Supervised Domain Adaptation via Saliency Augmentation

Atharv Singh Patlan
IIT Kanpur
atharvsp@iitk.ac.in

Koteswar Rao Jerripothula
IIIT Delhi
koteswar@iiitd.ac.in

Abstract

This paper aims to solve the supervised domain adaptation problem of image classifiers via saliency augmentation. The main idea here is to use domain-independent saliency extraction to enrich the source and target domains in this task. Such enrichment brings the two domains close enough, so much so that it becomes feasible to perform supervised domain adaptation by simply aligning their low-order statistics. We propose a novel supervised alignment method for aligning these statistics. Our experimental results obtained on several benchmark datasets (incl. recent Office-Home) prove our method's effectiveness and show its ability to give accurate predictions on the target dataset, even after being trained on significantly fewer labeled data from that domain than the source domain.

1. Introduction

Deep neural networks (DNNs) are nowadays being leveraged successfully in numerous computer vision tasks. However, they require a large amount of labeled training data to get to decent performance levels, provided the testing data (target domain) is drawn from a similar distribution as the data on which such DNNs have been trained (source domain). Their performance drastically reduces when the data is from a different domain because the differences between the domains' distributions may not be negligible. This phenomenon is well-known as Domain Shift [33]. As it may not always be feasible to annotate a large number of images to work with the new target domain, it is very attractive for the target task to exploit any existing fully-labeled source dataset and adapt the trained model to the target domain [4, 16, 19, 38].

The process aiming to alleviate this challenge caused by domain shift is commonly referred to as transfer learning. The main idea in transfer learning is to leverage features learnt and knowledge extracted from one or more source domains to improve the performance on problems defined in a related target domain [26, 34, 40]. In the image classification task, we may want to utilize the large number of labeled training samples in the ImageNet dataset to improve the performance on another image classification task in a very different domain. Donahue *et al*. popularized the idea of repurposing networks trained on ImageNet to be used as

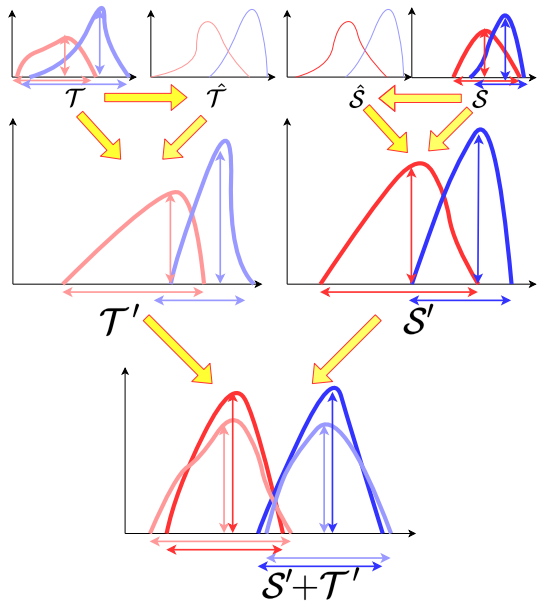

Figure 1. Working of our method. Here, $\mathcal{S}$ and $\mathcal{T}$ represent the samples from the source and target domains while $\hat{\mathcal{S}}, \hat{\mathcal{T}}, \mathcal{S}', \mathcal{T}'$ represent saliency masked Source domain, saliency masked Target domain, $\mathcal{S} \cup \hat{\mathcal{S}}, \mathcal{T} \cup \hat{\mathcal{T}}$, respectively. Plots with shades of the same color represent samples belonging to the same class, with a lighter shade representing the target samples and darker being the source samples of the same class. The colored vertical and horizontal arrowed lines represent the mean and the spread (an alternative to showing variance) of the same colored distribution. As can be seen, the distributions in $\hat{\mathcal{S}}, \hat{\mathcal{T}}$ are already very close to each other due to the domain invariant nature of saliency maps, and mixing them with the original source and target domains leads to more aligned distributions. Clearly, the separation between the means of both classes in both $\mathcal{S}'$ and $\mathcal{T}'$ has increased in the final results, and the means of the same class samples in $\mathcal{S}'$ and $\mathcal{T}'$ have come closer. Also, the spread (and hence, variance) of all the distributions has aligned close to a common value.

generic feature extractors [1]. However, usually, the classes in which the new dataset has to be classified are smaller in number and very different from the classes in ImageNet, but due to the large number of images in ImageNet, removing the existing last classification layer and retraining a freshly initialized one gives decent enough results.

Domain Adaptation is a particular transfer learning method which leverages the fact that a fair amount of labeled data belonging to a source domain (although not as many as ImageNet for example) are available, along with a small number of labeled images in the target domain, and it is known that the source and target domains both have the same class labels and images corresponding to the same objects. Domain shift is present here too, though, as the images are sampled from different distributions (for example, they might have different resolutions, background settings, pose). For Domain Adaptation, the sanitization of final layers is not required, and common features from the source domain can be effectively used in order to learn those features in the target domain [14, 22].

Our method presents a latent-space transformation approach to domain adaptation, where we create a joint-latent embedding space that is invariant to domain-shift, using divergence based methods. We also focus on the tougher problem where $N^t << N^s$, or few-shot supervised domain adaptation. This imposes a constraint that only a few labeled-target samples are available.

We aim to solve this problem by aligning each class's distributions, specifically the first and second-order statistics, in the source and target domain, and hence creating a joint latent embedding space, as can be seen in Figure 1. Before we do that, we enrich the two domains by saliency augmentation. By saliency augmentation, as it will be revealed later, we mean having augmented images by masking the images with their respective saliency maps, which essentially aim at segmenting the most visually attractive features in an image.

We employ the domain invariant nature of saliency detection methods in order to get suitable modifications to the input which facilitates the alignment of these low order statistics.

Extensive experimental results in different scenarios and different backbone networks indicate that our algorithm is robust and outperforms the state-of-the-art algorithms with only a few labeled data samples when tested using our method.

