A Dictionary Approach to Domain-Invariant Learning in Deep Networks

Ze Wang[1], **Xiuyuan Cheng**[2], **Guillermo Sapiro**[2], **Qiang Qiu**[1]
Purdue University[1] Duke University[2]
{zewang, qqiu}@purdue.edu {xiuyuan.cheng, guillermo.sapiro}@duke.edu

Abstract

In this paper, we consider domain-invariant deep learning by explicitly modeling domain shifts with only a small amount of domain-specific parameters in a Convolutional Neural Network (CNN). By exploiting the observation that a convolutional filter can be well approximated as a linear combination of a small set of dictionary atoms, we show for the first time, both empirically and theoretically, that domain shifts can be effectively handled by decomposing a convolutional layer into a domain-specific atom layer and a domain-shared coefficient layer, while both remain convolutional. An input channel will now first convolve spatially only with each respective domain-specific dictionary atom to "absorb" domain variations, and then output channels are linearly combined using common decomposition coefficients trained to promote shared semantics across domains. We use toy examples, rigorous analysis, and real-world examples with diverse datasets and architectures, to show the proposed plug-in framework's effectiveness in cross and joint domain performance and domain adaptation. With the proposed architecture, we need only a small set of dictionary atoms to model each additional domain, which brings a negligible amount of additional parameters, typically a few hundred.

1 Introduction

Training supervised deep networks requires large amount of labeled training data; however, well-trained deep networks often degrade dramatically on testing data from a significantly different domain. In real-world scenarios, such domain shifts are introduced by many factors, such as different illumination, viewing angles, and resolutions. Research topics such as transfer learning and domain adaptation are studied to promote invariant representations across domains with different levels of availabilities of annotated data.

Recent efforts on learning cross-domain invariant representations using deep networks generally fall into two categories. The first one is to learn a common network with constraints encouraging invariant feature representations across different domains [17, 19, 34]. The feature invariance is usually measured by feature statistics like maximum mean discrepancy, or feature discriminators using adversarial training [6]. While these methods introduce no additional model parameters, the effectiveness largely depends on the degree of domain shifts. The other direction is to explicitly model domain specific characteristics with a multi-stream network structure where different domains are modeled by corresponding sub-networks at the cost of extra parameters and computations [26].

In this paper, we model domain shifts through domain-adaptive filter decomposition (DAFD) with layer branching. At a branched layer, we decompose each filter over a small set of domain-specific dictionary atoms to model intrinsic domain characteristics, while enforcing shared cross-domain decomposition coefficients to align invariant semantics. A regular convolution is now decomposed into two steps. First, a domain-specific dictionary atom convolves spatially only each individual input channel for shift "correction." Second, the "corrected" output channels are weighted summed using

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

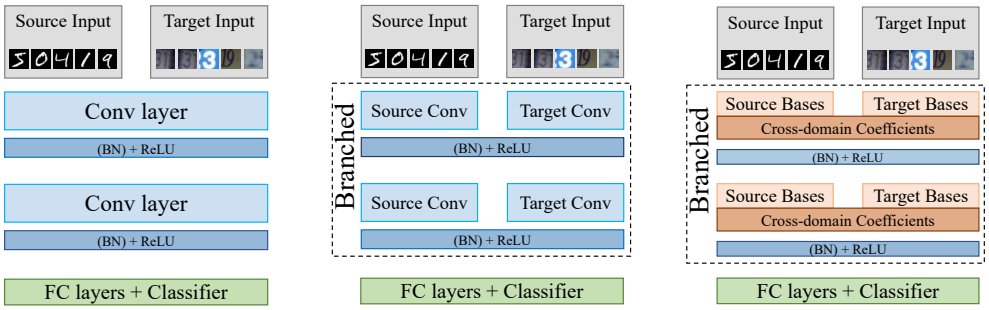
domain-shared decomposition coefficients (1×1 convolution) to promote common semantics. When domain shifts happen in space, we rigorously prove that such layer-wise "correction" by the same spatial transform applied to atoms suffices to align the learned features, contributing to the needed theoretical foundations in the field.

Comparing to the existing subnetwork-based methods, the proposed method has several appealing properties: First, only a very small amount of additional trainable parameters are introduced to explicitly model each domain, i.e., domain-specific atoms. The majority of the parameters in the network remain shared across domains, and learned from abundant training data to effectively avoid overfitting. Furthermore, the decomposed filters reduce the overall computations significantly compared to previous works, where computation typically grows linearly with the number of domains.

We conduct extensive real-world face recognition (with domain shifts and simultaneous multi-domains inputs), image classification, and segmentation experiments, and observe that, with the proposed method, invariant representations and performance across domains are consistently achieved without compromising the performance of individual domain.

