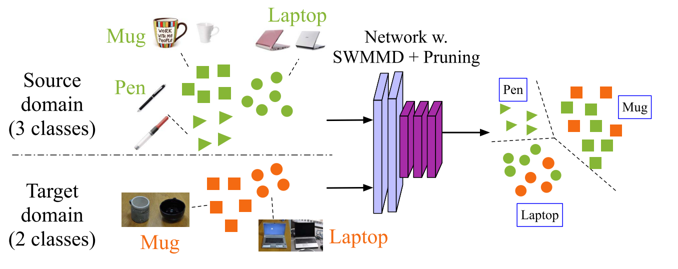
Most of the existing DL-based approaches assume that the source domain and the target domain share identical label space. More specifically, the class prior distributions in the source domain are assumed to be the same as that in the target domain. Thus, a set of approaches are proposed to explicitly match the distributions using criteria such as maximum mean discrepancy (MMD) [5], [9]. However, this assumption may not always hold in practice. Open set domain adaptation [10], [11] and partial domain adaptation (PDA) [12] are explored to deal with a more realistic scenario. PDA studies the situation that the target domain only contains a subset of categories. While open set domain adaptation introduces "unknown" classes in the target domain, which do not exist in the label set of the source domain [10], [13]–[15]. To further alleviate the assumption of the prior knowledge about the label sets, another highly related problem is the universal domain adaption in which there is no prior knowledge imposed on the label sets of both the domains [16], [17].

In this article, we investigate a practical situation of PDA, which is challenging since the class prior distributions are no longer consistent or even far apart between the source and target domains. Typical MMD-based methods may lead to negative transfer and notable performance degradation. The latest advance in PDA is based on adversarial training [8], [12], [18], [19]. The key idea is to assign different weights to different classes, where larger weights are assigned to shared classes and smaller weights are given to outlier classes such that negative transfer is alleviated and positive transfer is promoted.

Despite the appealing performance of DL models on PDA, the execution overhead remains a critical issue for modern deep neural networks in terms of power consumption and storage. Intuitively, the number of parameters in a neural network suggests its representation capability. Therefore, it is worth exploring model compression for PDA since redundancy is more likely to exist in this situation. For example, a large CNN is designed and trained on a large/difficult labeled dataset (e.g., ImageNet-1000), and it needs to be transferred to a small/easy dataset (e.g., Office-31). There is a high chance that the original model is overparameterized for the target task, which motivates us to compress the model. Network pruning or model compression removes the parameters in the network that are useless or even harmful to the target task, leading to a sparsified neural network. Several previous works use a mask to the network, and each value in the mask



(a)


Mug
Laptop
Network w. SWMMD + Pruning
Source domain (3 classes)
Pen
Target domain (2 classes)
Mug
Laptop
Pen
Mug
Laptop

(b)

Fig. 1. (a) Conventional MMD-based approach applied to PDA. Outlier class (pen, represented by triangle) leads to poor performance due to negative transfer. (b) Proposed approach with SWMMD and model pruning. Simultaneously improve classification performance and reduce model size.

serves as a scaling factor of a feature map. Then sparsifying the network is equivalent to sparsifying the mask. The mask can be trained along with the weights in the network with certain sparsity-driven regularizations [20]–[22]. As a result, extra parameters are introduced as masks to the training stage. Although there is rich literature studying model compression, most of them are developed for supervised learning, in which labeled data are needed to guide the pruning or retrain a pruned model to retain performance. Unfortunately, these methods cannot be directly applied to domain adaptation scenarios due to the unavailability of labeled data in the target domain.

In this article, we investigate a new perspective for the PDA problem, which is combined with model compression. The model compression and PDA are seamlessly integrated into a unified training process by iteratively pruning and training a base network. As a result, a slimmed model is obtained, and negative transfer can be circumvented. The resulting model achieves superior performance than general MMD-based approaches, as shown in Fig. 1. Specifically, we design two collaborative schemes for network training and pruning, respectively. On one hand, distribution discrepancy is bridged by minimizing a soft-weighted MMD (SWMMD) to learn domain-invariant features and promote knowledge transfer, which is more effective for the PDA problem than the hard-weighted scheme. The class weights can be directly computed for the source domain because the labels are available. By assigning a soft pseudo label to each sample in the target domain, the class weights of the target domain are estimated. Based on the class weights in both the domains, the shared classes and outlier classes can be distinguished, and an SWMMD is adopted in training. On the other hand, a channel pruning scheme is designed on top of network training. The importance of each channel is evaluated, and those less critical channels are identified and pruned. In contrast to other methods that directly evaluate channels, we leverage the scaling factors in batch normalization (BN) layers based on Taylor expansion, which uses gradient statistics during backpropagation. Thus, pruning can be naturally integrated with model training. Reducing the model size also reduces the chance of overfitting. Hence, our proposed model can achieve appealing performance on the target task and is energy-efficient. In summary, the main contributions of this work are as follows.

1) The PDA problem is investigated from the model compression perspective using a unified training and pruning process.
2) Domain discrepancy issue in the PDA problem is addressed by an SWMMD. It can omit outlier samples in contrast to conventional MMD and is beneficial to training convergence compared with hard-weighted MMD.
3) Model pruning is performed with BN scaling factors based on Taylor expansion, which can be naturally integrated into model training.
4) Experimental results demonstrate that the proposed method can reduce both computation and model size by more than 70% with little performance degradation compared with the state-of-the-art PDA methods.

II. Related Work

A. Unsupervised Domain Adaptation

Domain adaptation is a solution to reduce the need and effort to collect the training data [23]. Since the main issue is the distribution discrepancy across different domains, many methods are proposed to match the feature distributions in the source and target domains [24], [25]. Recently, more efforts on unsupervised domain adaptation with DL methods have been witnessed. The first category is based on explicit distribution matching with a well-defined criterion, e.g., MMD [5], [6], [26] and central moment discrepancy (CMD) [27]. Alternatively, the adversarial training scheme is investigated by leveraging a domain discriminator [4], [8], [28], assuming that a good representation for domain transfer is one that an algorithm cannot distinguish the origin domain. These methods assume that the label space is fully shared between the source and target domains, which may not always hold in PDA. The adversarial training scheme has been studied for tackling the unsupervised domain adaptation problem [12], [18], [19], [29]–[31], which introduce dedicated adversarial learning mechanisms to distinguish instances from different domains to enable an end-to-end learning of transferable representations.

B. Model Pruning

To ease the computation and storage overhead of deep models, various approaches have been used, including quantization [32], low-rank approximation [33], and model pruning [34], [35]. Among these methods, pruning has been the most widely applied, ranging from nonstructured pruning to structured pruning. It has attracted more attention since potential benefits can be attained from structured pruning, such as inference acceleration and hardware-friendliness. A sparse learning algorithm is proposed in [36], which enables learning a structured sparse network by applying group Lasso regularizations during training. Instead of training a sparse model from scratch, another way is to prune a pretrained model carefully. Channel pruning and filter pruning are effective ways for that purpose [20], [35], [37], [38]. Previous works on network compression focus more on the conventional supervised learning tasks in a single domain, and there are few studies showing how network compression can help in unsupervised domain adaptation tasks. A recent work [39] proposed a transfer channel pruning (TCP) approach for domain adaptation models by removing less critical channels iteratively. However, only an identical label space setting is explored in [39]. In contrast, our method considers a more practical scenario, PDA, and can be generalized to domain adaptation with identical label space as well.

III. Proposed Method

In this section, we first elaborate on the proposed SWMMD for minimizing the cross-domain distribution discrepancy. Then we introduce a model pruning method to address the redundancy in PDA models, followed by the overall training process.

