AutoDIAL: Automatic DomaIn Alignment Layers

Fabio Maria Carlucci
Sapienza, Roma, Italy

Lorenzo Porzi
IRI CSIC-UPC, Barcelona, Spain
Mapillary, Graz, Austria

Barbara Caputo
Sapienza, Roma, Italy

Elisa Ricci
FBK, Trento, Italy
University of Perugia, Italy

Samuel Rota Bulò
FBK, Trento, Italy
Mapillary, Graz, Austria

Abstract

Classifiers trained on given databases perform poorly when tested on data acquired in different settings. This is explained in domain adaptation through a shift among distributions of the source and target domains. Attempts to align them have traditionally resulted in works reducing the domain shift by introducing appropriate loss terms, measuring the discrepancies between source and target distributions, in the objective function. Here we take a different route, proposing to align the learned representations by embedding in any given network specific Domain Alignment Layers*, designed to match the source and target feature distributions to a reference one. Opposite to previous works which define* a priori *in which layers adaptation should be performed, our method is able to* automatically *learn the degree of feature alignment required at different levels of the deep network. Thorough experiments on different public benchmarks, in the unsupervised setting, confirm the power of our approach.*

1. Introduction

In spite of the progress brought by deep learning in visual recognition, the ability to generalize across different visual domains is still out of reach. The assumption that training (source) and test (target) data are independently and identically drawn from the same distribution does not hold in many real world applications. Indeed, it has been shown that, even with powerful deep learning models, the domain shift problem can be alleviated but not removed [7].

In the last few years the research community has devoted significant efforts in addressing domain shift. In this context, the specific problem of unsupervised domain adaptation, *i.e.* no labeled data are available in the target domain, deserves special attention. In fact, in many applications annotating data is a tedious operation or may not be possible at all. Several approaches have been proposed, both considering hand-crafted features [15, 11, 12, 20, 8] and deep models [21, 31, 9, 23, 10, 19]. In particular, recent works based on deep learning have achieved remarkable performance. Most of these methods attempt to reduce the discrepancy among source and target distributions by learning features that are invariant to the domain shift. Two main strategies are traditionally employed. One is based on the minimization of Maximum Mean Discrepancy (MMD) [21, 23]: the distributions of the learned source and target representations are optimized to be as similar as possible by minimizing the distance between their mean embeddings. The other strategy [31, 9] relies on the domain-confusion loss, introduced to learn an auxiliary classifier predicting if a sample comes from the source or from the target domain. Intuitively, by maximizing this term, *i.e.* by imposing the auxiliary classifier to exhibit poor performance, domain-invariant features can be obtained.

More recently, researchers have also started to investigate alternative directions [10, 3, 19, 4], such as the use of encoder-decoder networks to jointly learn source labels and reconstruct unsupervised target images, or the possibility of reducing the domain shift by designing specific distribution normalization layers. In particular, the latter idea is exploited in [19], where a simple parameter-free approach for deep domain adaptation, called Adaptive Batch Normalization (AdaBN), is proposed. Inspired by the popular Batch Normalization (BN) technique [16], AdaBN modifies the Inception-BN network and aligns the learned source and target representations by using different mean/variance terms for the source and target domain when performing BN at test time. This leads to learning domain-invariant features without requiring additional loss terms (*e.g.* MMD, domain-confusion) in the optimization function and the associated extra-parameters.

Inspired by [19], this paper introduces novel *Domain Alignment* layers (DA-layers) (Fig.1) which are embedded at different levels of the deep architecture to align the

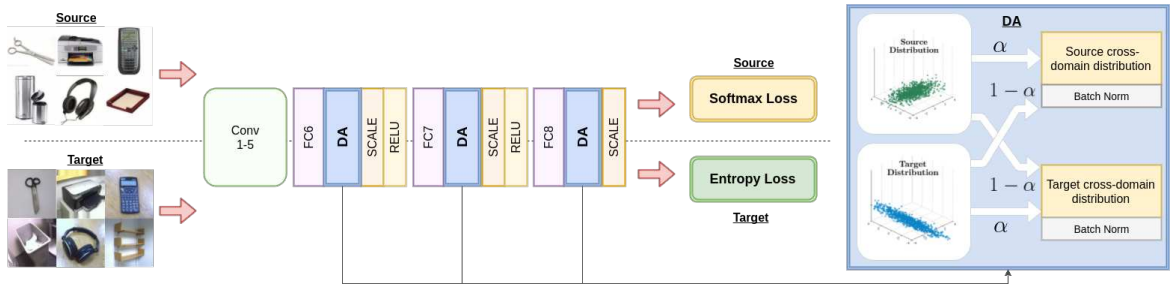

Figure 1. AutoDIAL as applied on AlexNet [18]. Source and target images are fed to the network. After passing through the first layers, they enter our DA-layer where source and target distributions are aligned. The DA-layer learns the statistics of newly defined *source and target cross domain distributions* and normalize the source and target mini-batches according to the computed mean and variance, different for the two domains (see Section 3.1). The amount by which each distribution is influenced by the other and therefore the degree of domain alignment, depends on a parameter, $\alpha \in [0.5, 1.0]$, which is also automatically learned. After flowing through the whole network, source samples contribute to a Softmax loss, while target samples contribute to an Entropy loss, which promotes classification models which maximally separate unlabeled data. Note that we use multiple DA-layers to align learned feature representations at different levels.

