Iterative Image Translation for Unsupervised Domain Adaptation

Sachin Chhabra
schhabr6@asu.edu
Arizona State University
Tempe, AZ, USA

Baoxin Li
baoxin.li@asu.edu
Arizona State University
Tempe, AZ, USA

Hemanth Venkateswara
hemanthv@asu.edu
Arizona State University
Tempe, AZ, USA

ABSTRACT

In this paper, we propose an image-translation-based unsupervised domain adaptation approach that iteratively trains an image translation and a classification network using each other. In Phase A, a classification network is used to guide the image translation to preserve the content and generate images. In Phase B, the generated images are used to train the classification network. With each step, the classification network and generator improve each other to learn the target domain representation. Detailed analysis and the experiments are testimony of the strength of our approach.

CCS CONCEPTS

• **Computing methodologies** → **Object recognition**; *Reconstruction*; *Image representations.*

KEYWORDS

Unsupervised domain adaptation; Generative domain adaptation; Image translation; Source-like target; Ternary feature alignment

ACM Reference Format:
Sachin Chhabra, Baoxin Li, and Hemanth Venkateswara. 2021. Iterative Image Translation for Unsupervised Domain Adaptation. In *Proceedings of the 1st Workshop on Multimedia Understanding with Less Labeling (MULL '21), October 24, 2021, Virtual Event, China.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3476098.3485050

1 INTRODUCTION

Deep convolutional neural networks require large labeled image datasets like ImageNet for training. Large labeled datasets are available only for certain applications like object recognition, detection, segmentation, etc. For applications with small datasets, transfer learning is applied to fine-tune pre-trained deep neural networks. Unsupervised domain adaptation is a special case of transfer learning where a deep neural network trained on a source domain is adapted to predict the labels for images from an unlabeled target domain whose images belong to a different distribution [33].

Unsupervised domain adaptation has been researched extensively in the past few years [4, 18, 19]. Adaptation is achieved through some form of alignment of the source and target distributions before the target labels are predicted. Popular approaches reduce the discrepancy between the features of the source and target domains either adversarially [5, 32] or using a distance metric like maximum mean discrepancy [17], Wasserstein distance [14] etc. While these are feature-based alignment approaches, adaptation can also be accomplished at the pixel level using image translation where images in one domain are translated to resemble images from the other domain. But pixel-level adaptation works well only on simple problems like digit classification because the images have limited variations which are nearly all captured in the datasets. Also, digit datasets from different domains vary mostly in the background with small changes to the foreground. On the other hand, real-world object classification datasets for domain adaptation have limited variations of the objects captured in the datasets. The intra-domain variations in terms of background are diverse and the inter-domain differences between the foregrounds (objects) are large [27, 34]. For effective pixel-based domain adaptation, large datasets are required that capture all the variations of the object, or the domains need to be very close. Due to these reasons, pixel-based domain adaptation approaches are mostly limited to digits, traffic signs and segmentation datasets [2, 9, 26].

Standard approaches in image-translation based unsupervised domain adaptaion area use coupled generator-discriminator pairs

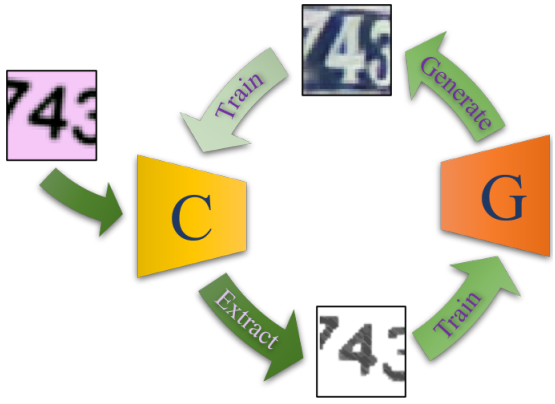

Figure 1: The learning paradigm of our approach. The classifier C extracts content from the input source image. Using the content extracted from the classifier, the image generator G is trained to generate target-like images. In next phase, the target-like generated images are used to train the classifier on target domain. The cyclic process continues until convergence.

MULL '21, October 24, 2021, Virtual Event, China

ACM ISBN 978-1-4503-8681-4/21/10…$15.00
https://doi.org/10.1145/3476098.3485050

from a GAN framework for image translation [37]. While one GAN translates source images to target, another GAN translates target images to the source. There is a cyclic loss to ensure consistency in translation [9, 23, 26]. The consistency loss however does not preclude the image from changing its category upon translation. For example, an image of digit 5 (source domain) can get converted to digit 7 in the target domain and back to digit 5 in the source domain all the while satisfying the cyclic consistency loss. The translated images are also of poor quality because the cyclic loss forces the network to embed non-relevant information like background and style into the translated image to be able to reconstruct it later.