A. Unsupervised PDA With Soft-Weighted MMD

Unsupervised domain adaptation is a challenging task because the labels in the target domain are not available. First, we briefly introduce a conventional MMD metric, which is used to represent the distance between distributions and is widely used in previous works for unsupervised domain adaptation [5], [7], [39]. Given the samples from the source domain $\mathcal{D}_s$ and target domain $\mathcal{D}_t$, MMD can be empirically estimated as follows [9]:

$$\text{MMD}^2(\mathcal{D}_s, \mathcal{D}_t) = \left\| \frac{1}{n_s} \sum_{\mathbf{x}_i \in \mathcal{D}_s} \phi(\mathbf{x}_i) - \frac{1}{n_t} \sum_{\mathbf{x}_j \in \mathcal{D}_t} \phi(\mathbf{x}_j) \right\|_{\mathcal{H}}^2 \quad (1)$$

where $\mathcal{H}$ denotes the reproducing kernel Hilbert space (RKHS), and $\phi(\cdot)$ represents the feature mapping from samples to RKHS and is associated with Gaussian kernel. n_s and n_t represent the number of samples in the source and target domains, respectively.

Previous works apply MMD based on the assumption that the label space is fully shared between the source and target domains. However, in PDA where the assumption does not hold, the MMD-based methods cannot be directly applied. Specifically, class prior distributions are significantly different between the source and target domains since a certain number of classes do not even exist in the target domain. To make MMD effective, a sample $\mathbf{x}_i \in \mathcal{D}_s$ needs to be distinguished whether it belongs to shared classes or outlier classes and relies on the samples in shared classes for knowledge transfer. However, it is not easy since which categories are shared is unknown. To handle this, we design a weighting mechanism at the class level to identify the shared classes and the outlier classes, which can be converted into instance level and lead to a weighted MMD criterion to tackle the PDA problem. A related approach is studied to address the class bias [7] by reweighting classes in label space. PDA can be seen as an extreme case of class bias.

Let $\mathcal{Y}_s$ and $\mathcal{Y}_t$ denote the label space of the source domain and the target domain, respectively. Then $\mathcal{Y}_t \subset \mathcal{Y}_s$ is the condition in PDA. Denote the weights of classes as a vector $\mathbf{w} \in \mathbb{R}^{|C_s|}$. Let $w_c^{(s)}$ and $w_c^{(t)}$ denote the weight of class $c \in \mathcal{Y}_s$ in the source domain and the target domain, respectively. Since the labels of the source domain are available, the number of samples for each class c in the source domain can be obtained, denoted by $n_c^{(s)}$, then the weight of class c in the source domain is calculated as $w_c^{(s)} = n_c^{(s)}/n^{(s)}$, where $n^{(s)}$ is the total number of samples in the source domain. Note $w_c^{(t)} = 0$ for $c \in \mathcal{Y}_s \backslash \mathcal{Y}_t$. Let $r_c = w_c^{(t)}/w_c^{(s)}$. Given a set of samples $\{(\mathbf{x}_i^{(s)}, y_i^{(s)})\}$ drawn from the source domain and $\{\mathbf{x}_j^{(t)}\}$ drawn from the target domain, the weighted MMD is empirically estimated as

$$\text{WMMD}^2(\mathcal{D}_s, \mathcal{D}_t) = \left\| \frac{1}{\sum_{\mathbf{x}_i \in \mathcal{D}_s} r_{y_i^{(s)}}} \sum_{\mathbf{x}_i \in \mathcal{D}_s} r_{y_i^{(s)}} \phi(\mathbf{x}_i) - \frac{1}{n_t} \sum_{\mathbf{x}_j \in \mathcal{D}_t} \phi(\mathbf{x}_j) \right\|_{\mathcal{H}}^2. \quad (2)$$

Typically, MMD is implemented as a loss layer in the network and integrated with training. In each training iteration, a hard pseudo label y'_j is assigned to each sample $\mathbf{x}_j \in \mathcal{D}_t$. An estimation of the class prior in the target domain is obtained based on the percentage of each class. However, it did not work well in PDA. The reasons are twofold: 1) assigning hard pseudo labels to the samples $\mathbf{x}_j \in \mathcal{D}_t$ makes the process easily get stuck at the local optima if the hard labels are wrong from the very beginning, and it is very likely to happen since the network is not well-trained and hence not discriminative enough and 2) due to the missing categories in $\mathcal{Y}_t$, several elements in $\mathbf{r}$ could go to 0, which is equivalent to omitting part of the samples $\mathbf{x}_i \in \mathcal{D}_s$ when calculating the weighted MMD based on (2). In this case, the number of effective samples to calculate MMD may be very small, making MMD not an effective estimation for distribution discrepancy if combined with reason 1), thus leading to inferior results. Detailed statistics and results will be presented in Section IV.

To address this issue, we assign soft pseudo labels instead of hard labels to the target samples, based on which an SWMMD is calculated as a regularization for training. In contrast to assigning a specific class $y'_j \in \mathcal{Y}_s$ to a target sample $\mathbf{x}_j \in \mathcal{D}_t$, the soft pseudo label is assigned based on the posterior distribution. Given an input $\mathbf{x}$, a network $f(\mathbf{W}, \cdot)$. The corresponding output is $\mathbf{y} = f(\mathbf{W}, \mathbf{x})$, where $\mathbf{y} \in \mathbb{R}^{|\mathcal{Y}_s|}$. To get the hard label (i.e., one-hot vector), we can apply the argmax function to decide the index. To get the soft pseudo label $\tilde{\mathbf{y}}$, we use the posterior predictive distribution, which is calculated by applying the Softmax function on the output of the network. $\tilde{\mathbf{y}} = \text{Softmax}(\mathbf{y}) = \text{Softmax}(f(\mathbf{W}, \mathbf{x}))$, where $\tilde{\mathbf{y}} \in \mathbb{R}^{|\mathcal{Y}_s|}$ and can be regarded as the posterior predictive distribution of the classes. Then the class weights of the target domain $\mathbf{w}^{(t)} \in \mathbb{R}^{|\mathcal{Y}_s|}$ are estimated by averaging the soft pseudo labels over all the samples $\mathbf{x}_j \in \mathcal{D}_t$. Denote the parameters as $\mathbf{W}$ and the entire forward computation as $f(\mathbf{W}, \cdot)$, we have

$$\mathbf{w}^{(t)} = \frac{1}{n_t}\sum_{j=1}^{n_t} \tilde{\mathbf{y}}_j = \frac{1}{n_t}\sum_{j=1}^{n_t} \text{Softmax}\big(f(\mathbf{W}, \mathbf{x}_j)\big) \tag{3}$$

where $\tilde{\mathbf{y}} \in \mathbb{R}^{|\mathcal{Y}_s|}$ is the posterior predictive distribution, i.e., soft pseudo label, of a sample $\mathbf{x}_j \in \mathcal{D}_t$. Then the soft-label-based class weights $\mathbf{r} \in \mathbb{R}^{|\mathcal{Y}_s|}$ are calculated as mentioned before. Recall that the weighted MMD can be calculated as (2) as long as the class weights are provided. Therefore, SWMMD can be similarly calculated based on (2) using the soft-label-based class weights $\mathbf{r}$.

The loss function of the network contains a supervised classification loss $\mathcal{L}_{\text{cl}}$ on the source domain data $\{(\mathbf{x}_i^{(s)}, y_i^{(s)})\}$ and SWMMD [see (2)] as regularization. The classification loss is formulated as

$$\mathcal{L}_{\text{cl}} = -\frac{1}{n_s}\sum_{\mathbf{x}_i \in \mathcal{D}_s}\sum_{c=1}^{|\mathcal{Y}_s|} y_{i,c}^{(s)} \log\big(\tilde{y}_{i,c}\big) \tag{4}$$

where $\tilde{\mathbf{y}}_i$ and $y_i^{(s)}$ is the prediction posterior distribution and the one-hot encoded label of data $\mathbf{x}_i$, and $\tilde{y}_{i,c}$ is the predicted probability of being class c.