learned source and target feature distributions to a canonical one. Different from [19] and all previous deep domain adaptation methods which decide *a priori* which layers should be adapted, we endow our DA-layers with the ability to *automatically* learn the degree of alignment that should be pursued at different levels of the network. This is to our knowledge the first work that tries to pursue this objective. Furthermore, we argue that in [19] unlabeled target data are not fully exploited (see Sec. 3.1). Instead, we leverage information from the target domain to construct a prior distribution on the network parameters, biasing the learned solution towards models that are able to separate well the classes in the target domain (see Sec. 3.2 and [13]). Our DA-layers and the considered prior distribution work in synergy during learning: the first aligning the source and target feature distributions, the second encouraging the network to learn features that lead to maximally separated target classes. We call our algorithm AutoDIAL – DomaIn Alignment Layers with Automatic alignment parameters. An extensive experimental evaluation demonstrates that AutoDIAL greatly alleviates the domain discrepancy and outperforms state of the art techniques on three popular benchmarks: Office-31 [26], Office-Caltech [12] and the Caltech-ImageNet setting of the Cross-Dataset Testbed[30].

Contributions. The contribution of this work is threefold. First, we present an approach for unsupervised domain adaptation, based on the introduction of DA-layers to explicitly address the domain shift problem, which act in synergy with an entropy loss which exploits unsupervised target data during learning. Our solution simultaneously aligns feature representations and learns where and to which extent adaptation should take place. Second, in contrast to previous works optimizing domain discrepancy regularization terms [23, 31, 9, 21], our DA-layers do not require any additional meta-parameters. Third, we perform an extensive experimental analysis on three different benchmarks. We find that our unsupervised domain adaptation approach outperforms state-of-the-art methods and can be applied to different CNN architectures, consistently improving their performance in domain adaptation problems.

2. Related Work

Unsupervised domain adaptation focuses on the scenario where labeled data are only available in the source domain. Traditional methods addressed the problem of reducing the discrepancy between the source and the target distributions by considering two main strategies. The first is based on instance re-weighting [15, 5, 33, 11, 34]. Initially, source samples are assigned different importance according to their similarity with the target data. Then, the re-weighted instances are used to learn a classification/regression model for the target domain. Following this scheme, Huang *et al.* [15] introduced Kernel Mean Matching, a nonparametric method to set source sample weights without explicitly estimating the data distributions. Gong *et al.* [11] proposed to automatically discover landmark datapoints, *i.e.* the subset of source instances being more similar to target data, and used them to create domain-invariant features. Chu *et al.* [5] formalized the two tasks of sample selection and classifier learning within a single optimization problem. While these works considered hand-crafted features, recently similar ideas have been applied to deep models. For instance, Zeng *et al.* [34] described an unsupervised domain adaptation approach for pedestrian detection using deep autoencoders to weight the importance of source training samples.

A second strategy for unsupervised domain adaptation is based on feature alignment, *i.e.* source and target data are projected in a common subspace as to reduce the distance among the associated distributions. This approach attracted considerable interest in the past years and several different methods have been proposed, both considering shallow models [12, 20, 8] and deep architectures [21, 31, 9, 10, 3]. Focusing on recent deep domain adaptation methods, two different schemes are typically considered for aligning fea-

ture representations: (i) multiple adaptation schemes are introduced in order to reduce Maximum Mean Discrepancy [21, 23, 28] or (ii) deep features are learned in a domain-adversarial setting, *i.e.* maximizing a domain confusion loss [31, 9]. Our approach belongs to the category of methods employing deep learning for domain adaptation. However, we significantly depart from previous works, reducing the discrepancy between source and target distributions by introducing a domain alignment approach based on DA-layers. The closest work to ours is [19], where Li *et al.* propose to use BN in the context of domain adaptation. Our approach can be seen as a generalization of [19], as our DA layers allows to automatically tune the required degree of adaptation at each level of the deep network. Furthermore, we also introduce a prior over the network parameters in order to fully benefit from the target samples during training. Experiments presented in Section 4 show the significant added value of our idea.

3. Automatic DomaIn Alignment Layers

Let $\mathcal{X}$ be the input space (*e.g.* images) and $\mathcal{Y}$ the output space (*e.g.* image categories) of our learning task. In unsupervised domain adaptation, we have a *source* domain and a *target* domain that are identified via probability distributions $p^s_{\mathtt{xy}}$ and $p^t_{\mathtt{xy}}$, respectively, defined over $\mathcal{X} \times \mathcal{Y}$. The two distributions are in general different and unknown, but we are provided with a source dataset $\mathcal{S} = \{(x^s_1, y^s_1), \ldots, (x^s_n, y^s_n)\}$ of *i.i.d.* observations from $p^s_{\mathtt{xy}}$ and an unlabeled target dataset $\mathcal{T} = \{x^t_1, \ldots, x^t_m\}$ of *i.i.d.* observations from the marginal $p^t_{\mathtt{x}}$. The goal is to estimate a predictor from $\mathcal{S}$ and $\mathcal{T}$ that can be used to classify sample points from the target domain. This task is particularly challenging because on one hand we lack direct observations of labels from the target domain and on the other hand the discrepancy between the source and target domain distributions prevents a predictor trained on $\mathcal{S}$ to be readily applied to the target domain.