Our main contributions are summarized as follows:

- We propose plug-in domain-invariant representation learning through filter decomposition with layer branching, where domain-specific atoms are learned to counter domain shifts, and semantic alignments are enforced with cross-domain common decomposition coefficients.
- We both theoretically prove, contributing the much needed foundations in CNN-based invariant learning, and empirically demonstrate that by stacking the atom-decomposed branched layer, invariant representations across domains are achieved progressively.
- The majority of network parameters remain shared across domains, which alleviates the demand for massive annotated data from every domain, and introduces only a small amount of additional computation and parameter overhead. Thus the proposed approach serves as an efficient way for domain invariant learning and its applications to domain shifts and simultaneous multi-domain tasks.



(a) Regular CNN (b) Basic Branching (c) Branching with DAFD

Figure 1: Three candidate architectures considered for domain-invariant representation learning. In (a), a set of common network parameters are trained to model both source and target domains. In (b), the domain characteristics are explicitly modeled by two sets of convolutional filters in each convolutional layer. Our approach is illustrated in (c) where domain-adaptive atoms are learned to "absorb" domain shifts, while the decomposition coefficients are shared across domains to promote and exploit common semantics.

2 Domain-adaptive Filter Decomposition for Invariant Learning

A straightforward way to address domain shifts is to learn from multi-domain training data a single network as in Figure 1(a). However, the lack of explicitly modelling of individual domains often results in unnecessary information loss and performance degradation as discussed in [26]. Thus, we often simultaneously observe underfitting for domains with abundant training data, and overfitting for domains with limited training. In this section, we start with a simplistic pedagogical formulation, domain-adaptive layer branching as in Figure 1(b), where domain shifts are modeled by a respective

branch of filters in a layer, one branch per domain. Each branch is learned only from domain-specific data, while non-branched layers are learned from data from all domains. We then propose to extend basic branching to atom-decomposed branching as in Figure 1(c), where domain characteristics are modeled by domain-specific dictionary atoms, and shared decomposition coefficients are enforced to align cross-domain semantics.

2.1 Pedagogical Branching Formulation

We start with the simple-minded branching formulation in Figure 1(b). To model the domain-specific characteristics, at the first several convolutional layers, we dedicate a separate branch to each domain. Domain shifts are modeled by an independent set of convolutional filters in the branch, trained respectively with errors propagated back from the loss functions of source and target domains. For supervised learning, the loss function is the cross-entropy for each domain. For unsupervised learning, the loss function for the target domain can be either the feature statistics loss or the adversarial loss. The remaining layers are shared across domains. We assume one target domain and one source domain in our discussion, while multiple domains are supported. Note that, though we adopt the source vs. target naming convention in the domain adaptation literature, we address here a general domain-invariant learning problem.

Domain-adaptive branching is simple and straightforward, however, it has the following drawbacks: First, both the number of model parameters and computation are multiplied with the number of domains. Second, with limited target domain training data, we can experience overfitting in determining a large amount of parameters dedicated to that domain. Third, no constraints are enforced to encourage cross-domain shared semantics. We address these issues through layer branching with the proposed domain-adaptive filter decomposition.

2.2 Atom-decomposed Branching

To simultaneously counter domain shifts and enforce cross-domain shared semantics, we decompose each convolutional filter in a branched layer into domain-specific dictionary atoms, and cross-domain shared coefficients, as illustrated in Figure 2.

In our approach, we decompose source and target domain filters over domain-adaptive dictionary atoms, with decomposition coefficients shared across domains. Specifically, at each branched layer, the source domain filter W_s and target domain filter W_t of size $L \times L \times C' \times C$, are decomposed as $W_s = \psi_s a$ and $W_t = \psi_t a$, where ψ_s and ψ_t with a size of $L \times L \times K$ are K domain-adaptive dictionary atoms for source and target domains, respectively; and $a \in \mathbb{R}^{K \times C' \times C}$ denotes the common decomposed coefficients shared across domains. Contrary to single domain works such as [23, 30, 35, 36] that incorporate dictionaries into CNNs, the domain-adaptive dictionary atoms are independently learned from the corresponding domain data to model domain shifts, and the shared decomposed coefficients are learned from the massive data from multiple domains. Note that thanks to this proposed structure, only a small amount of additional parameters is required here to model each additional domain, typically a few hundred.

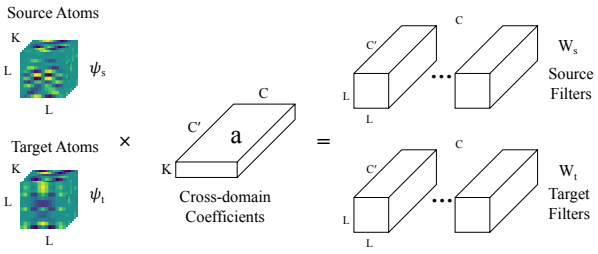


Figure 2: The proposed domain-adaptive filter decomposition for domain-invariant learning.