In addition, an entropy minimization principle [40] is included, which is to minimize the entropy over the posterior predictive probability of the samples on the target domain and is formulated as

$$\mathcal{L}_{\text{en}} = \frac{1}{n_t}\sum_{\mathbf{x}_j \in \mathcal{D}_t}\left(-\sum_{c=1}^{|\mathcal{Y}_s|} \tilde{y}_{j,c} \log \tilde{y}_{j,c}\right). \tag{5}$$

Therefore, the training loss is written as

$$\mathcal{L} = \mathcal{L}_{\text{cl}} + \beta \cdot \text{SWMMD}^2(\mathcal{D}_s, \mathcal{D}_t) + \gamma \cdot \mathcal{L}_{\text{en}}. \tag{6}$$

During training, the class weights $\mathbf{r}$ and model parameters $\mathbf{W}$ are alternatively updated. $\mathbf{r}$ is updated at the start of each iteration using the up-to-date parameters $\mathbf{W}$ in the network. Then the loss is calculated [see (6)] and back-propagated to update model parameters $\mathbf{W}$.

B. Model Pruning for Unsupervised PDA

In addition to achieving high performance on the target tasks, the execution overhead should also be taken into consideration since redundancy is highly speculated to exist in the base networks of the PDA model. To ease the overhead and remove redundancy in the network for PDA, a channel pruning method is introduced for a complement.

BN [41] is widely applied in conventional DNNs to facilitate training. In this work, we adopt a pruning scheme by exploiting the statistics in BN layers to directly prune redundant channels without introducing extra training weights. Regarding the PDA task, it is revealed in AdaBN [42] that BN layers contain the traits of the data domain, which suggests that manipulating the BN layers could be effective for DA problems. Moreover, pruning channels in a feature map are equivalent to setting the corresponding scaling factors to 0. Compared with other channel pruning methods such as TCP [39], the implementation of pruning BN scaling factors is much easier. Therefore, in this work, statistics in the BN layers are leveraged for channel pruning without introducing extra parameters.

Essentially, redundant channels are supposed to impact the least to the training loss compared with other important ones, based on which the model pruning can be formulated as an optimization problem. The objective is to minimize the loss change after removing a set of channels. For a network with BN layers, removing a channel $\mathbf{X}_{i,j}$ (the jth channel in the ith layer) is equivalent to setting the corresponding BN scaling factor $\gamma_{i,j}$ as 0. To facilitate the analysis, let $\Gamma = \{\gamma_{i,j}\}$ denote the full set of the BN scaling factors in the network. Given a set of pruning candidates $\Gamma' \subset \Gamma$, let function $h(\Gamma')$ denote the loss change after setting $\gamma \in \Gamma'$ to 0. The objective is to find such a subset of Γ such that the loss change is minimized

$$\begin{aligned} \min_{\Gamma'} \quad & h\big(\Gamma'\big) \\ \text{s.t.} \quad & \Gamma' \subset \Gamma \\ & h\big(\Gamma'\big) = |\mathcal{L}\big(\mathbf{W}, \Gamma' = \mathbf{0}\big) - \mathcal{L}\big(\mathbf{W}, \Gamma'\big)| \\ & \text{card}\big(\Gamma'\big) = P \end{aligned} \tag{7}$$

where $\mathcal{L}(\mathbf{W}, \Gamma' = \mathbf{0})$ and $\mathcal{L}(\mathbf{W}, \Gamma')$ represent the corresponding loss for those channels that are pruned and kept, respectively. P is the number of channels to be removed each time. $\text{card}(\Gamma')$ is the total number of elements in Γ'.

Solving this combinatorial problem exactly requires exhaustively evaluating all the possible combinations of P channels, which is not practical due to intensive computation. Instead, we use a greedy methodology based on the Taylor expansion for selection. A similar approach has been studied to prune individual parameters in the network [43]. We transform this approach to tackle channel pruning based on BN scaling factors. In contrast to evaluating a subset Γ' in the network, we first evaluate each individual scaling factor and rank them based on their impacts. Then a subset Γ' is formed by selecting P items with the most negligible impacts. The Taylor expansion for an infinitely differentiable function $f(x)$ at point $x = a$ is represented as follows:

$$f(x) = \sum_{p=0}^{P} \frac{f^{(p)}(a)}{p!}(x-a)^p + R_p(x). \tag{8}$$

Therefore, the loss function $\mathcal{L}(\mathbf{W}, \gamma_{i,j})$ near $\gamma_{i,j} = \mathbf{0}$ can be approximated as follows:

$$\mathcal{L}\big(\mathbf{W}, \gamma_{i,j} = 0\big) = \mathcal{L}\big(\mathbf{W}, \gamma_{i,j}\big) - \frac{\delta\mathcal{L}}{\delta\gamma_{i,j}}\gamma_{i,j} + R_1\big(\gamma_{i,j} = 0\big). \tag{9}$$

Here, $R_1(\gamma_{i,j} = 0)$ is the Lagrange form remainder which is neglected due to the heavy computation required and marginal impacts on the results [43]. Then $h(\gamma_{i,j})$ can be approximated with

$$h\big(\gamma_{i,j}\big) = \left|\frac{\delta\mathcal{L}}{\delta\gamma_{i,j}}\gamma_{i,j}\right|. \tag{10}$$

The first term can be derived in backward computation, and the second term is the current value of the scaling factor, and thus (10) can be computed efficiently.

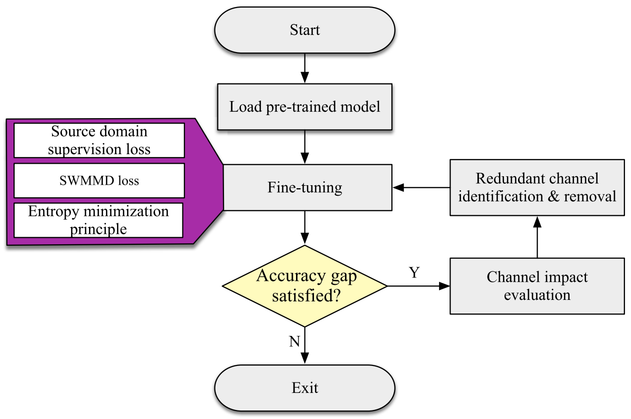

Start
Load pre-trained model
Source domain supervision loss
SWMMD loss
Entropy minimization principle
Fine-tuning
Redundant channel identification & removal
Accuracy gap satisfied?
Y
N
Channel impact evaluation
Exit

Fig. 2. Overall training process.

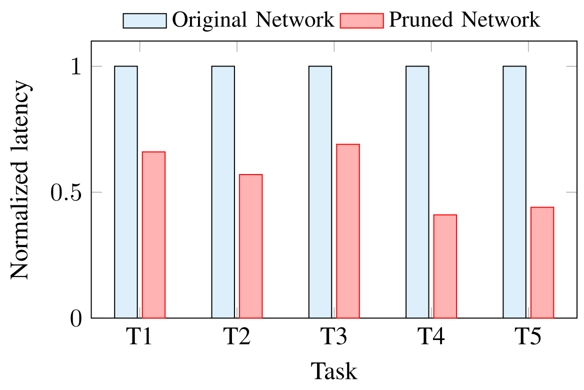


Fig. 3. Inference latency comparison between the original networks and pruned networks. T1: Resnet-50 on the Office31 dataset; T3: Resnet-50 on the Office-Home dataset; T3: Resnet-50 on the ImageCLEF dataset; T4: VGG-16 on the Office31 dataset; T5: VGG-16 on the ImageCLEF dataset.