The key contributions in this paper include the following:

1. Our novel saliency augmentation idea to bring source and target domains closer for supervised domain adaptation.

2. Our novel supervised alignment method of low order statistics to create a domain-agnostic latent-space.

3. State-of-the-art results on several benchmark datasets and experiments on the recent Office-Home dataset.

The rest of this article is organized as follows: section 2 surveys related literature in the field of domain adaptation split up into relevant sub-categories, section 3 dives deep into the proposed idea and provides a theoretical exposition, section 4 presents validation of this idea through experimental evidence and section 5 provides concluding remarks.

2. Related Works

In this section, we briefly describe recent related works in the fields of salient object detection and domain adaptation, subdivided into major categories [8] by techniques used to implement domain adaptation methods.

2.1. Salient Object Detection

Salient Object Detection (SOD) aims at segmenting the most visually attractive objects in an image. It is widely used in many fields, such as visual tracking and image segmentation. There are majorly two categories of Salient Object Detection methods, which are the traditional heuristic mechanisms and deep learning mechanisms. The heuristic methods mainly use intuitive priors like color contrast, boundary background and center prior, which are usually hand crafted. Examples of there include [23]. The deep learning based methods for salient object detection usually involve mechanisms such as attention. Examples of which include Pyramid Feature Attention Network[42] and U2Net [27].

2.2. Adversarial Based Domain Adaptation

This approach generates synthetic data using the labeled samples from the source domain. Generating synthetic samples similar to the source data has become very convenient since the advent of Generative Adversarial Networks (Goodfellow *et al*. [9]).

CoGAN [20] uses two generator/discriminator pairs for both source and target distributions, with some weights of the generators and discriminators being shared to learn a domain invariant feature space. ADDA [36], on the other hand, learns separate feature extraction networks for source and target domains and trains the target CNN so that a domain classifier cannot distinguish the embeddings produced by the source or target CNNs. DANN [2] uses a gradient reversal layer along with a domain confusion loss, which is similar to the discriminator of a GAN and does not require the use of a generator. IDDA [15] also uses gradient reversal and considers all the source label information and encourages target samples to be misclassified into one of the source class. It helps the target sample to preserve its multiple modes.

2.3. Reconstruction Based Domain Adaptation

This approach uses an auxiliary reconstruction task to create a shared representation for each of the domains.

DRCN [3] tries to classify the source data and perform reconstruction of the unlabeled target data and ensures that the network learns to discriminate correctly and preserve information about the target data. It tries to learn a common representation for both source and target data. GTA [30] does not use a GAN merely for data augmentation, but uses it to obtain rich gradient information that makes the learned embeddings domain adaptive. Conditional Adversarial Networks [10] translate images from one domain to another by conditioning the discriminator and the output of the generator on the input, which is achieved by using a simple encoder-decoder architecture.

2.4. Divergence Based Domain Adaptation

These work by minimizing some divergence metric between the source and target domains to achieve a domain invariant feature representation, usually in a common latent space. These methods are subdivided into two types:

2.4.1 Unsupervised Divergence Based Methods

These methods majorly used labeled source data and unlabeled target data, and the target data is available in large amounts (comparable in quantity to the source data).

MMD and CORAL are two divergence based methods used by [21] and [32] respectively, which are used in the unsupervised regime. Another method, CAN [12] tries to minimize inter-class and maximize intra-class Contrastive Domain Discrepancy in a completely unsupervised fashion.

2.5. Semi-Supervised Divergence Based Methods

d-SNE [41] also provides a semi-supervised extension to its supervised framework, to take advantage of the large amounts of unlabeled target data. SS-MMD [6] uses a combination of MMD as an unsupervised loss and a supervised pre - initialization of the network. Our method also tackles the problem of Domain Adaptation using a supervised divergence based method.

2.5.1 Supervised Divergence Based Methods

In these methods, the source and target data are both labeled, but the quantity of target images used is much less than the source images, which make it a non-trivial task.

d-SNE [41] tries to use the reduction method of Stochastic Neighbor Embedding (SNE) to derive a loss function that minimizes the distances in the embedding space between the same labeled source and target samples and maximizes the distance between different labels samples. DAGE-LDA [7] uses graph embeddings to model loss functions encoding pair-wise relationships between source and target domain data.

As stated by [41], comparison of totally supervised Domain Adaptation methods with unsupervised and semi-supervised methods is unfair, as they have their own advantages and disadvantages. Unsupervised methods, even though they have no labels, utilize very large amounts of target data. In this way, it has an advantage of properly learning distributions of the target domain. However, supervised methods, inspite of having access to labels has access to only a very small amount of target data, making it very difficult to properly learn the exact target data distribution. Owing to these reasons, we have avoided comparison of experimental results between supervised, unsupervised and semi supervised DA methods.

3. Proposed method

Our proposed method's main aim is to align the distributions corresponding to each class in both the source and target domain with each other. We do this by using methods that aim to take the first and second-order statistics of each class in the target domain and the source domain as close as possible to each other. At the same time, we also try to make the distributions of different classes as distinct from each other as possible. We further use saliency masked images as input to our network to introduce domain invariance in the inputs and hence better alignment of the lower order statistics.

3.1. Saliency Augmentation

We observed that the most common problem that Domain Adaptation methods are made to solve is to identify objects belonging to the same class from different distributions. The images of the objects in both the distributions can have different pose, backgrounds, foreground colors etc, however, it is certain that the main object in both the images will be visually similar in both the domains, and will be the object of attention in the image. All the other features such as background etc add additional noise to the distributions of the two domains, as they do not play a role in determining the class to which an object belongs. A good way to ensure this would be to make the images have a common background not containing any objects or features, with just the object to be detected colored in the foreground.

Having these goals, we decided to employ the power of saliency detection algorithms to the problem of domain adaptation, which can accurately extract the details of the foreground in images from the background and provide a binary mask showing a dark region which is the background and a light region, which is the foreground. This saliency map is then masked over all the channels of the input image to get the desired masked image.