A number of state of the art methods try to reduce the domain discrepancy by performing some form of alignment at the feature or classifier level. In particular, the recent, most successful methods try to *couple* the training process and the domain adaptation step within *deep* neural architectures [9, 23, 21], as this solution enables alignments at different levels of abstraction. The approach we propose embraces the same philosophy, while departing from the assumption that domain alignment can by pursued by applying the *same* predictor to the source and target domains. This is motivated by an impossibility theorem [2], which intuitively states that no learner relying on the *covariate shift* hypothesis, *i.e.* $p^s_{\mathtt{y}|\mathtt{x}} = p^t_{\mathtt{y}|\mathtt{x}}$, and achieving a low discrepancy between the source and target unlabeled distributions $p^s_{\mathtt{x}}$ and $p^t_{\mathtt{x}}$, is guaranteed to succeed in domain adaptation without further relatedness assumptions between training and target distributions. For this reason, we assume that the source and target predictors are in general *different* functions. Nonetheless, both predictors depend on a common parameter θ belonging to a set Θ, which couples explicitly the two predictors, while not being directly involved in the alignment of the source and target domains. This contrasts with the majority of state of the art methods that augment the loss function used to train their predictors with a regularization term penalizing discrepancies between source and target representations (see, *e.g.* [9, 23, 21]). The perspective we take is different and is close in spirit to AdaBN [19]. It consists in hard-coding the desired domain-invariance properties into the source and target predictors through the introduction of so-called *Domain-Alignment layers* (DA-layers). Moreover, we sidestep the problem of deciding which layers should be aligned, and to what extent, by endowing the architecture with the ability to *automatically* tune the degree of alignment that should be considered in each domain-alignment layer. The rest of this section is devoted to providing the details of our method.

3.1. Source and Target Predictors

The source and target predictors are implemented as two deep neural networks being almost identical, as they share the same structure and the same weights (given by the parameter θ). However, the two networks contain also a number of special layers, the DA-layers, which implement a domain-specific operation. Indeed, the role of such layers is to apply a data transformation that aligns the observed input distribution with a reference distribution. Since in general the input distributions of the source and target predictors differ, while the reference distribution stays the same, we have that the two predictors undergo different transformations in the corresponding DA-layers. Consequently, the source and target predictors de facto implement different functions, which is important for the reasons given in Sec. 3.

The actual implementation of our DA-layers is inspired by AdaBN [19], where Batch Normalization layers are used to independently align source and target distributions to a standard normal distribution, by matching the first- and second-order moments. The approach they propose consists in training on the source a network having BN-layers, thus obtaining the source predictor, and deriving the target predictor as a post-processing step, which re-estimates the BN statistics using target samples only. Accordingly, the source and target predictors share the same network parameters but have different BN statistics, thus rendering the two predictors different functions.

The approach we propose sticks to the same idea of using BN-layers to align domains, but we introduce fundamental changes. One limitation of AdaBN is that the target samples have no influence on the network parameters, as they are not observed during training. Our approach overcomes this limitation by coupling the network parameters

to both target and source samples at training time. This is achieved in two ways: first we introduce a prior distribution for the network parameters based on the target samples; second, we endow the architecture with the ability of learning the degree of adaptation by introducing a parametrized, cross-domain bias to the input distribution of each domain-specific DA-layer. The rest of this subsection is devoted to describe the new layer, while we defer to the next subsection the description of the prior distribution.

DA-layer. As mentioned before, our DA-layer is derived from Batch Normalization, but as opposed to BN, which computes first and second-order moments from the input distribution derived from the mini-batch, we let the latter statistics to be contaminated by samples from the other domain, thus introducing a cross-domain bias. Since the source and target predictors share the same network topology, each DA-layer in one predictor has a matching DA-layer in the other predictor. Let x_s and x_t denote inputs to matching DA-layers in the source and target predictor, respectively, for a given feature channel and spatial location. Assume q^s and q^t to be the distribution of x_s and x_t, respectively, and let $q^{st}_\alpha = \alpha q^s + (1-\alpha)q^t$ and, symmetrically, $q^{ts}_\alpha = \alpha q^t + (1-\alpha)q^s$ be cross-domain distributions mixed by a factor $\alpha \in [0.5, 1]$. Then, the output of the DA-layers in the source and target networks are given respectively by

$$\texttt{DA}(x_s;\alpha) = \frac{x_s - \mu_{st,\alpha}}{\sqrt{\epsilon + \sigma^2_{st,\alpha}}},\ \texttt{DA}(x_t;\alpha) = \frac{x_t - \mu_{ts,\alpha}}{\sqrt{\epsilon + \sigma^2_{ts,\alpha}}}, \quad (1)$$

where $\epsilon > 0$ is a small number to avoid numerical issues in case of zero variance, $\mu_{st,\alpha} = \texttt{E}_{x\sim q^{st}_\alpha}[x]$, $\sigma^2_{st,\alpha} = \texttt{Var}_{x\sim q^{st}_\alpha}[x]$, and similarly $\mu_{ts,\alpha}$ and $\sigma^2_{st,\alpha}$ are mean and variance of $x \sim q^{ts}_\alpha$. Akin to BN, we estimate the statistics based on the mini-batch and derive similarly the gradients through the statistics (see Supplementary Material).