C. Overall Training Steps

With the introduced training loss and pruning criterion, the training and compression of the PDA model can be seamlessly integrated into a single-stage process. The overall training process is illustrated in Fig. 2. The pretrained network is first loaded as initialization. Then fine-tuning and pruning iteratively were performed. In the fine-tuning stage, (6) is used as the loss function. Specifically, pruning is performed for every T epochs of model training, where T is user-defined. After fine-tuning, the accuracy and the model size are evaluated. Since the target domain does not associate with any labels, we can only acquire some labels on a small portion of images in the target domain to do evaluation. Since the accuracy may degrade as more channels are pruned, we monitor the accuracy gap on the small portion of images between the current model and the preceding one, which reflects whether the pruning still leads to a robust model. The satisfaction threshold is set as 5%. If the gap is satisfactory, then the pruning step begins, including channel impact evaluation and redundancy removal. Otherwise, the training will exit and we use the model obtained from the former iteration as the final result to conduct the final testing and get the performance.

IV. Experimental Results

A. Setup

Four public datasets are adopted in our experiments, including Office-31 [46], Office-Home [47], ImageCLEF-DA [48], and DomainNet [49]. There are in total 31 categories and 4652 images in Office-31. The images are divided into three distinct domains, including Amazon (`A`), Webcam (`W`), and DSLR (`D`). There are ten categories that are shared between Caltech-256 and Office-31. Then we use these shared ten categories only in each domain of Office-31 as the target domain, and thus we can build six PDA tasks: `A31-D10`, `A31-W10`, `W31-D10`, `W31-A10`, `D31-W10`, and `D31-A10`.

ImageCLEF-DA is a benchmark for the ImageCLEF 2014 domain adaptation challenge. It contains four domains which are formed by selecting images from four public datasets, including Caltech-256 (`C`), ImageNet ILSVRC 2012 (`I`), Pascal VOC 2012 (`P`), and Bing (`B`). Each domain consists of 12 categories, and each category contains 50 images. Therefore, we build 12 PDA tasks: `I12-P6`, `P12-I6`, `I12-C6`, `C12-I6`, `P12-C6`, `C12-P6`, `B12-C6`, `B12-P6`, `B12-I6`, `I12-B6`, `C12-B6`, and `I12-B6`.

Office-Home [47] contains four domains of distinct styles: Artistic, Clip Art, Product, and Real-World, which are denoted as A, C, P, and R, respectively. There are 65 object categories in each domain. We use images from the first 25 classes in alphabetical order as the target domain and images from all the 65 classes as the source domain for PDA. Then we obtain 12 PDA tasks: `A65-C25`, `A65-P25`, `A65-R25`, `C65-A25`, `C65-P25`, `C65-R25`, `P65-A25`, `P65-C25`, `P65-R25`, `R65-A25`, `R65-C25`, and `R65-P25`.

DomainNet [49] is another large-scale challenging dataset, composed of six domains with 345 classes. Following [50], we use four domains with 126 classes, including Clipart, Painting, Real, and Sketch. The first 40 classes in alphabetical order are selected to build the target domain, and all the classes are used as the source domain. Therefore, we can also obtain 12 PDA tasks on the DomainNet dataset.

We compare the performance of the proposed approach with other state-of-the-art unsupervised domain adaptation and network compression methods, including fine-tuned CNN, deep adaptation network (**DAN**) [5], weighted domain adaptation network (**WDAN**) [7], selective adversarial networks (**SANs**) [12], example transfer networks (**ETNs**) [19], and **TCP** [39]. Note that importance weighted GAN (**IWGAN**) [18] is also a method for PDA. ETN [19] has made a comparison with [18] on the same tasks, in which [18] is dominated by ETN or SAN. So we only list SAN and ETN as baselines for comparison. We use the same backbone network as the baseline method in each task.

We implement all the approaches based on `PyTorch`. The training starts from VGG-16 and ResNet-50 model pretrained on ImageNet. The SWMMD layer is added before the last fully connected layer. We use mini-batch stochastic gradient descent (SGD) with a momentum of 0.9 and a weight decay of 5×10^{-4}. The number of epochs for fine-tuning is 15. The learning rate is dynamically adjusted during the training process using the rule applied in [12]: $\text{lr} = \text{lr}_0/(1 + 1.5t)^{\tau}$, where $\text{lr}_0 = 10^{-3}$, $\tau = 0.75$, and $t \in [0, 15]$ is the current epoch number. The user-defined penalty weights of SWMMD loss and entropy loss are gradually increasing during training.

TABLE I
PERFORMANCE COMPARISON ON THE OFFICE-31 DATASET WITH VGG-16 AS THE BASE NETWORK

Tasks	VGG [44]	DAN [5]	WDAN [7]	SAN [12]	ETN [19]	TCP [39]			Ours-Pr.	Ours		
	Acc.	Acc.	Acc.	Acc.	Acc.	Acc.	Param.	FLOPs	Acc.	Acc.	Param.	FLOPs
A31-W10	60.34	58.78	71.52	83.39	85.66	52.88	-65%	-57%	83.27	85.08	-63%	-88%
A31-D10	76.34	54.76	73.88	90.70	89.43	46.50	-63%	-60%	94.27	91.08	-61%	-87%
W31-A10	79.12	67.29	92.28	91.85	92.28	54.18	-56%	-59%	92.28	92.28	-62%	-76%
W31-D10	99.36	92.78	96.17	100.00	100.00	91.00	-58%	-61%	99.36	99.36	-68%	-84%
D31-A10	72.96	55.42	71.18	87.16	95.93	50.73	-63%	-51%	94.98	91.85	-57%	-64%
D31-W10	97.97	85.86	87.45	99.32	100.00	84.41	-61%	-64%	100.00	99.32	-58%	-70%
Average	81.03	69.15	82.08	92.07	93.88	63.28	-61%	-59%	94.02	93.16	-62%	-78%

TABLE II
PERFORMANCE COMPARISON ON THE OFFICE-31 DATASET WITH RESNET-50 AS THE BASE NETWORK

Tasks	ResNet [45]	DAN [5]	WDAN [7]	SAN [12]	ETN [19]	TCP [39]			Ours-Pr.	Ours		
	Acc.	Acc.	Acc.	Acc.	Acc.	Acc.	Param.	FLOPs	Acc.	Acc.	Param.	FLOPs
A31-W10	75.59	59.32	73.55	93.90	94.52	48.81	-59%	-50%	95.25	94.58	-70%	-80%
A31-D10	83.44	61.78	78.17	94.27	95.03	60.50	-49%	-40%	94.27	91.72	-63%	-50%
W31-A10	84.97	67.64	93.52	88.73	94.64	56.57	-58%	-50%	94.89	94.05	-73%	-60%
W31-D10	98.09	90.45	98.08	99.36	100.00	91.71	-67%	-60%	99.36	99.36	-80%	-70%
D31-A10	89.92	74.95	92.17	94.15	96.21	55.32	-57%	-50%	95.30	94.08	-63%	-50%
D31-W10	96.27	73.90	87.11	99.32	100.00	78.64	-67%	-60%	99.32	98.64	-71%	-60%
Average	87.05	71.34	87.10	94.96	96.73	65.26	-60%	-52%	96.40	95.41	-70%	-62%