With the above domain-adaptive filter decomposition, at each branched layer, a regular convolution is now decomposed into two: First, a domain-specific atom convolves each individual input channel for domain shift "correction." Second, the "corrected" output channels are weighted summed using domain-shared decomposition coefficients (1×1 convolution) to promote common semantics. A toy example is presented in supplementary material Figure A.1 for illustrating the intuition behind the reason why manipulating dictionary atoms alone can address domain shifts. We generate target domain data by applying two trivial operations to source domain images: First, every 3×3 non-overlapping patch in each image is locally rotated by 90^o. Then, images are negated by multiplying with -1. Domain-invariant features are observed by manipulating dictionary atoms alone. We will

rigorously prove in Section 3 why such layer-wise "correction" aligns features across domains, and present real-world examples in the experiments.

Parameters and Computation Reduction. Suppose that both input and output features have the same spatial resolution of $W \times W$, in each forward pass in a regular convolutional layer, there are totally $W^2 \times C' \times C \times (2L^2 + 1)$ flops for each domain. While in our model, each domain only introduces $W^2 \times C' \times 2K(L^2 + C)$ flops, where K is the number of dictionary atoms. For parameters, there are totally $D \times C' \times C \times L^2$ parameters in a regular convolutional layer where D is the number of domains which is typically 2 in our case. In our model, each layer has only $K \times (C' \times C + D \times L^2)$ parameters. Taking VGG-16 [29] as an example with an input size of 224×224, a regular VGG-16 with branching, Fig 1(b) and [25, 26], requires adding 14.71M parameters and 15.38G flops in convolutional layers to handle each additional domain. With the proposed method (Fig 1(c)), VGG-16 only requires adding 702 parameters and 10.75G flops to handle one additional domain (K=6).

3 Provable Invariance with Adaptive Atoms

In this section, we theoretically prove that the features produced by the source and target networks from domain-transferred inputs can be aligned by the proposed framework of only adjusting multi-layer atoms, assuming a generative model of the source and target domain images via CNN. Since convolutional generative networks are a rich class of models for domain transfer [13, 22], our analysis provides a theoretical justification of the proposed approach, providing a contribution to the theoretical foundations of domain adaptation. Diverse examples in the experiment section show the applicability of the proposed approach is potentially larger than what is proved here. All proofs are in the supplementary material Section D.

Filter Transform via Atom Transform. Let w_s and w_t be the filters in the branched convolutional layer for the source and target domains respectively, and similarly denote the source and target atoms by $\psi_{k,s}$ and $\psi_{k,t}$. In the proposed atom decomposition architecture, the source and target domain filters are linear combinations of the domain-specific dictionary atoms with shared decomposition coefficients, namely

$$w_s(u) = \sum_k a_k \psi_{k,s}(u), \quad w_t(u) = \sum_k a_k \psi_{k,t}(u).$$

Certain transforms of the filter can be implemented by only transforming the dictionary atoms, including

(1) A linear correspondence of filter values. Let $\lambda : \mathbb{R} \to \mathbb{R}$ be a linear mapping, by linearity,

$$\psi_{k,s}(u) \to \psi_{k,t}(u) = \lambda(\psi_{k,s}(u)) \text{ applies } w_s(u) \to w_t(u) = \lambda(w_s(u)).$$

E.g. the negation $\lambda(\xi) = -\lambda(\xi)$, as shown in supplementary material Figure A.1.

(2) The transformation induced by a displacement of spatial variable, i.e., "spatial transform" of filters, defined as $D_\tau w(u) = w(u - \tau(u))$, where $\tau : \mathbb{R}^2 \to \mathbb{R}^2$ is a differentiable displacement field. Note that the dependence on spatial variable u in a filter is via the atoms, thus $\psi_{k,s} \to \psi_{k,t} = D_\tau \psi_{k,s}$ applies $w_s \to w_t = D_\tau w_s$.

If such filter adaptations are desired in the branching network, then it suffices to branch the dictionary atoms while keeping the coefficients a_k shared, as implemented in the proposed architecture shown in Figure 1(c). A fundamental question is thus how large is the class of possible domain shifts that can be corrected by these "allowable" filter transforms. In the rest of the section, we show that if the domain shifts in the images are induced from a generative CNN where the filters for source and target differ by a sequence of allowable transforms, then the domain shift can be provably eliminated by another sequence of filter transforms which can be implemented by atom branching only.

Provable Invariance. Stacking the approximate commuting relation, *Lemma 1* in supplementary material Section D, in multiple layers allows to correct a sequence of filter transforms in previous convolutional layers by another sequence of "symmetric" ones. This means that if we impose a convolutional generative model on the source and target input images, and assume that the domain transfer results from a sequence of spatial transforms of filters in the generative net, then by correcting

these filter transforms in the subsequent convolutional layers we can guarantee the recovery of the same feature mapping. The key observation is that the filter transfers can be implemented by atoms transfer only.

We summarize the standard theoretical assumptions as follows:

(A1) The nonlinear activation function σ in any layer is non-expansive,

(A2) In the generative net (where layer is indexed by negative integers), $w_t^{(-l)} = D_l w_s^{(-l)}$, where $D_l = D_{\tau_l}$, τ_l is odd and $|\nabla \tau_l|_\infty \leq \varepsilon < \frac{1}{5}$ for all $l = 1, \cdots, L$. The biases in the target generative net are mildly adjusted accordingly due to technical reasons (to preserve the "baseline output" from zero-input, c.f. detail in the proof).