The rationale behind the introduction of the mixing factor α is that we can move from having an independent alignment of the two domains akin to AdaBN, when $\alpha = 1$, to having a coupled normalization when $\alpha = 0.5$. In the former case the DA-layer computes two different functions in the source and target predictors and is equivalent to considering a full degree of domain alignment. The latter case, instead, yields the same function since $q^{st}_{0.5} = q^{ts}_{0.5}$ thus transforming the two domains equally, which yields no domain alignment. Since the mixing parameter α is not fixed a priori but learned during the training phase, we obtain as a result that the network can decide how strong the domain alignment should be at each level of the architecture where DA-layer is applied. More details about the actual CNN architectures used to implement the two domain predictors are given in Section 4.1.

3.2. Training

During the training phase we estimate the parameter θ, which holds the neural network weights shared by the source and target predictors including the mixing factors pertaining to the DA-layers, using the observations provided by the source dataset $\mathcal{S}$ and the target dataset $\mathcal{T}$. As we stick to a discriminative model, the unlabeled target dataset cannot be employed to express the data likelihood. However, we can exploit $\mathcal{T}$ to construct a prior distribution of the parameter θ. Accordingly, we shape a posterior distribution of θ given the observations $\mathcal{S}$ and $\mathcal{T}$ as

$$\pi(\theta|\mathcal{S},\mathcal{T}) \propto \pi(y_\mathcal{S}|x_\mathcal{S},\mathcal{T},\theta)\pi(\theta|\mathcal{T},x_\mathcal{S})\,, \quad (2)$$

where $y_\mathcal{S} = \{y^s_1,\dots,y^s_n\}$ and $x_\mathcal{S} = \{x^s_1,\dots,x^s_n\}$ collect the labels and data points of the observations in $\mathcal{S}$, respectively. The posterior distribution is maximized over Θ to obtain a maximum a posteriori estimate $\hat{\theta}$ of the parameter used in the source and target predictors:

$$\hat{\theta} \in \arg\max_{\theta\in\Theta} \pi(\theta|\mathcal{S},\mathcal{T})\,. \quad (3)$$

The term $\pi(y_\mathcal{S}|x_\mathcal{S},\mathcal{T},\theta)$ in (2) represents the likelihood of θ with respect to the source dataset, while $\pi(\theta|\mathcal{T},x_\mathcal{S})$ is the prior term depending on the target dataset, which acts as a regularizer in the classical learning theory sense. Both terms actually, depend on both domains due to the cross-domain statistics that we have in our DA-layers for $\frac{1}{2} \leq \alpha < 1$ and are estimated from samples from the source *and* target domains.

The likelihood decomposes into the following product over sample points, due to the data sample *i.i.d.* assumption:

$$\pi(y_\mathcal{S}|x_\mathcal{S},\mathcal{T},\theta) = \prod_{i=1}^{n} f^\theta_s(y^s_i;x^s_i)\,, \quad (4)$$

where $f^\theta_s(y^s_i;x^s_i)$ is the probability that sample point x^s_i takes label y^s_i according to the source predictor (we omitted the dependence on $\mathcal{T}$ and $x_\mathcal{S}$ for notational convenience).

Before delving into the details of the prior term, we would like to remark on the absence of an explicit component in the probabilistic model that tries to align the source and target distributions. This is because in our model the domain-alignment step is taken over by each predictor, independently, via the domain-alignment layers as shown in the previous subsection.

Prior distribution. The prior distribution of the parameter θ shared by the source and target predictors is constructed from the observed, target data distribution. This choice is motivated by the theoretical possibility of squeezing more bits of information from unlabeled data points insofar as they exhibit low levels of class overlap [24]. Accordingly, it is reasonable to bias a priori a predictor based

on the degree of label uncertainty that is observed when the same predictor is applied to the target samples. Uncertainty in this sense can be measured for an hypothesis θ in terms of the empirical entropy of $\mathtt{y}|\theta$ conditioned on $\mathtt{x}$ as follows

$$h(\theta|\mathcal{T}, x_{\mathcal{S}}) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{y\in\mathcal{Y}} f_t^{\theta}(y; x_i^t)\log f_t^{\theta}(y; x_i^t)\,, \quad (5)$$

where $f_t(y; x_i^t)$ represents the probability that sample point x_i^t takes label y according to the target predictor (again we omitted the dependence on $\mathcal{T}$ and $x_{\mathcal{S}}$).

It is now possible to derive a prior distribution $\pi(\theta|\mathcal{T}, x_{\mathcal{S}})$ in terms of the label uncertainty measure $h(\theta|\mathcal{T}, x_{\mathcal{S}})$ by requiring the prior distribution to maximize the entropy under the constraint $\int h(\theta|\mathcal{T}, x_{\mathcal{S}})\pi(\theta|\mathcal{T}, x_{\mathcal{S}})d\theta = \varepsilon$, where the constant $\varepsilon > 0$ specifies how small the label uncertainty should be on average. This yields a concave, variational optimization problem with solution:

$$\pi(\theta|\mathcal{T}, x_{\mathcal{S}}) \propto \exp\left(-\lambda\, h(\theta|\mathcal{T}, x_{\mathcal{S}})\right)\,, \quad (6)$$

where λ is the Lagrange multiplier corresponding to ε. The resulting prior distribution satisfies the desired property of preferring models that exhibit well separated classes (*i.e.* having lower values of $h(\theta|\mathcal{T}, x_{\mathcal{S}})$), thus enabling the exploitation of the information content of unlabeled target observations within a discriminative setting [13].