TABLE III
PERFORMANCE COMPARISON ON THE OFFICE-HOME DATASET WITH RESNET-50 AS THE BASE NETWORK

Tasks	ResNet [45]	DAN [5]	WDAN [7]	SAN [12]	ETN [19]	TCP [39]			Ours-Pr.	Ours		
	Acc.	Acc.	Acc.	Acc.	Acc.	Acc.	Param.	FLOPs	Acc.	Acc.	Param.	FLOPs
A-C	46.33	43.76	48.66	44.42	59.24	42.23	-52%	-53%	59.40	60.60	-71%	-61%
A-P	67.51	67.90	60.17	68.68	77.03	50.78	-42%	-40%	75.07	75.18	-62%	-50%
A-R	75.87	77.47	80.18	74.60	79.54	70.27	-47%	-46%	84.59	85.31	-61%	-50%
C-A	59.14	63.73	67.68	67.49	62.92	46.64	-61%	-57%	69.70	67.40	-60%	-52%
C-P	59.94	58.99	62.24	64.99	65.73	45.46	-57%	-60%	63.19	66.78	-78%	-71%
C-R	62.73	67.59	75.32	77.80	75.01	56.32	-49%	-53%	75.04	77.08	-71%	-60%
P-A	58.22	56.84	72.18	59.78	68.29	40.17	-57%	-61%	70.43	70.16	-62%	-50%
P-C	41.79	37.07	52.18	44.72	55.37	36.12	-57%	-60%	57.01	57.97	-60%	-50%
P-R	74.88	76.37	81.39	80.07	84.37	67.95	-53%	-54%	84.04	84.65	-71%	-60%
R-A	67.40	69.15	75.57	72.18	75.72	55.96	-54%	-58%	76.22	74.20	-62%	-50%
R-C	48.18	44.30	56.66	50.21	57.66	35.82	-55%	-57%	60.54	53.67	-70%	-60%
R-P	74.17	77.48	81.29	78.66	84.54	67.86	-49%	-52%	81.23	81.18	-70%	-60%
Average	61.35	61.72	67.79	65.30	70.45	42.88	-53%	-54%	71.37	71.18	-67%	-56%

A similar way is applied in TCP [39]. Since model training should focus on the source data first, the cross-entropy loss should be relatively large. As the knowledge is retrieved during optimization, the model should switch to focus on the target dataset, and the SWMMD loss and the entropy loss should have larger weights. The update rule of the coefficients β and γ in (6) is set as $2/(1+e^{-i/I})-1$, where I is the total number of training iterations and $i \in [0, I]$ is the current iteration. The number of channels pruned P is set to 128. The specific hyperparameters are selected through cross-validation.

B. Results

In our experiments, three metrics are leveraged for evaluation on the PDA tasks, including classification accuracy on the target domain, model size, and total floating-point operations (FLOPs) of a complete inference. The FLOPs in a convolutional layer are calculated as $2HW(C_{\text{in}}K^2+1)C_{\text{out}}$, where H, W, and C_{in} are height, width, and number of channels of the input feature map, respectively. K is the kernel width and height, and C_{out} is the number of channels of the output feature map. For a fully connected layer, the FLOPs are $(2I-1)O$, where I and O are the input dimensionality and the output dimensionality of that layer, respectively.

First, we compare the performance of our method with WDAN, SAN, ETN, and TCP on the six PDA tasks of the Office-31 dataset, which is presented in Tables I and II. We use the model size and FLOPs of the original base networks as baselines and demonstrate model size and computation

TABLE IV
PERFORMANCE ON IMAGECLEF-DA WITH VGG-16 AS BASE NETWORK

Tasks	VGG [44]	DAN [5]	TCP [39]			Ours-Pr.	Ours		
	Acc.	Acc.	Acc.	Param.	FLOPs	Acc.	Acc.	Param.	FLOPs
P12-C6	94.00	92.67	63.67	-40%	-50%	94.00	96.00	-46%	-60%
C12-P6	77.67	74.00	54.33	-36%	-50%	86.33	86.67	-45%	-60%
P12-I6	88.67	85.67	61.33	-37%	-50%	88.00	88.67	-49%	-60%
I12-P6	88.00	83.00	60.33	-38%	-50%	88.00	89.00	-47%	-60%
C12-I6	82.33	82.00	56.00	-40%	-50%	90.00	87.33	-45%	-60%
I12-C6	96.00	94.67	73.67	-35%	-50%	98.33	98.00	-42%	-60%
B12-C6	86.00	59.33	66.33	-38%	-52%	94.67	94.67	-60%	-60%
B12-P6	70.33	50.33	47.37	-40%	-52%	81.67	77.00	-60%	-60%
B12-I6	72.00	58.67	45.33	-37%	-52%	84.67	81.33	-61%	-60%
P12-B6	58.00	43.00	43.67	-36%	-53%	64.33	62.33	-59%	-60%
C12-B6	55.33	49.00	48.33	-40%	-51%	59.67	57.00	-62%	-50%
I12-B6	59.67	48.33	45.00	-42%	-51%	66.33	64.33	-66%	-60%
Average	77.33	68.39	55.45	-38%	-51%	83.00	81.86	-54%	-59%

TABLE V
PERFORMANCE ON IMAGECLEF-DA WITH RESNET-50 AS BASE NETWORK

Tasks	ResNet [45]	DAN [5]	WDAN [7]	SAN [12]	TCP [39]			Ours-Pr.	Ours		
	Acc.	Acc.	Acc.	Acc.	Acc.	Param.	FLOPs	Acc.	Acc.	Param.	FLOPs
P12-C6	94.67	91.67	97.67	96.33	65.33	-58%	-50%	98.00	97.00	-72%	-60%
C12-P6	79.00	73.67	88.33	68.67	53.00	-58%	-51%	88.33	83.33	-73%	-60%
P12-I6	89.33	87.00	89.67	92.33	63.67	-53%	-51%	92.00	92.00	-71%	-60%
I12-P6	89.67	85.67	89.33	87.67	62.57	-57%	-51%	87.00	86.67	-73%	-60%
C12-I6	86.00	83.67	91.33	70.33	57.33	-58%	-50%	93.00	89.33	-73%	-60%
I12-C6	96.00	94.00	97.00	97.33	77.00	-58%	-50%	97.00	97.00	-73%	-60%
B12-C6	89.00	69.33	98.67	96.67	68.67	-60%	-50%	98.00	97.33	-71%	-60%
B12-P6	82.00	52.33	87.33	81.67	49.00	-60%	-50%	90.00	89.67	-60%	-50%
B12-I6	81.00	55.00	87.67	85.00	46.67	-60%	-50%	90.33	87.33	-62%	-50%
P12-B6	60.67	45.67	61.00	59.67	45.00	-60%	-50%	68.00	62.67	-62%	-50%
C12-B6	59.00	55.00	61.00	60.00	50.33	-60%	-50%	66.33	64.67	-61%	-50%
I12-B6	61.33	50.00	62.00	66.33	49.00	-60%	-51%	67.00	65.67	-72%	-60%
Average	80.64	64.33	84.25	80.17	57.30	-59%	-51%	86.25	84.35%	-69%	-57%

TABLE VI
PERFORMANCE COMPARISON ON THE DOMAINNET DATASET WITH RESNET-50 AS THE BASE NETWORK

Tasks	ResNet [45]	DANN [8]	SAN [12]	BA^{3}US [29]	Ours-Pr.	Ours		
	Acc.	Acc.	Acc.	Acc.	Acc.	Acc.	Param.	FLOPs
C-P	41.21	27.83	34.35	42.87	43.05	39.36	-43%	-30%
C-R	60.01	36.64	51.62	54.72	58.41	54.13	-41%	-30%
C-S	42.13	29.91	46.23	53.79	63.02	62.77	-43%	-30%
P-C	54.52	31.79	57.13	64.03	54.63	51.82	-40%	-30%
P-R	70.80	41.98	70.21	76.39	71.69	66.96	-30%	-20%
P-S	48.32	36.58	58.25	64.69	72.04	72.31	-20%	-29%
R-C	63.10	47.64	69.61	79.99	72.44	67.27	-40%	-30%
R-P	58.63	46.81	67.49	74.31	74.61	72.64	-41%	-30%
R-S	50.26	40.85	67.88	74.02	77.65	77.23	-40%	-30%
S-C	45.43	25.82	41.69	50.36	46.37	43.35	-40%	-30%
S-P	39.30	29.54	41.15	42.69	51.01	47.10	-39%	-30%
S-R	49.75	32.72	48.44	49.65	55.00	47.94	-39%	-30%
Average	51.96	35.68	54.50	60.63	61.66	58.57	-40%	-29%

reduction. The negative transfer can be circumvented well in SAN and ETN. However, once the network is trained, the model size computation overhead is the same as the original base network (i.e., VGG-16 or ResNet-50). TCP only considers fully shared label space between domains. Thus, the negative transfer issue is not well-addressed, leading to significant accuracy degradation when models become smaller. With the proposed SWMMD and pruning methods, all the three aspects are taken good care of. For comparison, we also disable the pruning process and train the network for PDA