(A3) In the generative net, $\|w_s^{(-l)}\|_1 \leq 1$ for all l, and so is $w_t^{(-l)} = D_l w_s^{(-l)}$. Same for the feed-forward convolutional net taking the generated images as input, called "feature net": The source net filters have $\|w_s^{(l)}\|_1 \leq 1$ for $l = 1, 2 \cdots$, and same with $D_l w_s^{(l)}$ which will be set to be $w_t^{(l)}$. Also, $w_s^{(-l)}$ and $w_s^{(l)}$ are both supported on $2^{j_l} B$ for $l = 1, \cdots, L$.

One can show that $\|D_\tau w\|_1 = \|w\|_1$ when $(I_d - \rho)$ is a rigid motion, and generally $|\|D_\tau w\|_1 - \|w\|_1| \leq c|\nabla \tau|_\infty \|w\|_1$ which is negligible when ε is small. Thus in (A3) the boundedness of the 1-norm of the source and target filters imply one another exactly or approximately. The boundedness of 1-norm of the filters preserves the non-expansiveness of the mapping from input to output in a convolutional layer, and in practice is qualitatively preserved by normalization layers. Also, as a typical setting, (A3) assumes that the scales j_l in the generative net (the $(-l)$-th layer) and the feature net (the l-th layer) are matched, which simplifies the analysis and can be relaxed.

Theorem 1. *Suppose that X_s and X_t are source and target images generated by L-layer generative CNN nets with source and target filters $w_s^{(-l)}$, $w_t^{(-l)}$ respectively from the common representation h. Under (A1)-(A3), the output at the L-th layer of the target feature CNN from X_t, by setting $w_t^{(l)} = D_l w_s^{(l)}$ in all layers which can be implemented by atom branching, approximates that of the source feature CNN from X_s up to an error which is bounded in 1-norm by $4\varepsilon \left\{ (\sum_{l=1}^{L} 2^{j_l}) \|\nabla h\|_1 + 2L\|h\|_1 \right\}$, and the second term vanishes if $(I_d - \tau_l)$ are rigid motions, e.g., rotation.*

4 Experiments

In this section, we perform extensive experiments to evaluate the performance of the proposed domain-adaptive filter decomposition. We start with the comparisons among the 3 architectures listed in Figure 1 on two supervised tasks. To demonstrate the proposed framework as one principled way for domain-invariant learning, we then conduct a set of domain adaptation experiments. There we show, by simply plugging the proposed domain filter decomposition into regular CNNs used in existing domain adaptation methods, we consistently observe performance improvements, which well-illustrate that our method is orthogonal to other domain adaptation methods.

4.1 Architecture Comparisons

We start with two supervised tasks performed on the three architectures listed in Figure 1, regular CNN (A1), basic branching (A2), and branching with domain-adaptive filter decomposition (A3). The networks with DAFD are trained end-to-end with a summed loss for domains, and the domain-specific atoms are only updated by the error from the corresponding domain, while the decomposition coefficients are updated by the joint error across domains.

Supervised domain adaptation on images. The first task is supervised domain adaptation, where we adopt a challenging setting by using MNIST as the source domain, and SVHN as the target domain. We perform a series of experiments by progressively reducing the annotated training data for the target domain. We start the comparisons at 10% of the target domain labeled samples, and end at 0.5% where only 366 labeled samples are available for the target domain. The results on test set for both domains are presented in Table 1. It is clearly shown that when training the target domain with small amount of data, a network with basic branching suffers from overfitting to the target domain

Table 1: Accuracy (%) on MNIST->SVHN for supervised domain adaptation. A1, A2, and A3 correspond to regular CNN, basic branching, and branching with DAFD shown in Figure 1, respectively.

Scales	Source domain			Target domain		
	0.1	0.05	0.005	0.1	0.05	0.005
A1	98.4	96.4	98.0	81.6	80.2	61.0
A2	99.2	98.6	97.6	81.4	78.4	49.6
A3	**99.4**	**98.8**	**98.8**	**85.6**	**82.2**	**64.4**

Table 2: Cross-domain simultaneous face recognition on NIR-VIS-2.0. A1, A2, and A3 correspond to regular CNN, basic branching, and branching with domain-adaptive filter decomposition shown in Figure 1, respectively.

Methods	VIS Acc (%)	NIR Acc (%)	NIR+VIS (%)
A1	75.57	52.71	98.44
A2	94.46	87.50	98.58
A3	**97.16**	**95.03**	**99.15**

because of the large amount of domain specific parameters. While regular CNN generates well on target domain, the performance on source domain degrades when the number of target domain data is comparable. A network with the proposed domain-adaptive filter decomposition significantly balances the learning of both the source and the target domain, and achieves best accuracies on both domains regardless of the amount of annotated training data for the target domain. The feature space of the three candidate architectures are visualized in Figure 3.