Prior distributions of this kind have been adopted also in other works [23] in order to exploit more information from the target distribution, but has never been used before in conjunction to explicit domain alignment methods (*i.e.* not based on additional regularization terms such as MMD and domain-confusion) like the one we are proposing.

Inference. Once we have estimated the optimal network parameters $\hat{\theta}$ by solving (3), we can remove the dependence of the target predictor on $\mathcal{T}$ and $x_{\mathcal{S}}$. In fact, after fixing $\hat{\theta}$, the input distribution to each DA-layer also becomes fixed, and we can thus compute and store the required statistics once at all, akin to standard BN.

3.3. Implementation Notes

DA-layer can be implemented as a mostly straightforward modification of standard Batch Normalization. We refer the reader to the supplementary material for a complete derivation. In our implementation in particular, we treat each pair of DA-layers as a single network layer which simultaneously computes the two normalization functions in Equation (1) and learns the α parameter. During training each batch contains a fixed number of source samples, followed by a fixed number of target samples, allowing our DA-layers to simply differentiate between the two. Similarly to standard BN, we keep separate moving averages of the source and target statistics. Note that, as mentioned before, $\alpha \in [0.5, 1]$. We enforce this by clipping its value in the allowed range in each forward pass of the network.

By replacing the optimization problem in (3) with the equivalent minimization of the negative logarithm of $\pi(\theta|\mathcal{S}, \mathcal{T})$ and combining (2), (4), (5) and (6) we obtain a loss function $L(\theta) = L^s(\theta) + \lambda L^t(\theta)$, where:

$$L^s(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\log f_s^{\theta}(y_i^s; x_i^s)\,,$$

$$L^t(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{y\in\mathcal{Y}} f_t^{\theta}(y; x_i^t)\log f_t^{\theta}(y; x_i^t)\,.$$

The term $L^s(\theta)$ is the standard log-loss applied to the source samples, while $L^t(\theta)$ is an entropy loss applied to the target samples. The second term can be implemented by feeding $f_t^{\theta}(y; x_i^t)$ to both inputs of a cross-entropy loss layer, where supported by the deep learning toolkit of choice. In our implementation, based on Caffe [17], we obtain it by slightly modifying the existing `SoftmaxLoss` layer.[1]

4. Experiments

In this section we extensively evaluate our approach and compare it with state of the art unsupervised domain adaptation methods. We also provide a detailed analysis of the proposed framework, demonstrating empirically the effect of our contributions. Note that all the results in the following are reported as averages over five training/testing runs.

4.1. Experimental Setup

Datasets. We evaluate the proposed approach on three publicly-available datasets.

The **Office 31**[26] dataset is a standard benchmark for testing domain-adaptation methods. It contains 4652 images organized in 31 classes from three different domains: Amazon (A), DSRL (D) and Webcam (W). Amazon images are collected from `amazon.com`, Webcam and DSLR images were manually gathered in an office environment. In our experiments we consider all possible source/target combinations of these domains and adopt the *full protocol* setting [11], *i.e.* we train on the entire labeled source and unlabeled target data and test on annotated target samples.

The **Office-Caltech** [12] dataset is obtained by selecting the subset of 10 common categories in the Office31 and the Caltech256[14] datasets. It contains 2533 images of which about half belong to Caltech256. Each of Amazon (A), DSLR (D), Webcam (W) and Caltech256 (C) are regarded as separate domains. In our experiments we only consider the source/target combinations containing C as either the source or target domain.

[1]The source code is available at `https://github.com/ducksoup/autodial`

To further perform an analysis on a large-scale dataset, we also consider the recent **Cross-Dataset Testbed** introduced in [30] and specifically the **Caltech-ImageNet** setting. This dataset was obtained by collecting the images corresponding to the 40 classes shared between the Caltech256 (C) and the Imagenet (I) [6] datasets. To facilitate comparison with previous works [31, 29, 27] we perform experiments in two different settings. The first setting, adopted in [29, 31], considers 5 splits obtained by selecting 5534 images from ImageNet and 4366 images from Caltech256 across all 40 categories. The second setting, adopted in [27], uses 3847 images for Caltech256 and 4000 images for ImageNet.

Networks and Training. We apply the proposed method to two state of the art CNNs, *i.e.* AlexNet [18] and Inception-BN [16]. We train our networks using mini-batch stochastic gradient descent with momentum, as implemented in the Caffe library, using the following meta-parameters: weight decay 5×10^{-4}, momentum 0.9, initial learning rate 10^{-3}. We augment the input data by scaling all images to 256×256 pxls, randomly cropping 227×227 pxls (for AlexNet) or 224×224 pxls (Inception-BN) patches and performing random flips. In all experiments we choose the parameter λ by cross-validation on the source set according to the protocol in [23].