Figure 2. Image and corresponding saliency maps produced by Pyramid Feature Attention Network[42] and [23] for a monitor in Amazon, DSLR and Webcam domains respectively. As can be seen, there is a clear loss of certain features in the deep method owing to variations in the contents of images depending on the domain (biases inadvertently introduced while training the deep method), while no such flaws are seen in the heuristic method.

Hence in this way, we define function $h(x) = \text{Mask}(x, \text{Sal}(x))$, which represents the process of channel-wise masking an image x with its saliency map retrieved using the $\text{Sal}(\cdot)$ function.

An important requirement for our method to work is to find a suitable $\text{Sal}(\cdot)$ function. There are several powerful methods currently being employed for the problem of saliency detection as highlighted in 2.1. We employ a heuristic method proposed by [23] to perform this task. An alternative for this heuristic mechanism could have been one of the popular deep learning methods like Pyramid Feature Attention Network[42] and U2Net [27]. However, they appeared to us to be unsuitable for this task, the reason for which have been examined in section 4.

First, a Gaussian filter with a 3x3 window is used twice, in order to smooth the image be robust to noise. Then, the system calculates on-center and off-center differences separately using a unique integral image with variable size filter windows over the original grayscale image, with the help of an on-center intensity map and subsequently calculated intensity submaps.

Being a heuristic method, the algorithm gets applied on any domain and any image in the same way, resulting in a **domain invariant** saliency map and masked image.

Hence, we use this heuristic method as our $\text{Sal}(\cdot)$ function in order to construct saliency maps and hence the masked input images for both the source data and target training data.

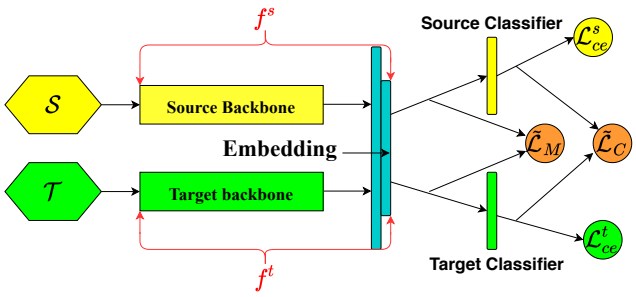

Figure 3. Proposed network architecture. Here, $\mathcal{S}$ and $\mathcal{T}$ represent the samples from the source and target domains respectively, while f^s and f^t represent the feature extractors for $\mathcal{S}$ and $\mathcal{T}$ respectively, consisting separate backbone architectures, with two shared layers at the end, with the latter being the embedding layer, the outputs of which are used to calculate $\tilde{\mathcal{L}}_M$ and also go to the Source and Target classifier layers, whose activations are used to calculate the $\tilde{\mathcal{L}}_C$, and the cross entropy losses $\mathcal{L}^s_{\text{ce}}$ for the source domain and $\mathcal{L}^t_{\text{ce}}$ for the target domain.

As in the end, the final images to be evaluated will be original, unmasked images, it is important for the method to be able to associate the masked and unmasked images together. Thus, the data being input into the method should contain the original source images and target train images along with the masked images in order to make the method learn properties from the original images as well and associate them with the masked images.

Hence, in this way, we get our final domains $\mathcal{S}, \mathcal{T}$ on which we apply domain adaptation. We define the Augmented source domain $\mathcal{S} = \mathcal{S}' \cup \hat{\mathcal{S}}$ and the augmented target domain $\mathcal{T} = \mathcal{T}' \cup \hat{\mathcal{T}}$ where $\mathcal{S}', \mathcal{T}'$ represent the original source and target domains and $\hat{\mathcal{S}} = \{h(x) | x \in \mathcal{S}'\}$, $\hat{\mathcal{T}} = \{h(x) | x \in \mathcal{T}'\}$ represent the source and target domains obtained as a result of masking generated saliency maps using $\text{Sal}(\cdot)$ on the original images.

Figures of masked images using the heuristic method our present in our supplementary material.

3.2. Network Architecture

The usage of saliency maps allows us to remove the unnecessary background noise from the domains. However, there is still some distinction in the objects present in the foreground, and in order to align the two domains, we need to used deep learning based methods to perform domain adaptation.

Our feature extractors are two independent neural networks f^s and f^t, parameterized by weights w_s and w_t respectively, ending in two shared layers, with the latter being the embedding layers, as seen in Figure 3. Pragmatically, a single network can be shared between the two domains

($f^s = f^t$) if the samples from both the source and target domains have the same dimensions. Separate classification layers follow the embedding layer for source and target data, respectively.

3.3. Aligning First Order Statistic

The first task at hand is to align the first order statistic of the two input domains.

Let $f^s_{i,c} = f^s(x^s_i | y^s_i = c)$ represent the activations of the embedding layer of the source neural network for a source sample $x^s_i \in \mathcal{S}$, having the label $y^s_i = c$. Similarly, let $f^t_{i,c'} = f^t(x^t_i | y^t_i = c')$ represent the activations of the embedding layer of the target neural network for a target sample $x^t_i \in \mathcal{T}$ having the label $y^t_i = c'$. Here $c, c' \in \mathcal{C}$, the set of classes present in the input minibatch which is common to both domains.

A very intuitive formulation that comes to one's mind to properly align the first order statistc of distribution f^s_c of source samples belonging to class c and the distribution $f^t_{c'}$ of target samples belonging to class c' is to find the difference of the classwise means of the two domains,

$$\mathcal{D}^2_M(f^s_c, f^t_{c'}) = \left\| \sum_{i=1}^{N^s_c} \frac{f^s_{i,c}}{N^s_c} - \sum_{i=1}^{N^t_{c'}} \frac{f^t_{i,c'}}{N^t_{c'}} \right\|_2 \tag{1}$$

and modify this difference in such a way that the mean of samples belonging to a class in one domain should be as close to each other, while they should be as far as possible for samples belonging to different classes. Here N^s_c and $N^t_{c'}$ represent the total number of samples in $\mathcal{S}$ and $\mathcal{T}$ having labels c and c' respectively. Formally, equation (1) represents the norm of the difference between the means of source data samples from class c and target data samples from class c', and represents how far or near the means of the distributions of two class c, c' are.