(a) Feature space in (a) (b) Feature space in (b) (c) Feature space in (c)

Figure 3: The feature space of the three candidate architectures in Figure 1, MNIST $\rightarrow$ SVHN, are visualized using t-SNE [21] in (a), (b), (c), respectively. The obtained superior cross-domain invariance of the proposed framework can be clearly observed in (c).

Supervised simultaneous cross-domain face recognition. Besides standard domain adaptation, the proposed domain-adaptive filter decomposition can be extended to general tasks that involves more than one visual domain; domain adaptation is performed without loosing the power of the original domain and multiple-domains can be simultaneously exploited. Here we demonstrate this by performing experiments on supervised cross-domain face recognition. We adopt the NIR-VIS 2.0 [16], which consists of 17,580 NIR (near infrared) and VIS (visible light) face images of 725 subjects, and perform cross-domain face recognition. We adopt VGG16 as the base network structure, branch all the convolutional layers with the proposed domain-adaptive filter decomposition, and train the network from scratch. In each convolutional layer, two set of dictionary atoms are trained for modeling the NIR and the VIS domain, respectively. Specifically, one VIS image and one NIR image are fed *simultaneously* to the network, and the feature vectors of both domains are averaged to produce the final cross-domain feature, which is further fed into a linear classifier for classifying the identity. While the training is conducted using both domains simultaneously, we test the network under three settings including feeding single domain inputs only (VIS Acc and NIR Acc in Table 2) and both domain inputs (VIS+NIR Acc in Table 2). Quantitative comparisons demonstrate that branching with the proposed DAFD performs superiorly even with a missing input domain. Note that A2 requires additional 14.71M parameters over A1, while our method requires only 0.0007M as shown in supplementary material Table A.1.

4.2 Experiments on Standard Domain Adaptations

In this section, we perform extensive experiments on unsupervised domain adaptation. Note that the objective of the experiments in this section is not to validate the proposed domain-adaptive filter decomposition as just another new method for domain adaptation. Instead, since most of the state-of-the-art domain adaptation methods adopt the regular CNN (A1) with completely shared parameters for domains, we show the compatibility and the generality of the proposed domain-adapting

filter decomposition by plugging it into underlying domain adaptation methods, and evaluate the effectiveness by retraining the networks using exactly the same setting and observing the performance improvement over the underlying methods. Diverse real-world domain shifts including different sensors, different image sources, and synthetic images, and applications on both classification and segmentation are examined in these experiments. Together with the experiments in the previous section, this further stresses the plug-and-play virtue of the proposed framework.

In practise, instead of learning independent source and target domain atoms, we learn the *residual* between the source and the target domain atoms. The residual is initialized by full zeros, and trained by loss for encouraging invariant features in the underlying methods, e.g., the adversarial loss in ADDA [33]. We consistently observe that this stabilizes the training and promotes faster convergence.

Table 3: Accuracy (%) on Digits for unsupervised domain adaptation.

Methods	M → U	U → M	S → M	Avg.
DANN	-	-	73.9	-
ADDA	89.4	90.1	76.0	85.1
CDAN+E	95.6	98.0	89.2	94.3
DANN + DAFD	92.0	95.2	82.1 (**11.1**% ↑)	89.8
ADDA + DAFD	91.4	94.8	82.9	89.7 (**5.5**% ↑)
CDAN+E + DAFD	96.8	98.8	96.6	97.4 (**3.2**% ↑)

Image classification. We perform experiments on three public digits datasets: MNIST, USPS, and Street View House Numbers (SVHN), with three transfer tasks: USPS to MNIST (U → M), MNIST to USPS (M → U), and SVHN to MNIST (S → M). Classification accuracy on the target domain test set samples is adopted as the metric for measuring the performance. We perform domain-adaptive domain decomposition on state-of-the-art methods DANN [6], ADDA [33], and CDAN+E [18]. Quantitative comparisons are presented in Table 3, demonstrating significant improvements over underlying methods.

Office-31. Office-31 [27] is one of the most widely used datasets for visual domain adaptation, which has 4,652 images and 31 categories collected from three distinct domains: Amazon (**A**), Webcam (**W**), and DSLR (**D**). We evaluate all methods on six transfer tasks **A** → **W**, **D** → **W**, **W** → **D**, **A** → **D**, **D** → **A**, and **W** → **A**. Two feature extractors, AlexNet [14] and ResNet [9] are adopted for fair comparisons with underlying methods. Specifically, ImageNet initialization are widely used for ResNet in the experiments with Office-31, and we consistently observe that initialization is important for the training on Office-31. Therefore, when training ResNet based networks with domain-adaptive filter decomposition, we initialize the feature extractor using parameters decomposed from ImageNet initialization. The quantitative comparisons are in Table 4.

Table 4: Accuracy (%) on Office-31 for unsupervised domain adaptation (AlexNet and ResNet).