AlexNet [18] is a well-know architecture with five convolutional and three fully-connected layers, denoted as `fc6`, `fc7` and `fc8`. The outputs of `fc6` and `fc7` are commonly used in the domain-adaptation literature as pre-trained feature representations [7, 27] for traditional machine learning approaches. In our experiments we modify AlexNet by appending a DA-layer to each fully-connected layer. Differently from the original AlexNet, we *do not* perform dropout on the outputs of `fc6` and `fc7`. We initialize the network parameters from a publicly-available model trained on the ILSVRC-2012 data, we freeze all the convolutional layers, and increase the learning rate of `fc8` by a factor of 10. During training, each mini-batch contains a number of source and target samples proportional to the size of the corresponding dataset, while the batch size remains fixed at 256. We train for a total of 60 epochs (where "epoch" refers to a complete pass over the source set), reducing the learning rate by a factor 10 after 54 epochs.

Inception-BN [16] is a very deep architecture obtained by concatenating "inception" blocks. Each block is composed of several parallel convolutions with batch normalization and pooling layers. To apply the proposed method to Inception-BN, we replace each batch-normalization layer with a DA-layer. Similarly to AlexNet, we initialize the network's parameters from a publicly-available model trained on the ILSVRC-2012 data and freeze the first three inception blocks. The α parameter is also fixed to a value of 0.5 in the DA-layers of the first three blocks, which is equivalent to preserving the original batch normalization layers. Due to GPU memory constraints, we use a much smaller batch size than for AlexNet and fix the number of source and target samples in each batch to, respectively, 32 and 16. In the Office-31 experiments we train for 1200 iterations, reducing the learning rate by a factor 10 after 1000 iterations, while in the Cross-Dataset Testbed experiments we train for 2000 iterations, reducing the learning rate after 1500.

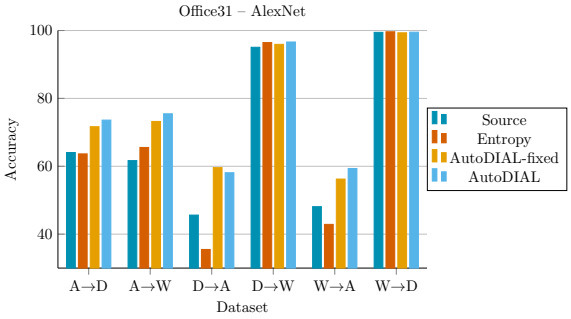

Figure 2. Accuracy on the Office31 dataset when considering different architectures based on AlexNet.

4.2. Analysis of the proposed method

We conduct an in-depth analysis of the proposed approach, evaluating the impact of our three main contributions: i) aligning features by matching source and target distributions to a reference one; ii) learning the adaptation coefficients α; iii) applying an entropy-based regularization term. As a first set of experiments, we perform an ablation study on the Office31 dataset and report the results in Fig. 2. Here, we compare the performance of four variations of the AlexNet network: trained on source data only (Source); with the addition of the entropy loss (Entropy); with DA-layers and α fixed to 1 (AutoDIAL-fixed); with DA-layers and learned α (AutoDIAL). Here the advantage of learning α is evident, as AutoDIAL outperforms AutoDIAL-fixed in all but one of the experimental settings. Interestingly, the addition of the entropy term by itself seems to have mixed effects on the final accuracy: in D→A and W→A in particular, the performance drastically decreases in Entropy compared to Source. This is not surprising as these two settings correspond to cases where the number of labeled source samples is very limited and the domain shift is more severe. However, using DA-layers in conjunction with the entropy loss always leads to a sizable performance increase. These results confirms the validity of our contribution: the entropy regularization term is especially beneficial when source and target data representations are aligned.

In Fig. 3 we plot the values of α learned by the DA-layers in AutoDIAL – AlexNet and AutoDIAL – Inception-BN on the Office31 dataset. In both networks we observe that lower layers tend to learn values closer to 1, *i.e.* require an higher degree of adaptation compared to the layers

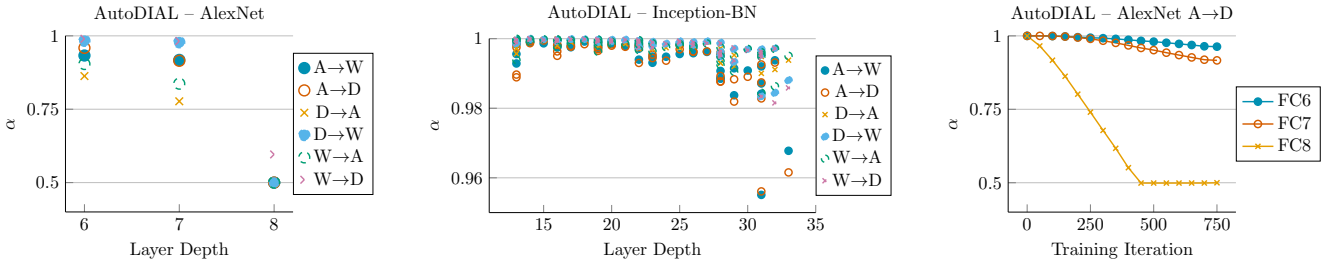

Figure 3. α parameters learned on the Office31 dataset, plotted as a function of layer depth (left and center) and training iteration (right).

closer to the classifier. This behavior, however, seems to be more pronounced in AutoDIAL – AlexNet compared to AutoDIAL – Inception-BN. Our results agree with recent findings in the literature [1], according to which lower layers in a network are subject to domain shift even more than the very last layers. During training, the α are able to converge to their final values in a few iterations (Fig. 3, right).