However, proving that Equation (1) indeed aligns the first order statistics of the two distributions is non-trivial. Hence, we use the Maximum Mean Discrepancy *(MMD)* [5], as it is a popular method to model this problem in Domain Adaptation tasks, to prove that this is indeed the case. Note that the formulation of MMD is such that it can be used only in unsupervised tasks, while our task is totally supervised in nature, hence we can not employ MMD directly to our task. MMD works with the entire distribution of the data from a domain, while we can only use it for comparing the distribution of a particular class in a domain.

As the name suggests, given two sets of data, the MMD measures the distance between the mean of the two sets after mapping each sample to a Reproducing Kernel Hilbert Space (RKHS).

Then, the MMD between the class-only distributions f^s_c and $f^t_{c'}$ is :

$$\text{MMD}^2(f^s_c, f^t_{c'}) = \left\| \sum_{i=1}^{N^s_c} \frac{\phi(f^s_{i,c})}{N^s_c} - \sum_{i=1}^{N^t_{c'}} \frac{\phi(f^t_{i,c'})}{N^t_{c'}} \right\|^2_{\mathbb{R}^d} \tag{2}$$

Here, $\phi(\cdot)$ denotes the mapping to an RKHS in $\mathbb{R}^d$, where d represents the dimension of the latent space. In practice, this mapping is typically unknown. Expanding equation (2) and using the kernel trick to replace inner products by kernel values lets us rewrite the MMD^2 value as:

$$\sum_{i,i'} \frac{k(f^s_{i,c}, f^s_{i',c})}{(N^s_c)^2} - 2\sum_{i,j} \frac{k(f^s_{i,c}, f^t_{j,c'})}{N^s_c N^t_{c'}} + \sum_{j,j'} \frac{k(f^t_{j,c'}, f^s_{j',c'})}{(N^t_{c'})^2} \tag{3}$$

where the dependency on the network parameters comes via the f_i values and where $k(\cdot, \cdot)$ is a kernel function. In unsupervised domain adaptation, the standard Gaussian kernel $k(u, v) = e^{\frac{\|u-v\|}{2\sigma^2}}$ is used in order to reach the final formulation.

However, in order to prove that the intuitive mean difference value $\mathcal{D}^2_M$ obtained in equation (1) is indeed a consequence of the MMD, we make use of the **bilinear kernel** $k(u, v) = \langle u, v \rangle$, which is the scalar product of u and v, which reduces the MMD^2 value to the same values as $\mathcal{D}^2_M$ in (1):

It can be intuitively seen in equation (1) that the mean of target samples belonging to class c' should be as close as possible to the mean of the source samples belonging to class $c = c'$ to bring down the $\mathcal{D}_M$ value, and hence make their distributions closer to each other, so that they are classified as the same. Also, it should be as far as possible to the mean of source samples of any class $c \neq c'$, making the $\mathcal{D}_M$ value larger and making it easier to distinguish between the samples from two classes, as they would have a larger gap in the means of their distributions. This can be seen in Figure 1.

Mathematically, this problem can be represented by considering a situation where the value $f^t_{c'} = \sum_{i=1}^{N^t_{c'}} \frac{f^t_{i,c'}}{N^t_{c'}}$ represents a data point in itself which can be classified by comparing the $\mathcal{D}_M$ values between this data point and data points represented by $f^s_c = \sum_{i=1}^{N^s_c} \frac{f^s_{i,c}}{N^s_c} \forall c \in \mathcal{C}$.

Hence in this latent-space,

$$p_{c'} = \frac{\exp(-\mathcal{D}_M(f^s_{c'}, f^t_{c'}))}{\sum_{c \in \mathcal{C}} \exp(-\mathcal{D}_M(f^s_c, f^t_c))} \tag{4}$$

is the probability that $f^t_{c'}$ has been correctly classified to belong to class c'.

Notice that $\sum_{c \in \mathcal{C}} \exp(-\mathcal{D}_M(f^s_c, f^t_c))$ can be written as $\exp(-\mathcal{D}_M(f^s_{c'}, f^t_{c'})) + \sum_{c \in \mathcal{C}_{\not c'}} \exp(-\mathcal{D}_M(f^s_c, f^t_c))$ where $\mathcal{C}_{\not c'}$ represents the set of classes excluding c'. Hence by us-

ing this, we can write:

$$\frac{1}{p_{c'}} = 1 + \frac{\sum_{c \in \mathcal{C}_{\not{c'}}} \exp(-\mathcal{D}_M(f_c^s, f_c^t))}{\exp(-\mathcal{D}_M(f_{c'}^s, f_{c'}^t))} \tag{5}$$

Since we want to maximize the probability $p_{c'}$ of making the correct prediction of $f_{c'}^t$, we minimize the log-likelihood of $\frac{1}{p_{c'}}$, which is equivalent to minimizing the ratio of intra-class $\mathcal{D}_M$ to inter-class $\mathcal{D}_M$ in the latent space, and hence:

$$\mathcal{L}_{M_{c'}} = \log\left(1 + \frac{\sum_{c \in \mathcal{C}_{\not{c'}}} \exp(-\mathcal{D}_M(f_c^s, f_c^t)}{\exp(-\mathcal{D}_M(f_{c'}^s, f_{c'}^t))}\right). \tag{6}$$