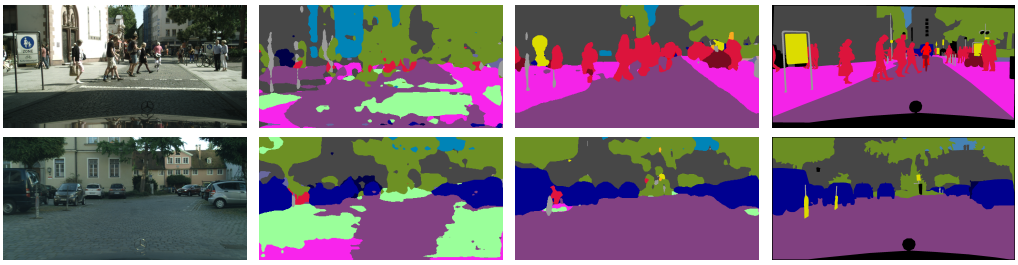
	Method	A → W	D → W	W → D	A → D	D → A	W → A	Avg.
AlexNet	AlexNet (no adaptation)	61.6±0.5	95.4±0.3	99.0±0.2	63.8±0.5	51.1±0.6	49.8±0.4	70.1
	DANN [6]	73.0±0.5	96.4±0.3	99.2±0.3	72.3±0.3	53.4±0.4	51.2±0.5	74.3
	ADDA [33]	73.5±0.6	96.2±0.4	98.8±0.4	71.6±0.4	54.6±0.5	53.5±0.6	74.7
	DANN + DAFD	74.4±0.3	97.1±0.4	99.1±0.4	74.2±0.3	56.8±0.5	53.1±0.7	75.8 (**2.3**% ↑)
	ADDA + DAFD	77.2±0.5	97.9±0.4	98.5±0.2	73.2±0.4	55.4±0.6	57.8±0.5	76.7 (**2.7**% ↑)
ResNet	ResNet-50 (no adaptation)	68.4±0.2	96.7±0.1	99.3±0.1	68.9±0.2	62.5±0.3	60.7±0.3	76.1
	DANN [6]	82.0±0.4	96.9±0.2	99.1±0.1	79.7±0.4	68.2±0.4	67.4±0.5	82.2
	ADDA [33]	86.2±0.5	96.2±0.3	98.4±0.3	77.8±0.3	69.5±0.4	68.9±0.5	82.9
	CDAN+E [18]	94.1±0.1	98.6±0.1	100.0±.0	92.9±0.2	71.0±0.3	69.3±0.3	87.7
	CAN [5]	94.5±0.3	99.1±0.2	99.8±0.2	95.0±0.3	78.0±0.3	77.0±0.3	90.6
	DANN + DAFD	86.4±0.4	96.8±0.2	99.2±0.1	84.4±0.4	70.5±0.4	68.8±0.4	84.35 (**2.3**% ↑)
	ADDA + DAFD	86.8±0.4	97.7±0.1	98.4±0.1	80.5±0.3	71.1±0.4	69.1±0.5	83.9 (**1.2**% ↑)
	CDAN+E + DAFD	95.6±0.1	98.8±0.1	100.0±0.0	93.5±0.2	76.6±0.5	71.3±0.4	89.3 (**1.8**% ↑)
	CAN + Ours	95.2±0.2	99.2±0.2	99.9±0.1	96.1±0.2	78.9±0.3	78.2±0.3	91.27 (**0.7**% ↑)

Image segmentation. Beyond image classification tasks, we perform a challenging experiment on image segmentation to demonstrate the generality of the proposed domain-adaptive filter decomposition. We perform unsupervised adaptation from the GTA dataset [24] (images generated from video games) to the Cityscapes dataset [4] (real-world images), which has a significant practical value considering the expensive cost on collecting annotations for image segmentation in real-world scenarios. Two underlying methods FCNs in the wild [12] and AdaptSegNet [31] are adopted for comprehensive comparisons. Based on the underlying methods, all the convolutional layers are decomposed using domain-adaptive filter decomposition, and all the transpose-convolutional layers are kept sharing by both domains. For quantitative results in Table 5, we use intersection-over-union,

i.e., IoU = $\frac{\text{TP}}{\text{TP+FP+FN}}$, where TP, FP, and FN are the numbers of true positive, false positive, and false negative pixels, respectively, as the evaluation metric. As with the previous examples, our method improves all state-of-the-art architectures. Qualitative results are shown in Figure 4 and supplementary material Figure A.2, and data samples are in Figure 5.