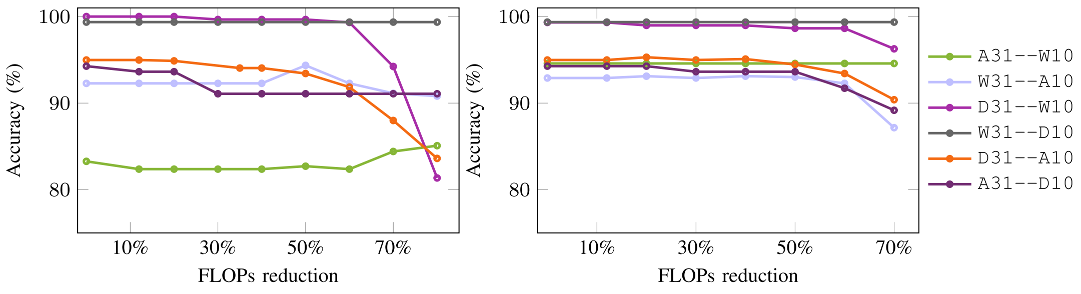


Fig. 4. Accuracy versus FLOPs reduction on Office-31. **Left**: VGG-16 as base network. **Right**: ResNet-50 as base network.

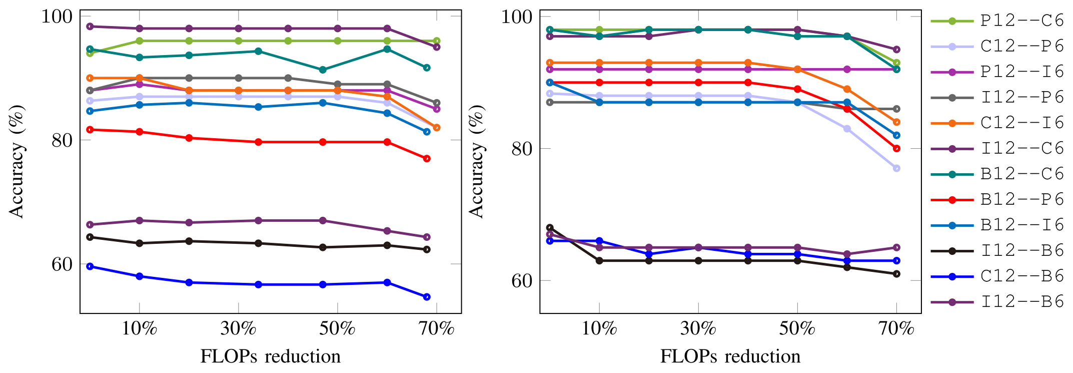


Fig. 5. Accuracy versus FLOPs reduction on ImageCLEF-DA. **Left**: VGG-16 as base network; **Right**: ResNet-50 as base network.

("Ours-Pr" in Tables I and II). Nearly 80% of FLOPs and more than 62% of parameters are reduced with less than 1% accuracy degradation compared with ETN. Notably, the accuracy results shown in Table I achieve the same level or even outperform ETN on specific tasks, which validates that the redundancy does exist and removing that can be beneficial to PDA. In addition, when ResNet-50 serves as the base network, 70% of parameters and 62% of FLOPs are reduced with only 1.3% loss on performance, which indicates that the proposed approach can generalize well to different CNN architectures.

Next, we compare the performance with the baseline method in the Office-Home dataset. We use the same backbone network as the baseline method SAN [12] and ETN [19], which is ResNet-50. The results are presented in Table III. It can be observed that although Office-Home is a much larger dataset, the proposed method can still achieve compelling accuracy with a substantial reduction in model size. Notably, enabling model pruning can achieve a nearly 70% reduction in model size and nearly 60% reduction on FLOPs, at the cost of merely 0.2% loss on classification accuracy. Moreover, the performance of the pruned network can still outperform all the baseline methods.

On the ImageCLEF-DA dataset, there are 12 tasks performed, including six tasks presented in TCP [39] and additional six tasks involving domain B. When comparing with the baseline methods SAN [12] and WDAN [7], we use the same backbone network, which is ResNet-50. DAN [5] ignores the change in the class prior distribution, and hence it will lead to negative transfer in PDA settings. Therefore, using base networks directly to perform domain adaptation can achieve higher accuracy than DAN. Suppose we do not apply the pruning process and only train the base network

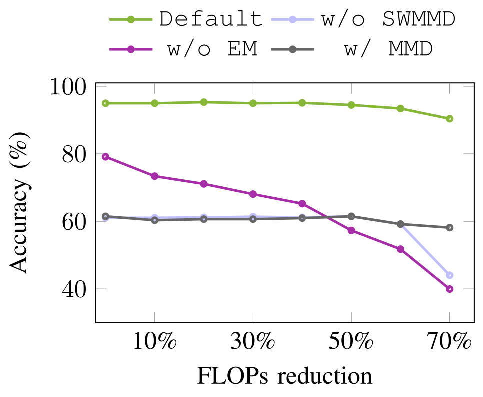


Fig. 6. Ablation for SWMMD and entropy minimization. Trade-off between FLOPs' reduction and accuracy on D31-A10.

with the SWMMD scheme. In that case, it can be observed from Tables IV and V that the accuracy can be improved by a large margin, which validates the effectiveness of the SWMMD method on negative transfer alleviation. If further combined with pruning, the redundancy is proven to exist and can be removed substantially. When the FLOPs are reduced by nearly 60%, the number of parameters can be reduced by more than 50% on VGG-16 and nearly 70% on ResNet-50.

In addition, higher accuracy can be attained on VGG-16 when the pruning process is enabled, even though the pruning ratio is as large as 60%, revealing that massive redundancy exists in VGG-16. For ResNet-50, the accuracy slightly degrades when the pruning ratio is 60%, as shown in Table V, which indicates that the redundancy issue is less severe than VGG-16 but still exists.

We also tested the proposed approach on the DomainNet dataset with ResNet-50 as the backbone. We keep the settings

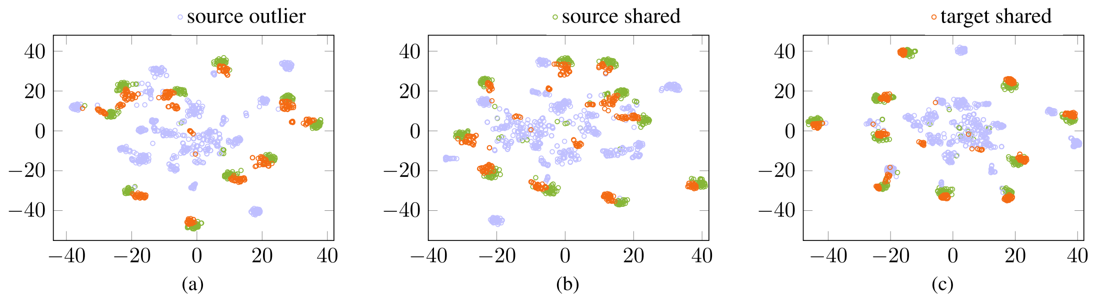


Fig. 7. t-SNE visualization on learned features on `A31-W10`. (a) ResNet-50. (b) DAN with ResNet-50. (c) Ours.