4.3. Comparison with State of the Art methods

In this section we compare our approach with state-of-the art deep domain adaptation methods. We first consider the Office-31 dataset. The results of our evaluation, obtained embedding the proposed DA-layers in the AlexNet and the Inception-BN networks as explained in Section 4.1, are summarized in Tables 1 and 2, respectively. As baselines, we consider: Deep Adaptation Networks (DAN) [21], Deep Domain Confusion (DDC) [32], the ReverseGrad network [9], Residual Transfer Network (RTN) [23], Joint Adaptation Network (JAN) [22], Deep Reconstruction-Classification Network (DRCN) [10] and AdaBN [19] with and without CORAL feature alignment [27]. The results associated to the baseline methods are derived from the original papers. As a reference, we further report the results obtained considering standard AlexNet and Inception-BN networks trained only on source data.

Among the deep methods based on the AlexNet architecture, AutoDIAL – AlexNet shows the best average performance, clearly demonstrating the benefit of the proposed adaptation strategy. Similar results are found in the experiments with Inception-BN network, where our approach also outperforms all baselines. It is interesting to compare AutoDIAL with the AdaBN method [19], as this approach also develops from a similar intuition than ours. Our results clearly demonstrate the added value of our contributions: the introduction of the alignment parameters α, together with the adoption of the entropy regularization term, produce a significant boost in performance.

In our second set of experiments we analyze the performance of several approaches on the Office-Caltech dataset. The results are reported in Table 3. We restrict our attention to methods based on deep architectures and, for a fair comparison, we consider all AlexNet-based approaches. Here we report results obtained with DDC [32], DAN [21], and the recent Residual Transfer Network (RTN) in [23]. As it is clear from the table, our method and RTN have roughly the same performance (90.6% vs 90.4% on average), while they significantly outperform the other baselines.

Finally, we perform some experiments on the Caltech-ImageNet subset of the Cross-Dataset Testbed of [30]. As explained above, to facilitate comparison with previous works which have also considered this dataset we perform experiments in two different settings. As baselines we consider Geodesic Flow Kernel (GFK) [12], Subspace Alignment (SA) [8]), CORAL [27], Transfer Component Analysis (TCA) [25], Simultaneous Deep Transfer (SDT) [31], and the recent method in [29]. Table 4 and Table 5 show our results. The proposed approach significantly outperforms previous methods and sets the new state of the art on this dataset. The higher performance of our method is not only due to the use of Inception-BN but also due to the effectiveness of our contributions. Indeed, the proposed alignment strategy, combined with the adoption of the entropy regularization term, makes our approach more effective than previous adaptation techniques based on Inception-BN, *i.e.* AdaBN [19].

5. Conclusions

We presented AutoDIAL, a novel framework for unsupervised, deep domain adaptation. The core of our contribution is the introduction of novel Domain Alignment layers, which reduce the domain shift by matching source and target distributions to a reference one. Our DA-layers are endowed with a set of alignment parameters, also learned by the network, which allow the CNN not only to align the source and target feature representations but also to automatically decide at each layer the required degree of adaptation. Our framework exploits target data both by computing statistics in the DA-layers and by introducing an entropy loss which promotes classification models with high confidence on unlabeled samples. The results of our experiments demonstrate that our approach outperforms state of the art domain adaptation methods.

While this paper focuses on the challenging problem of unsupervised domain-adaptation, our approach can be also exploited in a semi-supervised setting. Future works will be devoted to analyze the effectiveness of AutoDIAL in

Method (Source / Target)	Amazon DSLR	Amazon Webcam	DSLR Amazon	DSLR Webcam	Webcam Amazon	Webcam DSLR	Average
AlexNet – source [18]	63.8	61.6	51.1	95.4	49.8	99.0	70.1
DDC [32]	64.4	61.8	52.1	95.0	52.2	98.5	70.6
DAN [21]	67.0	68.5	54.0	96.0	53.1	99.0	72.9
ReverseGrad [9]	67.1	72.6	54.5	96.4	52.7	99.2	72.7
DRCN [10]	66.8	68.7	56.0	96.4	54.9	99.0	73.6
RTN [23]	71.0	73.3	50.5	**96.8**	51.0	**99.6**	73.7
JAN [22]	71.8	74.9	**58.3**	96.6	55.0	99.5	76.0
AutoDIAL – AlexNet	**73.6**	**75.5**	58.1	96.6	**59.4**	99.5	**77.1**

Table 1. AlexNet-based approaches on Office31 / full sampling protocol.

Method (Source / Target)	Amazon DSLR	Amazon Webcam	DSLR Amazon	DSLR Webcam	Webcam Amazon	Webcam DSLR	Average
Inception-BN – source [16]	70.5	70.3	60.1	94.3	57.9	**100.0**	75.5
AdaBN [19]	73.1	74.2	59.8	95.7	57.4	99.8	76.7
AdaBN + CORAL [19]	72.7	75.4	59.0	96.2	60.5	99.6	77.2
DDC [32]	73.2	72.5	61.6	95.5	61.6	98.1	77.1
DAN [21]	74.4	76.0	61.5	95.9	60.3	98.6	77.8
JAN [22]	77.5	78.1	**68.4**	96.4	**65.0**	99.3	80.8
AutoDIAL – Inception-BN	**82.3**	**84.2**	64.6	**97.9**	64.2	99.9	**82.2**

Table 2. Inception-based approaches on Office31 / full sampling protocol.