Relaxation: Since we have sum of exponentials in the likelihood formulation, the ratio in equation (6) may have a scaling issue. This leads to adverse effects in stochastic optimization techniques such as stochastic gradient descent. Since our feature extractors f^s and f^t are neural networks, it is essential to avoid this. Therefore, we relax this likelihood by only minimizing the $\mathcal{D}_M$ between the $f_{c'}^t$ and $f_{c'}^s$ and maximize the smallest $\mathcal{D}_M$ between $f_{c'}^t$ and $f_c^s \forall c \in \mathcal{C}_{\not{c'}}$. Thus, the final loss is,

$$\tilde{\mathcal{L}}_{M_{c'}} = \mathcal{D}_M(f_{c'}^s, f_{c'}^t) - \min_{c \in \mathcal{C}_{\not{c'}}} \{a | a \in \mathcal{D}_M(f_c^s, f_{c'}^t)\} \tag{7}$$

We take the mean of the calculated value of $\tilde{\mathcal{L}}_{M_{c'}} \forall c' \in \mathcal{C}$. Therefore:

$$\tilde{\mathcal{L}}_M = \frac{1}{N} \sum_{c' \in \mathcal{C}} \tilde{\mathcal{L}}_{M_{c'}} \tag{8}$$

where N is the total number of classes common to both $\mathcal{S}$ and $\mathcal{T}$ in the minibatch. Figure 3 shows where the value of $\tilde{\mathcal{L}}_M$ is calculated.

3.4. Aligning Second Order Statistic

In order to align the second order statistic, we employ the CORAL (Correlation Alignment) Loss [32]. It is a loss mainly used in the unsupervised setting, hence it is required to modifiy it such that it can be employed in a class wise fashion. We calculate the CORAL loss using the final classification layers, as done in [32], in order to align the 2nd order statistics of the domains. This has been shown in Figure 3.

Hence, we use :

$$\tilde{\mathcal{L}}_C = \sum_{c \in \mathcal{C}} \frac{1}{4N_{tot}^2} \|\sigma_{S,c} - \sigma_{T,c}\|_F^2 \tag{9}$$

Where $\|\cdot\|_F^2$ represents the squared matrix Frobenius norm, N_{tot} represents the total number of classes common to both $\mathcal{S}$ and $\mathcal{T}$ which is the dimension of the final (classification) layer, and $\sigma_{S,c}, \sigma_{T,c}$ represent covariance matrices corresponding to source and target domain samples belonging to class c.

3.5. Combined Learning Formulation

Our method allows supervision to be transferred from the source to target data as it allows the target points to select neighbors from the source domain. Since we have labeled data from both domains, standard cross-entropy losses, represented by $\mathcal{L}_{\text{ce}}^s$ for the source domain and $\mathcal{L}_{\text{ce}}^t$ for the target domain, can be used as regularization on top of the domain adaptation losses to train the network. Our learning formulation is therefore defined by:

$$\mathcal{L}_{\text{LOSAWS}} = \underset{w_s, w_t}{\text{argmin}}\ \tilde{\mathcal{L}}_M + \tilde{\mathcal{L}}_C + \mathcal{L}_{\text{ce}}^s + \mathcal{L}_{\text{ce}}^t \tag{10}$$

Where LOSAWS Loss is the name given to our learning formulation, standing for Low Order Statistics Alignment With Saliency Loss.

When we use only the original domains (without saliency), we term the loss as LOSA (Low Order Statistics Alignment Loss).

4. Experiments and Results

In this section, we describe the experimental setup used to evaluate and compare supervised Domain Adaptation methods.

4.1. Datasets

We evaluate our method on various benchmarks to establish its performance on both standard and new tasks.

The **MNIST** [18] and **USPS** [17] datasets contain handwritten digits from 0 to 9 captured in grayscale. Using these, we perform the MNIST→USPS DA task

Office-31 [29] is a standard DA benchmark containing three domains (Amazon ($\mathcal{A}$), DSLR ($\mathcal{D}$) and Webcam ($\mathcal{W}$)), each containing images of 31 object classes found in an ordinary office.

Office-Home [37] is a challenging medium-sized benchmark, which consists of four distinct domains (Artistic images (A), Clip Art (C), Product images (P), and Real-World images (R)), each containing images of 65 everyday object classes. It has only been used on unsupervised DA tasks earlier and is a new task for supervised DA.

4.2. Experimental Setup

We employ the standard experimental setup used to evaluate the performance of Domain Adaptation methods, used by [25, 41], which is as follows: A set number of samples of each class are drawn from the source domain, and a given small number of samples per class are drawn from the target domain to be used for training.

Because very few unique samples from the target domain are used for training in each experiment, the results usually vary significantly between runs and depend a lot on the random seed used for creating the training and test splits.

Table 1. MNIST → USPS multi-class accuracy (%) for a varying number of available target samples, and 200 source samples per class. The mean is reported across ten runs. The standard-deviation has been omitted as it was small. Top rows: The results published in other works Bottom row: Our results. The best accuracy is reported in **Red** color and the second best in **Blue**.

	One-shot	Three-shot
CCSA [25]	85.0	90.1
FADA [24]	89.1	91.9
d-SNE [41]	92.9	93.6
NEM [39]	72.2	86.6
LOSA	**93.2**	**94.2**
LOSAWS	**93.1**	**94.4**

Hence, several seeds are used to create training and testing splits for target data and to get the source dataset images. The mean and standard deviation obtained for each dataset pair across all the datasets is reported.

Evaluation Metric: We report multi-class accuracy, which is obtained by computing per-class accuracy independently, and then averaging over all 31 categories, a method to report accuracy used by [7, 13, 35]. We use this as the Office-31 dataset is imbalanced. It is essential to give each class equal importance in calculating the effectiveness of the methods, which is not facilitated well by the usual classification accuracy.

4.3. Results

4.3.1 MNIST-USPS

We first try our model on the MNIST-USPS task in order to test the strength of our formulation without the use of saliency maps. For our experiments in the MNIST to USPS domain adaptation problem, we used the same network architecture as used in [25, 41].

For performing domain adaptation, we randomly sampled 200 images per class from the MNIST dataset, which was our source domain. Experiments using just 1 and 3 random target samples per class from the USPS dataset, which was our target domain, were conducted and each experiment was repeated 10 times.