TABLE VII
ACCURACY WHEN FLOPs ARE REDUCED BY 60% WITH DIFFERENT η

η	1	0.75 (default)	0.5	0.25
A31-W10	94.52	94.58	94.58	94.52
A31-D10	91.72	91.72	91.72	91.72
W31-A10	93.01	92.28	93.11	93.11
W31-D10	98.09	98.64	98.73	98.73
D31-A10	91.13	93.42	91.13	93.32
D31-W10	97.97	99.36	97.63	96.27
Average	94.41	95.00	94.48	94.62

to be the same as other experiments and present the results in Table VI. If we do not apply the pruning step, the accuracy outperforms that of the other baseline methods. If pruning is enabled, eventually the FLOPs can be reduced by 30% and the number of parameters can be reduced by 40% with only 3% loss on performance. While comparing with other methods, the pruned model still demonstrates a fairly good accuracy.

To demonstrate more benefits of network compression, we further compare the inference latency between the original and compressed networks. As mentioned before, the comparison is conducted using both the VGG-16 network and the ResNet-50 network on various datasets. To ensure the measured latency is convincing, the inference is carried out on the entire dataset, and the overall runtime is recorded. The batch size remains the same all the time. Fig. 3 demonstrates the comparison of the normalized inference latency for different tasks with different networks. It can be observed that the latency of compressed VGG-16 can be reduced by nearly 50% compared with the original one, indicating a 2× speed-up. While the latency of compressed ResNet-50 can be reduced by roughly 40% compared with the original ResNet-50. The results show that compression can also help achieve faster computation.

C. Analysis

1) Pruning Impacts on the Accuracy: Intuitively, only when there exists considerable redundancy can we find that smaller size can boost the performance. To analyze whether the assumed intensive redundancy exists in PDA, the trade-off curves can provide insight into this problem. The relationship curves on the Office-31 dataset are shown in Fig. 4. With VGG-16 as the base network, three tasks (`D31-A10`, `D31-W10`, and `A31-D10`) reflect an obvious trade-off between accuracy and computation. During training, we set the exit condition as 80% reduction in FLOPs so that we can get a full view of the performance variations. The accuracy of the W31-D10 task keeps steady all the time until FLOPs are reduced by 80%, which validates the existence of significant redundancy. On the `A31-W10` and `W31-A10` tasks, there is even an increase in the accuracy as models get smaller. These observations reflect that transferring a sizeable pretrained model to a small-scale task is unnecessary, and slimming a model can be beneficial to both performance and efficiency. With ResNet-50 as the base network, noticeable trade-offs are observed in four out of six tasks. We set the exit condition as 70% reduction in FLOPs. Similarly, the accuracy of the `A31-W10` and `W31-D10` tasks are not impacted by the model size. While on ImageCLEF-DA, the trade-offs are reflected in nearly all the tasks on both bottleneck networks. Despite the performance degradation on end, which may be because pruning is already very aggressive, all the curves shown in Figs. 4 and 5 keep steady for a while, indicating the room and the necessity of performing model pruning.

2) Ablation Study: We proposed a specialized SWMMD formulation for PDA. In addition, there is another entropy minimization loss in (6). It is essential to conduct ablation studies for these components, denoted as `w/o MMD` and `w/o EM`, respectively. Moreover, we also compare with the pruning convergence compared with using the conventional MMD (denoted as `w/MMD`). During training, we set the exit condition as 70% reduction in FLOPs. We presented the accuracy versus FLOPs' reduction curve in Fig. 6. It can be observed that the performance degrades significantly when either SWMMD or entropy minimization is removed.

3) Impact of Parameters: In the proposed approach and experimental settings, we use an adaptive learning rate scaling which has a hyperparameter η. We then test its impact on the performance on the six PDA tasks in the Office-31 dataset. We use ResNet-50 as the backbone and evaluate the accuracy on each task when FLOPs are reduced by 60%. η is increased from 0.25 to 1. It can be seen from Table VII that the accuracy is under subtle change ($<1\%$) with respect to this scaling parameter.

4) Feature Visualization: We visualize the learned representations to demonstrate the effectiveness of different methods with the t-SNE method [51]. The representations learned in two tasks are presented, including `A31-W10` on Office-31 and `C12-P6` on ImageCLEF-DA, as shown in Fig. 7 and 8, respectively. Orange dots and green dots represent the shared classes between the source and target domains. Blue dots

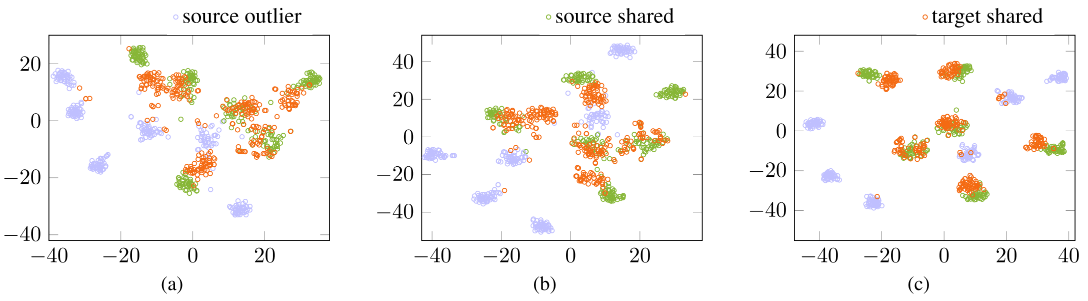


Fig. 8. t-SNE visualization on learned features on `C12-P6`. (a) ResNet-50. (b) DAN with ResNet-50. (c) Ours.

represent outlier classes. Strong alignment between orange dots and green dots indicates the effective circumvention of negative transfer. It can be observed that Figs. 7(a) and 8(a) show better alignment than Figs. 7(b) and 8(b), which is expected. Figs. 7(c) and 8(c) show the strongest alignment, which suggests the advantage of the proposed method.

V. Conclusion

This article presents a new perspective on the challenging PDA problem by integrating with neural network compression. An SWMMD is applied to match the cross-domain distribution when the label spaces are not identical in the source and target domains. On top of that, a channel pruning method is developed to iteratively prune channels based on the corresponding scaling factors in the BN layer. The experimental results indicate the proposed approach can simultaneously achieve compelling accuracy, smaller model size, and fewer computations compared with other model pruning and domain adaptation works. We hope this article will stimulate more future research in this area.

References

[1] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[2] Y. Ji, H. Zhang, Z. Zhang, and M. Liu, "CNN-based encoder–decoder networks for salient object detection: A comprehensive review and recent advances," *Inf. Sci.*, vol. 546, pp. 835–857, Feb. 2021.

[3] Z. Zhang, Y. Zhang, M. Xu, L. Zhang, Y. Yang, and S. Yan, "A survey on concept factorization: From shallow to deep representation learning," *Inf. Process. Manage.*, vol. 58, no. 3, May 2021, Art. no. 102534.

[4] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7167–7176.

[5] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *Proc. ICML*, 2015, pp. 97–105.

[6] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Unsupervised domain adaptation with residual transfer networks," in *Proc. NIPS*, 2016, pp. 136–144.

[7] H. Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, and W. Zuo, "Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2272–2281.

[8] Y. Ganin *et al.*, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2030–2096, 2016.

[9] S. J. Pan *et al.*, "Transfer learning via dimensionality reduction," in *Proc. AAAI*, 2008, pp. 677–682.

[10] P. P. Busto and J. Gall, "Open set domain adaptation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 754–763.

[11] K. Saito, S. Yamamoto, Y. Ushiku, and T. Harada, "Open set domain adaptation by backpropagation," in *Proc. ECCV*, Sep. 2018, pp. 153–168.

[12] Z. Cao, M. Long, J. Wang, and M. I. Jordan, "Partial transfer learning with selective adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2724–2732.

[13] Y. Luo, Z. Wang, Z. Huang, and M. Baktashmotlagh, "Progressive graph learning for open-set domain adaptation," in *Proc. ICML*, 2020, pp. 6468–6478.

[14] Z. Fang, J. Lu, F. Liu, J. Xuan, and G. Zhang, "Open set domain adaptation: Theoretical bound and algorithm," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 10, pp. 4309–4322, Oct. 2021.