Table 5: Unsupervised DA for semantic segmentation: GTA → Cityscapes

Methods	IoU	Class-wide IoU																		
		road	sidewalk	building	wall	fence	pole	t-light	t-sign	veg	terrain	sky	person	rider	car	truck	bus	train	mbike	bicycle
No Adapt (VGG)	17.9	26.0	14.9	65.1	5.5	12.9	8.9	6.0	2.5	70.0	2.9	47.0	24.5	0.0	40.0	12.1	1.5	0.0	0.0	0.0
No Adapt (ResNet)	36.6	75.8	16.8	77.2	12.5	21.0	25.5	30.1	20.1	81.3	24.6	70.3	53.8	26.4	49.9	17.2	25.9	6.5	25.3	36.0
FCN WLD (VGG)	27.1	70.4	32.4	62.1	14.9	5.4	10.9	14.2	2.7	79.2	21.3	64.6	44.1	4.2	70.4	8.0	7.3	0.0	3.5	0.0
AdaptSegNet (VGG)	35.0	87.3	29.8	78.6	21.1	18.2	22.5	21.5	11.0	79.7	29.6	71.3	46.8	6.5	80.1	23.0	26.9	0.0	10.6	0.3
AdaptSegNet (ResNet)	42.4	86.5	36.0	79.9	23.4	23.3	23.9	35.2	14.8	83.4	33.3	75.6	58.5	27.6	73.7	32.5	35.4	3.9	30.1	28.1
FCN WLD + DAFD	32.7 (**20.7**% ↑)	76.4	36.7	68.8	17.6	5.8	11.1	13.9	2.9	80.0	24.4	69.1	47.5	4.3	74.4	14.1	6.3	0.0	2.1	0.0
AdaptSegNet (VGG) + DAFD	36.4 (**4.0**% ↑)	86.7	35.3	78.8	22.8	14.5	23.9	21.9	18.2	82.1	32.2	66.8	49.6	10.1	81.2	19.6	27.1	1.1	11.4	4.2
AdaptSegNet (ResNet) + DAFD	45.0 (**6.1**% ↑)	88.2	38.5	8.12	25.0	23.8	22.9	35.1	14.4	84.9	34.1	79.9	59.5	29.1	75.5	30.1	35.2	2.9	28.7	29.1



(a) Target domain image. (b) Before adaptation. (c) After adaptation. (d) Ground truth.

Figure 4: Qualitative results for domain adaptation segmentation. The samples are randomly selected from the validation subsets of Cityscapes.

(a) Source domain (*GTA*: video game images). (b) Target domain (Cityscapes: real-world images).

Figure 5: Dataset samples for segmentation experiments (video games → street views).

5 Related Work

Recent achievements on domain-invariant learning generally follow two directions. The first direction is learning a single network,which is encouraged to produce domain-invariant features by minimizing additional loss functions in the network training [6, 17, 20, 19, 34]. The Maximum Mean Discrepancy (MMD) [7], and MK-MMD [8] in [17], are adopted as the discrepancy metric among domains. Beyond the first order statistic, second-order statistics are utilized in [10]. Besides the hand-crafted distribution distance metrics, [6, 32, 18] resort to adversarial training and achieve superior performances. Various distribution alignment methods, e.g., [37, 15], are proposed to improve the invariant feature learning. While effective in certain scenarios, the performance of learning invariant features using a shared network is largely constrained by the degree of domain shift as discussed in [26]. Meanwhile, some recent works like [37, 39] suggest important insights on whether it is sufficient to do domain adaptation by invariant representation and small empirical source risk, which shed light on exploring more effective alignment methods that are robust to common issues like different marginal label distributions. Another popular direction is modeling each domain explicitly using auxiliary network structures. [1] proposes feature representation by two components where domain similarities and shifts are modeled by a private component and a shared component separately. A completely two-stream network structure is proposed in [26], where auxiliary residual networks are trained to adapt the layer parameters of the source to the target domain. [3] proposes attacking domain shifts by domain-specific batch normalization, which we believe is compatible with the proposed DAFD

for better performance. Another popular direction for domain adaptation is to remap the input data between the source and the target domain for domain adaptation [22, 13, 11], which is not included in the discussion since we are focusing on learning invariant feature space. Also, as discussed in [26], while remarkable performances are witnessed by adopting pseudo-labels [17, 28, 38], we consider adopting pseudo-labels as a plug-and-play improvement that can be equipped to our method, but does not align with the main focus of our research. Finally, learning invariance is of relevance beyond domain adaptation, e.g., in the field of causal inference [2].

6 Conclusion

We proposed to perform domain-invariant learning through domain-adaptive filter decomposition. To model domain shifts, convolutional filters in a deep convolutional network are decomposed over domain-adaptive dictionary atoms to counter domain shifts, and cross-domain decomposition coefficients are constrained to unify common semantics. We present the intuitions of countering domain shifts by adapting atoms through toy examples, and further provide theoretical analysis. Extensive experiments on multiple tasks and network architectures with significant improvements validate that, by stacking domain-adaptive branched layers with filter decomposition, complex domain shifts in real-world scenarios can be bridged to produce domain-invariant representation, which are reflected by both experimental results and feature space visualizations, all this at virtual no additional memory or computational cost when adding domains.

7 Broader Impact

In this paper, we introduced a plug-in framework to explicitly model domain shifts in CNNs. With the proposed architecture, we need only a small set of dictionary atoms to model each additional domain, which brings a negligible amount of additional parameters, typically a few hundred. We consider our plug-and-play method a general contribution to deep learning, assuming no particular application.