Method (Source / Target)	Amazon Caltech	Webcam Caltech	DSLR Caltech	Caltech Amazon	Caltech Webcam	Caltech DSLR	Average
AlexNet – source [18]	83.8	76.1	80.8	91.1	83.1	89.0	84.0
DDC [32]	85.0	78.0	81.1	91.9	85.4	88.8	85.0
DAN [21]	85.1	84.3	82.4	92.0	90.6	90.5	87.5
RTN [23]	**88.1**	86.6	84.6	93.7	**96.9**	**94.2**	**90.6**
AutoDIAL – AlexNet	87.4	**86.8**	**86.9**	**94.3**	96.3	90.1	90.3

Table 3. Results on the Office-Caltech dataset using the full protocol.

Method (Source / Target)	Caltech Imagenet	Imagenet Caltech
SDT [31]	–	73.6
Tommasi *et al.* [29]	–	75.4
Inception-BN – source [16]	82.1	88.4
AdaBN [19]	82.2	87.3
AutoDIAL – Inception-BN	**85.2**	**90.5**

Table 4. Results on the Cross-Dataset Testbed using the experimental setup in [30].

Method (Source / Target)	Caltech Imagenet	Imagenet Caltech
SA [8]	43.7	52.0
GFK [12]	52.0	58.5
TCA [25]	48.6	54.0
CORAL [27]	66.2	74.7
Inception-BN – source [16]	82.1	88.4
AdaBN [19]	81.9	86.5
AutoDIAL – Inception-BN	**84.2**	**89.8**

Table 5. Results on the Cross-Dataset Testbed using the experimental setup in [27].

this scenario. Additionally, we plan to extend the proposed framework to handle multiple source domains.

6. Acknowledgements

This work was partially founded by: project CHIST-ERA ALOOF, project ERC #637076 RoboExNovo (F.M.C., B. C.), and project DIGIMAP, funded under grant #860375 by the Austrian Research Promotion Agency.

References

[1] R. Aljundi and T. Tuytelaars. Lightweight unsupervised domain adaptation by convolutional filter reconstruction. In *ECCV TASK-CV Workshops*, 2016.

[2] S. Ben-David, T. Lu, T. Luu, and D. Pl. Impossibility theorems for domain adaptation. In *AISTATS*, 2010.

[3] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. Domain separation networks. In *NIPS*, 2016.

[4] F. M. Carlucci, L. Porzi, B. Caputo, E. Ricci, and S. R. Bulò. Just dial: Domain alignment layers for unsupervised domain adaptation. In *ICIAP*, 2017.

[5] W.-S. Chu, F. De la Torre, and J. F. Cohn. Selective transfer machine for personalized facial action unit detection. In *CVPR*, 2013.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[7] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014.

[8] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *ICCV*, 2013.

[9] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. *ICML*, 2015.

[10] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *ECCV*, 2016.

[11] B. Gong, K. Grauman, and F. Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *ICML*, 2013.

[12] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012.

[13] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. In *NIPS*, 2004.

[14] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. 2007.

[15] J. Huang, A. Gretton, K. M. Borgwardt, B. Schölkopf, and A. J. Smola. Correcting sample selection bias by unlabeled data. In *NIPS*, 2006.

[16] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[17] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[19] Y. Li, N. Wang, J. Shi, J. Liu, and X. Hou. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016.

[20] M. Long, G. Ding, J. Wang, J. Sun, Y. Guo, and P. S. Yu. Transfer sparse coding for robust image representation. In *CVPR*, 2013.

[21] M. Long and J. Wang. Learning transferable features with deep adaptation networks. In *ICML*, 2015.

[22] M. Long, J. Wang, and M. I. Jordan. Deep transfer learning with joint adaptation networks. *arXiv preprint arXiv:1605.06636*, 2016.

[23] M. Long, J. Wang, and M. I. Jordan. Unsupervised domain adaptation with residual transfer networks. *NIPS*, 2016.

[24] T. J. O'Neill. Normal discrimination with unclassified observations. *Jounral of the American Statistical Association*, 73(364):821–826, 1978.

[25] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2011.

[26] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *ECCV*, 2010.

[27] B. Sun, J. Feng, and K. Saenko. Return of frustratingly easy domain adaptation. In *AAAI*, 2016.

[28] B. Sun and K. Saenko. Deep coral: Correlation alignment for deep domain adaptation. *arXiv preprint arXiv:1607.01719*, 2016.

[29] T. Tommasi, M. Lanzi, P. Russo, and B. Caputo. Learning the roots of visual domain shift. *arXiv preprint arXiv:1607.06144*, 2016.

[30] T. Tommasi and T. Tuytelaars. A testbed for cross-dataset analysis. In *ECCV*, 2014.

[31] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko. Simultaneous deep transfer across domains and tasks. In *ICCV*, 2015.

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

[33] M. Yamada, L. Sigal, and M. Raptis. No bias left behind: Covariate shift adaptation for discriminative 3d pose estimation. In *ECCV*, 2012.

[34] X. Zeng, W. Ouyang, M. Wang, and X. Wang. Deep learning of scene-specific classifier for pedestrian detection. In *ECCV*, 2014.