We used the predefined test-train splits for the MNIST and USPS datasets, the source training set was sampled from the default train set of MNIST, the target training set was sampled from the training split of USPS, and the target test set was the entire test split of the USPS dataset.

It can be seen in Table 1 that our method performs the best among all the given methods in tasks having significantly fewer (just 1 and 3) samples from the target domain, and hence performs well in one-shot and very few-shot learning. As expected, the saliency masking does not make a significant difference in the performance on this dataset as the images are already quite salient.

We also present the t-SNE visualizations of the USPS test dataset without and with domain adaptation respectively in our supplementary material, showing the ability of our model to easily distinguish between target dataset images belonging to different classes and leads to better classification upon performing domain adaptation.

4.3.2 Office-31

We then run our experiments on the Office-31 dataset, which is a standard dataset used to compare supervised domain adaptation tasks. In our experiments on the Office31 dataset, We use the same network as described in Figure 3 using either VGG16 [31] or ResNet-101 [11] networks pretrained on ImageNet [28] as the backbone, which is customary in Domain Adaptation literature, to provide a fair comparison with the relevant methods. We first finetune the embedding layers on the source data reported as as FT-Source in Table 2, which are then used in our Domain Adaptation task.

We follow the experimental procedure described in subsection 4.2. We create the training set using 20 source samples per class for the Amazon domain, and 8 source samples per class for DSLR and Webcam. From the target domain, 3 samples per class are drawn in each case. The remaining target data is used as the test set.

This experiment is performed for all six combinations of source and target domain in $\{\mathcal{A}, \mathcal{D}, \mathcal{W}\}$, and each combination is run five times using different seeds.

As the train-test splits are the same, we directly cite the results using the multi-class accuracy metric for CCSA, d-SNE, and DAGE-LDA as reported by [7], and also report the results from [35] and [13].

The accuracy of various supervised methods, including ours, on Office-31, are reported in table 2.

As can be seen in the results of our experiments, our method performs much better than the current best methods on the Office-31 task d-SNE [41], and DAGE-LDA [7]. We compare our method with d-SNE and DAGE-LDA using the VGG-16 as the backbone network, which is also the backbone network used by them. Our method works extremely well in the difficult domain adaptation tasks of $\mathcal{D} \rightarrow \mathcal{A}$ and $\mathcal{W} \rightarrow \mathcal{A}$, which have very large domain shifts, clearly outperforming all the other methods, which shows the ability of our model to work well in the toughest scenarios with high domain shifts, even with very less samples from the target domain.

We also report our results on Office-31 using the ResNet-101 as the backbone in Table 2.

We also perform experiments using other salient object detection methods. Two of these methods rely on deep features learned by training on a large input dataset, such as

Table 2. Office-31 Multi-class accuracy (%), using 3 target training samples per class. The results are reported as the mean and standard deviation across five runs. Top rows: FT-Source baseline and the experimental results of various papers as reported in [7], using the experimental methodology in sub section 4.2. Bottom rows: Our results (LOSAWS) using VGG-16 along with ablation study (LOSA, FOSA, SOSA), and ResNet-101. We report the overall best accuracy in **Red** color and the best accuracy using VGG-16 in **Blue** color.

	$\mathcal{A} \rightarrow \mathcal{D}$	$\mathcal{A} \rightarrow \mathcal{W}$	$\mathcal{D} \rightarrow \mathcal{A}$	$\mathcal{D} \rightarrow \mathcal{W}$	$\mathcal{W} \rightarrow \mathcal{A}$	$\mathcal{W} \rightarrow \mathcal{D}$	Avg.
FT-Source	62.3 ± 0.8	61.2 ± 0.9	58.5 ± 0.8	80.1 ± 0.6	51.6 ± 0.9	95.6 ± 0.7	68.2
SDA [35]	86.1 ± 1.2	82.7 ± 0.8	66.2 ± 0.3	95.7 ± 0.5	65.0 ± 0.5	97.6 ± 0.2	82.2
So-HoT [13]	86.3 ± 0.8	84.5 ± 1.7	66.5 ± 1.0	95.5 ± 0.6	65.7 ± 1.7	97.5 ± 0.7	82.7
CCSA [25]	84.8 ± 2.1	87.5 ± 1.5	66.5 ± 1.9	97.2 ± 0.7	64.0 ± 1.6	98.6 ± 0.4	83.1
d-SNE [41]	86.5 ± 2.5	88.7 ± 1.9	65.9 ± 1.1	97.6 ± 0.7	63.9 ± 1.2	99.0 ± 0.5	83.6
DAGE-LDA [7]	85.9 ± 2.8	87.8 ± 2.3	66.2 ± 1.4	$\mathbf{97.9 \pm 0.6}$	64.2 ± 1.2	$\mathbf{99.5 \pm 0.5}$	83.6
FOSA (VGG-16)	85.3 ± 1.7	82.0 ± 1.9	64.7 ± 1.1	96.2 ± 1.7	63.9 ± 2.4	97.5 ± 2.1	81.6
SOSA (VGG-16)	87.2 ± 1.4	84.8 ± 2.7	65.3 ± 1.9	97.3 ± 1.9	64.9 ± 1.8	99.3 ± 0.6	83.1
LOSA (VGG-16)	89.7 ± 2.2	85.9 ± 0.9	67.5 ± 1.5	97.5 ± 0.9	67.2 ± 1.5	99.2 ± 0.5	84.5
LOSAWS (VGG-16)	$\mathbf{92.1 \pm 1.8}$	$\mathbf{90.9 \pm 0.7}$	$\mathbf{69.7 \pm 1.2}$	$\mathbf{97.9 \pm 1.6}$	$\mathbf{70.5 \pm 1.5}$	99.0 ± 0.7	**86.7**
LOSA (ResNet-101)	91.6 ± 1.2	90.0 ± 0.9	73.1 ± 0.8	98.0 ± 0.5	72.5 ± 1.0	99.6 ± 0.4	87.5
LOSAWS (ResNet-101)	$\mathbf{93.3 \pm 0.8}$	$\mathbf{95.5 \pm 0.9}$	$\mathbf{74.4 \pm 1.2}$	$\mathbf{99.1 \pm 0.2}$	$\mathbf{74.1 \pm 1.3}$	$\mathbf{99.5 \pm 0.4}$	**89.3**

Table 3. Multi Class Accuracy (%) on Office-Home dataset. The best accuracy is in **Red** color. $|\mathcal{T}_c|$ represents the number of target samples in the input per class.