8 Acknowledgements

Work partially supported by NSF, ONR, ARO, NGA, NIH, and gifts from Microsoft, Google, and Amazon.

References

[1] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. In *NIPS*, 2016.

[2] Peter Bühlmann. Invariance, causality and robustness. *arXiv preprint arXiv:1812.08233*, 2018.

[3] Woong-Gi Chang, Tackgeun You, Seonguk Seo, Suha Kwak, and Bohyung Han. Domain-specific batch normalization for unsupervised domain adaptation. In *CVPR*, 2019.

[4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[5] Guoliang Kang et al. Contrastive adaptation network for unsupervised domain adaptation. In *CVPR*, 2019.

[6] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

[7] Arthur Gretton, Karsten M Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex J Smola. A kernel method for the two-sample-problem. In *NIPS*, 2007.

[8] Arthur Gretton, Dino Sejdinovic, Heiko Strathmann, Sivaraman Balakrishnan, Massimiliano Pontil, Kenji Fukumizu, and Bharath K Sriperumbudur. Optimal kernel choice for large-scale two-sample tests. In *NIPS*, 2012.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[10] Judy Hoffman, Sergio Guadarrama, Eric S Tzeng, Ronghang Hu, Jeff Donahue, Ross Girshick, Trevor Darrell, and Kate Saenko. LSDA: Large scale detection through adaptation. In *NIPS*, 2014.

[11] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. CyCADA: Cycle-consistent adversarial domain adaptation. In *ICML*, 2018.

[12] Judy Hoffman, Dequan Wang, Fisher Yu, and Trevor Darrell. FCNs in the wild: Pixel-level adversarial and constraint-based adaptation. *arXiv preprint arXiv:1612.02649*, 2016.

[13] Lanqing Hu, Meina Kan, Shiguang Shan, and Xilin Chen. Duplex generative aaversarial network for unsupervised domain qdaptation. In *CVPR*, 2018.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[15] Abhishek Kumar, Prasanna Sattigeri, Kahini Wadhawan, Leonid Karlinsky, Rogerio Feris, Bill Freeman, and Gregory Wornell. Co-regularized alignment for unsupervised domain adaptation. In *NIPS*, pages 9345–9356, 2018.

[16] Stan Li, Dong Yi, Zhen Lei, and Shengcai Liao. The casia nir-vis 2.0 face database. In *CVPR Workshops*, 2013.

[17] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I Jordan. Transferable representation learning with deep adaptation networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(12):3071–3085, 2019.

[18] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *NIPS*, 2018.

[19] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. In *NIPS*, 2016.

[20] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. *ICML*, 2017.

[21] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[22] Zak Murez, Soheil Kolouri, David Kriegman, Ravi Ramamoorthi, and Kyungnam Kim. Image to image translation for domain adaptation. *CVPR*, 2018.

[23] Qiang Qiu, Xiuyuan Cheng, Robert Calderbank, and Guillermo Sapiro. DCFNet: Deep neural network with decomposed convolutional filters. *ICML*, 2018.

[24] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016.

[25] Artem Rozantsev, Mathieu Salzmann, and Pascal Fua. Beyond sharing weights for deep domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[26] Artem Rozantsev, Mathieu Salzmann, and Pascal Fua. Residual parameter transfer for deep domain adaptation. In *CVPR*, 2018.

[27] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, 2010.

[28] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. Asymmetric tri-training for unsupervised domain adaptation. *ICML*, 2017.

[29] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2014.

[30] Jeremias Sulam, Vardan Papyan, Yaniv Romano, and Michael Elad. Multilayer convolutional sparse modeling: Pursuit and dictionary learning. *IEEE Transactions on Signal Processing*, 66(15):4090–4104, 2018.

[31] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *CVPR*, 2018.

[32] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *ICCV*, 2015.

[33] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, 2017.

[34] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

[35] Ze Wang, Xiuyuan Cheng, Guillermo Sapiro, and Qiang Qiu. Acdc: Weight sharing in atom-coefficient decomposed convolution. *arXiv preprint arXiv:2009.02386*, 2020.

[36] Ze Wang, Xiuyuan Cheng, Guillermo Sapiro, and Qiang Qiu. Stochastic conditional generative networks with basis decomposition. In *ICLR*, 2020.

[37] Yifan Wu, Ezra Winston, Divyansh Kaushik, and Zachary Lipton. Domain adaptation with asymmetrically-relaxed distribution alignment. In *ICML*, 2019.

[38] Weichen Zhang, Wanli Ouyang, Wen Li, and Dong Xu. Collaborative and adversarial network for unsupervised domain adaptation. In *CVPR*, 2018.

[39] Han Zhao, Remi Tachet Des Combes, Kun Zhang, and Geoffrey Gordon. On learning invariant representations for domain adaptation. In *ICML*, 2019.