[15] L. Zhong, Z. Fang, F. Liu, B. Yuan, G. Zhang, and J. Lu, "Bridging the theoretical bound and deep algorithms for open set domain adaptation," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Oct. 29, 2021, doi: 10.1109/TNNLS.2021.3119965.

[16] K. You, M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Universal domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2720–2729.

[17] K. Saito, D. Kim, S. Sclaroff, and K. Saenko, "Universal domain adaptation through self supervision," in *Proc. NIPS*, vol. 33, 2020, pp. 16282–16292.

[18] J. Zhang, Z. Ding, W. Li, and P. Ogunbona, "Importance weighted adversarial nets for partial domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8157–8164.

[19] Z. Cao, K. You, M. Long, J. Wang, and Q. Yang, "Learning to transfer examples for partial domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2985–2994.

[20] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2736–2744.

[21] Z. Huang and N. Wang, "Data-driven sparse structure selection for deep neural networks," in *Proc. ECCV*, Sep. 2018, pp. 304–320.

[22] S. Lin *et al.*, "Towards optimal structured CNN pruning via generative adversarial learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2790–2799.

[23] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2009.

[24] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 199–210, Feb. 2011.

[25] B. Gong, K. Grauman, and F. Sha, "Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation," in *Proc. ICML*, 2013, pp. 222–230.

[26] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," 2014, *arXiv:1412.3474*.

[27] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschläger, and S. Saminger-Platz, "Central moment discrepancy (CMD) for domain-invariant representation learning," 2017, *arXiv:1702.08811*.

[28] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Mach. Learn.*, vol. 79, nos. 1–2, pp. 151–175, May 2010.

[29] J. Liang, Y. Wang, D. Hu, R. He, and J. Feng, "A balanced and uncertainty-aware approach for partial domain adaptation," in *Proc. ECCV*, Aug. 2020, pp. 123–140.

[30] J. Dong, Y. Cong, G. Sun, B. Zhong, and X. Xu, "What can be transferred: Unsupervised domain adaptation for endoscopic lesions segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 4023–4032.

[31] J. Dong, Y. Cong, G. Sun, Z. Fang, and Z. Ding, "Where and how to transfer: Knowledge aggregation-induced transferability perception for unsupervised domain adaptation," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Nov. 16, 2021, doi: 10.1109/TPAMI.2021.3128560.

[32] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: ImageNet classification using binary convolutional neural networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Oct. 2016, pp. 525–542.

[33] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Speeding up convolutional neural networks with low rank expansions," in *Proc. Brit. Mach. Vis. Conf.*, 2014, pp. 1–13.

[34] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," in *Proc. ICLR*, 2016, pp. 1–14.

[35] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1389–1397.

[36] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," in *Proc. NIPS*, 2016, pp. 2074–2082.

[37] J. Ye, X. Lu, Z. Lin, and J. Z. Wang, "Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers," in *Proc. ICLR*, 2018, pp. 1–11.

[38] Y. He, P. Liu, Z. Wang, Z. Hu, and Y. Yang, "Filter pruning via geometric median for deep convolutional neural networks acceleration," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4340–4349.

[39] C. Yu, J. Wang, Y. Chen, and Z. Wu, "Accelerating deep unsupervised domain adaptation with transfer channel pruning," 2019, *arXiv:1904.02654*.

[40] Y. Grandvalet and Y. Bengio, "Semi-supervised learning by entropy minimization," in *Proc. NIPS*, 2005, pp. 529–536.

[41] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015, pp. 448–456.

[42] Y. Li, N. Wang, J. Shi, X. Hou, and J. Liu, "Adaptive batch normalization for practical domain adaptation," *Pattern Recognit.*, vol. 80, pp. 109–117, Aug. 2018.

[43] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," in *Proc. ICLR*, 2017, pp. 1–17.

[44] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–14.

[45] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[46] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *Proc. ECCV*, Sep. 2010, pp. 213–226.

[47] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5018–5027.

[48] J. Kalpathy-Cramer, H. Müller, S. Bedrick, I. Eggel, A. G. S. de Herrera, and T. Tsikrika, "The CLEF 2011 medical image retrieval and classification tasks," in *Proc. Working Notes (CLEF)*, 2011, pp. 1–6.

[49] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1406–1415.

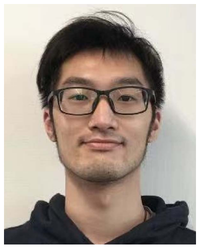
[50] K. Saito, D. Kim, S. Sclaroff, T. Darrell, and K. Saenko, "Semi-supervised domain adaptation via minimax entropy," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 8050–8058.

[51] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

Yuzhe Ma (Member, IEEE) received the Ph.D. degree from the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, in 2020.

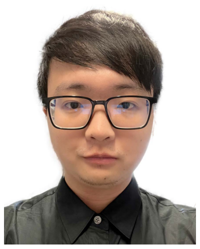
He is currently an Assistant Professor with the Microelectronics Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China. His research interests include agile VLSI design methodologies, machine-learning-aided VLSI design, and hardware-friendly machine learning.

Dr. Ma received the Best Paper Awards from the IEEE/ACM International Conference on Computer-Aided Design (ICCAD) 2021, the IEEE/ACM Asian and South Pacific Design Automation Conference (ASPDAC) 2021, and the IEEE International Conference on Tools with Artificial Intelligence (ICTAI) 2019 and the Best Paper Award Nomination from ASPDAC 2019.



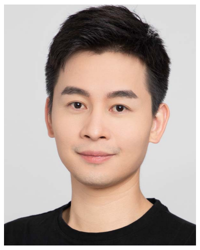
Xufeng Yao (Graduate Student Member, IEEE) received the B.Eng. degree in information system and information management from Fudan University (FDU), Shanghai, China, in 2016, and the M.Sc. degree in computer science from The Chinese University of Hong Kong (CUHK), Hong Kong, in 2020, where he is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering.

His research interests include computer vision and machine learning.



Ran Chen received the B.E. degree from the Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong, in 2018, where he is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering.

His research interests include machine learning application in VLSI design for manufacturing, yield learning, and pattern recognition.



Ruiyu Li (Member, IEEE) received the B.S. degree in computer science and technology from Sun Yat-sen University, Guangzhou, China, in 2014, and the Ph.D. degree in computer science and engineering from The Chinese University of Hong Kong, Hong Kong, in 2018.

He is currently a Co-Founder at SmartMore Corporation Limited, Hong Kong. His research interests include scene text recognition and interplay between vision and language.

Xiaoyong Shen (Member, IEEE) received the Ph.D. degree from the Computer Science and Engineering Department, The Chinese University of Hong Kong, Hong Kong, in 2016.

He is currently a Co-Founder and the CEO at SmartMore Corporation Limited, Hong Kong. His research interests include computer graphics and computer vision.

Bei Yu (Member, IEEE) received the Ph.D. degree from The University of Texas at Austin, Austin, TX, USA, in 2014.

He is currently an Associate Professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong.

Dr. Yu received nine Best Paper Awards from the IEEE/ACM Proceedings Design, Automation and Test in Europe (DATE) 2022, the IEEE/ACM International Conference on Computer-Aided Design (ICCAD) 2021 and 2013, the IEEE/ACM Asian and South Pacific Design Automation Conference (ASPDAC) 2021 and 2012, the IEEE International Conference on Tools with Artificial Intelligence (ICTAI) 2019, *Integration, the VLSI Journal* in 2018, the ACM International Symposium on Physical Design (ISPD) 2017, SPIE Advanced Lithography Conference 2016, and six ICCAD/ISPD contest awards. He has served as the TPC Chair for the ACM/IEEE Workshop on Machine Learning for CAD and in many journal editorial boards and conference committees. He is an Editor of IEEE TCCPS NEWSLETTER.