	$\lvert\mathcal{T}_c\rvert$	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	Avg.
FT-Source	-	45.22	68.38	74.91	53.12	62.91	64.83	53.15	41.04	73.45	65.56	46.61	79.04	60.69
LOSA	3	52.70	75.63	75.12	60.42	74.35	72.82	60.96	53.93	75.12	63.98	57.01	79.36	66.78
LOSA	4	55.98	76.57	75.60	61.26	76.07	73.18	61.72	55.44	75.58	64.88	57.89	80.24	67.87
LOSA	5	56.72	78.27	77.19	63.33	78.17	74.71	62.57	56.94	76.67	66.29	59.63	81.24	69.31
LOSA	6	58.96	80.45	77.80	63.75	79.98	75.97	64.40	58.34	77.67	65.82	59.67	82.75	70.46
LOSAWS	3	56.84	76.41	76.94	61.94	76.86	75.21	63.51	56.28	77.03	67.56	61.13	81.31	69.25
LOSAWS	4	59.20	78.25	78.05	64.73	79.41	76.16	63.90	58.45	77.63	68.83	61.60	82.41	70.72
LOSAWS	5	60.53	79.87	78.73	66.27	80.70	76.90	66.36	60.25	78.94	69.09	62.54	83.92	72.01
LOSAWS	6	**61.73**	**80.97**	**79.46**	**67.71**	**81.40**	**78.07**	**67.53**	**61.43**	**79.09**	**70.28**	**64.21**	**84.12**	**73.00**

Pyramid Feature Attention Network[42] and U2Net [27], which makes them perform well on most ordinary images in order to separate the foreground from the background. However, this is not always the case. Certain domain dependence can get introduced in such methods as well, as shown in figure 2, which leads to the masked image losing some inherent properties of the original image, and poor training and results by the Deep Domain Adaptation network which we use (to be introduced subsequently). We observed this in our experiments too.

A possible reason for this worsening of results can be that as the deep salient object detection networks are pretrained on a particular dataset, they treat the images from both the domains in a way biased to the dataset on which the network was pretrained.

Ablation study: We consider two baselines for the Office-31 domain adaptation task to compare with our method. First, we train the network by aligning just the first-order statistic ($\mathcal{L}_{FOSA} = \tilde{\mathcal{L}}_M + \mathcal{L}_{\text{ce}}^s + \mathcal{L}_{\text{ce}}^t$) FOSA for the VGG-16 network. Similarly we train the network by aligning the second-order statistic ($\mathcal{L}_{SOSA} = \tilde{\mathcal{L}}_C + \mathcal{L}_{\text{ce}}^s + \mathcal{L}_{\text{ce}}^t$) which is reported as SOSA for the VGG-16 network, to compare with the LOSA and LOSAWS results reported for the VGG-16 network. As can be seen, LOSAWS performs the best out of the four and works successfully in the ablation study. The reported results of ablation study can be seen in Table 2

4.3.3 Office-Home

We also run our tests on a comparatively larger and more difficult dataset, the Office-Home dataset. We use the same network architecture as we used for the Office-31 dataset, using the ResNet-101 as the backbone architecture.

As there were no other methods to compare the performance of supervised domain adaptation methods on the Office-Home dataset, we take the number of source samples in an amount proportional to what we took for the Office-31 dataset. We used 12 source samples per class for the Art cat-

Table 4. Multi Class Accuracy (%) on Office-Home dataset using deep salient object detection methods

	$\mathcal{A} \rightarrow \mathcal{D}$	$\mathcal{A} \rightarrow \mathcal{W}$	$\mathcal{D} \rightarrow \mathcal{A}$	$\mathcal{D} \rightarrow \mathcal{W}$	$\mathcal{W} \rightarrow \mathcal{A}$	$\mathcal{W} \rightarrow \mathcal{D}$	Avg.
LOSAWS + [42]	93.35 ± 1.11	91.62 ± 1.00	73.25 ± 1.13	96.85 ± 0.44	72.96 ± 0.97	98.73 ± 0.82	87.79
LOSAWS + [27]	94.50 ± 1.73	92.77 ± 0.71	73.46 ± 1.15	97.62 ± 1.42	74.21 ± 1.58	99.10 ± 0.61	88.61

egory and 16 for all other categories, and we vary the number of target samples per class (reported as $|\mathcal{T}_c|$ from 3(used in Office-31) to 6 (owing to almost double the number of classes and several times the total number of images as compared to Office-31) to conduct experiments, and present the obtained results with and without saliency in table 4.

In this experiment also, the domination of LOSAWS (using saliency masked images) can be seen clearly over the LOSA.

We conducted experiments on this dataset to show the capability of supervised domain adaptation tasks on tougher datasets as well, which were not handled by them earlier.

Although it is not fair to compare supervised and unsupervised methods, as mentioned earlier, We also present a comparison of our method on Office-Home dataset with unsupervised methods in the supplementary material for benchmarking purposes.

5. Conclusion

Recently, there has been tremendous interest in deep learning based domain adaptation methods, which exploit vast amounts of data available belonging to different domains and try to learn a generalized feature representation. In this article, we propose a supervised domain adaptation method that effectively uses the inherent properties of input distributions to align two domains modified by our novel saliency augmentation idea. Experiments conducted on standard domain adaptation datasets along with ablation studies prove the effectiveness of our method, establishing clear state-of-the-arts in the Office-31 dataset using both VGG-16 and ResNet-101, as well as a significant improvement of performances in challenging tasks of Office-31 and MNIST-USPS, by overcoming the huge domain shifts using very few samples from the target domain. We also present experiments on the recently introduced Office-Home dataset, which is relatively larger.

References

[1] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, page I–647–I–655. JMLR.org, 2014. 2

[2] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research (JMLR)*, 17(1):2016–2030, 2016. 2

[3] Muhammad Ghifary, W. Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *Computer Vision – ECCV 2016*, pages 597–613, Cham, 2016. Springer International Publishing. 3

[4] B. Gholami, P. Sahu, O. Rudovic, K. Bousmalis, and V. Pavlovic. Unsupervised multi-target domain adaptation: An information theoretic approach. *IEEE Trans. Image Process*, 29:3993–4001, January 2020. 1

[5] Arthur Gretton, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex J. Smola. A kernel method for the two-sample-problem. In *Advances in Neural Information Processing Systems 19*, pages 513–520. MIT Press, 2007. 5

[6] Mark Hamilton. Semi-supervised translation with MMD networks. *CoRR*, abs/1810.11906, 2018. 3

[7] Lukas Hedegaard, Omar Ali Sheikh-Omar, and Alexandros Iosifidis. Supervised domain adaptation: Were we doing graph embedding all along? 2020. 3, 7, 8

[8] Branislav Holländer. *Deep Domain Adaptation In Computer Vision*, 2019. 2

[9] G. Ian, P.-A. Jean, M. Mehdi, X. Bing, W.-F. David, O. Sherjil, C. Aaron, and B. Yoshua. *Generative adversarial nets*. in *NIPS*, 2014. 2

[10] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017. 3

[11] H. Kaiming, Z. Xiangyu, R. Shaoqing, and S. Jian. *Deep residual learning for image recognition*. in *CVPR*, 2016. 7

[12] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *CVPR*, 2019. 3

[13] Piotr Koniusz, Yusuf Tas, and F. Porikli. Domain adaptation by mixture of alignments of second-or higher-order scatter tensors. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7139–7148, 2017. 7, 8

[14] B. Konstantinos, T. George, S. Nathan, K. Dilip, and E. Dumitru. *Domain separation networks*. in *NIPS*, 2016. 2

[15] Vinod Kumar Kurmi and Vinay P. Namboodiri. Looking back at labels: A class based domain adaptation technique. In *2019 International Joint Conference on Neural Networks, IJCNN 2019*, Proceedings of the International Joint Conference on Neural Networks. IEEE, 2019. 2

[16] A. Lahiri, S. C. Ragireddy, P. Biswas, and P. Mitra. Unsupervised adversarial visual level domain adaptation for learning video object detectors from images. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1807–1815, 2019. 1

[17] Yann LeCun, Bernhard E. Boser, John S. Denker, Donnie Henderson, R. E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel. Handwritten digit recognition with a

back-propagation network. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 396–404. 1990. 6

[18] Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998. 6

[19] J. Li, M. Jing, K. Lu, L. Zhu, and H. T. Shen. Locality preserving joint transfer for domain adaptation. *IEEE Trans. Image Process*, 28:6103–6115, December 2019. 1

[20] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. In *Advances in Neural Information Processing Systems 29*, pages 469–477. 2016. 2

[21] M. Long, Y. Cao, J. Wang, and M. I. Jordan. *Learning transferable features with deep adaptation networks*. in *ICML*, 2015. 3

[22] C. Minmin, W. K. Q, and B. John. *Co-training for domain adaptation*. in *NIPS*, 2011. 2

[23] Sebastian Montabone and Alvaro Soto. Human detection using a mobile platform and novel features derived from a visual saliency mechanism. *Image Vision Comput.*, page 391–402, Mar. 2010. 2, 4

[24] Saeid Motiian, Quinn Jones, Seyed Mehdi Iranmanesh, and Gianfranco Doretto. Few-shot adversarial domain adaptation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6673–6683, 2017. 7

[25] Saeid Motiian, Marco Piccirilli, Donald A Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. In *IEEE International Conference on Computer Vision*, pages 5715–5725, 2017. 6, 7, 8

[26] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010. 1

[27] Xuebin Qin, Zichen Zhang, Chenyang Huang, Masood Dehghan, Osmar Zaiane, and Martin Jagersand. U2-net: Going deeper with nested u-structure for salient object detection. volume 106, page 107404, 2020. 2, 4, 8, 9

[28] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 7

[29] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European Conference on Computer Vision*, pages 213–226, 2010. 6

[30] S. Sankaranarayanan, Y. Balaji, Carlos D. Castillo, and R. Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8503–8512, 2018. 3

[31] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2015. 7

[32] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *Computer Vision – ECCV 2016 Workshops*, pages 443–450, Cham, 2016. Springer International Publishing. 3, 6

[33] A. Torralba and A. A. Efros. *Unbiased look at dataset bias*. in *CVPR*, 2011. 1

[34] Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, pages 242–264. 2010. 1

[35] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *IEEE International Conference on Computer Vision*, pages 4068–4076, 2015. 7, 8

[36] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. *Adversarial Discriminative Domain Adaptation*. in *CVPR*, 2017. 2

[37] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017. 6

[38] S. Wang, L. Zhang, W. Zuo, and B. Zhang. Class-specific reconstruction transfer learning for visual recognition across domains. *IEEE Trans. Image Process*, 29:2424–2438, January 2020. 1

[39] Z. Wang, B. Du, and Y. Guo. Domain adaptation with neural embedding matching. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–11, 2019. 7

[40] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):9, 2016. 1

[41] Xiang Xu, Xiong Zhou, Ragav Venkatesan, Gurumurthy Swaminathan, and Orchid Majumder. d-sne: Domain adaptation using stochastic neighborhood embedding. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2497–2506, 2019. 3, 6, 7, 8

[42] T. Zhao and X. Wu. Pyramid feature attention network for saliency detection. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3080–3089, 2019. 2, 4, 8, 9