We propose a model to overcome these problems with a single GAN framework that translates source images to the target domain. The classifier module of our model retains the content (category) information from the source image into the translated target images using a content-consistency loss. The generator in our GAN framework ensures the translated image appears to be from the target domain while preserving the content of the source image. In an iterative process, we train the classifier on the generated target images along with source labels, making it a better content extractor with each update. This way the classifier and the generator are used to train each other iteratively. Since there is no need for cyclic translation, the need to preserve the domain-specific content is eliminated while the generator can introduce target specific content resulting in superior quality in the translated images. The procedure is depicted in Fig. 1.

The contributions of our model are as follows: (1) A novel image translation framework using a classifier to guide the image generator and vice-versa, (2) a content-consistency loss to retain source information in the translated image, (3) a three-way discriminator loss to align the features of the source, target and target-like source images giving more leeway to the GAN framework in the alignment space, (4) extensive empirical and subjective analysis to demonstrate the superiority of our translation framework.

2 RELATED WORK

In this section, we discuss the related work to our approach and also compare how our approach is similar to the discussed approaches. We only present image translation-based domain adaptation approaches here. A survey of unsupervised domain adaptation using feature level alignment can be found in [33, 35] for the interested reader.

The image-to-Image translation problem has been become popular in the era of deep neural networks due to their generative capabilities. Early methods used style transfer techniques where content information from one image was presented using style information from another image. These methods modeled a style loss and content loss to train the generative models [6, 15]. One of the most popular adversarial approaches, CycleGAN, proposed to translate a source domain image to the target domain and then back to the source domain [37]. The source and the reconstructed source image have an identity loss called cycle consistency loss which is responsible for preserving the content across the image translation. However, since there is no direct constraint in the target domain, this approach tends to change the content or the category of the image when applied for domain adaptation.

To solve this problem, [2] proposed a masked pairwise mean squared error to preserve the content between the source and the translated image. This loss function requires a binary mask of the content of the image which is not easily available and works only in a few scenarios. Another approach was to add a classifier trained on the source as the pseudo-labeler for the target domain. It is then used for the semantic consistency where the generated image needs to have the same class as the original image as per the classifier [9]. A procedure to disentangle the source and target images using a shared content encoder and a pair of private domain-specific encoders was proposed in [3].

A self-learning based approach was used to train the generators in [26]. They also used an ensemble of source and target domain classifiers at test time to make predictions. Xie et al. present an object preserving image translation model using a self-supervised CycleGAN [36]. Adversarial image translation for domain adaptation has been applied for image segmentation of traffic scenes [23, 24]. Gong et al. proposed to generate intermediate domains between the source and target and gradually adapting networks to the target domain by training on the intermediate domains [8]. Liu et al. used a pair of Variational Autoencoders to map source and target domain to a common latent space [16]. This latent space was used to generate the original images as well as translate the images to other domains.

In our approach, we use a single GAN framerwork to translate from the source to the target. Our translation process is not encumbered by the need to retain the source style for translating back to the source domain. The translation is more stable and the images are of superior quality.

3 METHODOLOGY

3.1 Problem Statement

Let $S = \{x_s^i, y_s^i\}_{i=1}^{N_s}$ denote the source dataset, where N_s is number of samples drawn from the source distribution p_s and let $T = \{x_t^i\}_{i=1}^{N_t}$ be the target dataset, where N_t is number of samples drawn from the target distribution q_t. The goal of UDA is to learn the target labels $\{y_t^i\}_{i=1}^{N_t}$ using S and T. It is known that S and T share the same number of K classes but $p_s \neq q_t$ which is why a classifier trained on S is not sufficient to predict the target labels. To solve this problem, we use a classifier C, a feature-level discriminator D_f and a pair of image Generator G and discriminator D_p. We pass the Source dataset S from the generator G to generate fake target dataset $G(S) = \{G(x_s^i), y_s^i\}_{i=1}^{N_s}$.

We propose an iterative approach involving two phases, A and B, to translate a source image to a target-like image using a single GAN framework. In Phase A, we train an image generator to generate a target-like image using a source image as input while retaining only the content (category) information from the source image. In Phase B, we train a classifier to extract the content information from the input images to be used for generation. We further outline the steps of our proposed procedure below.

3.2 Content Consistency Loss

We begin with a classifier module C trained with standard cross-entropy loss on the labeled source images S,

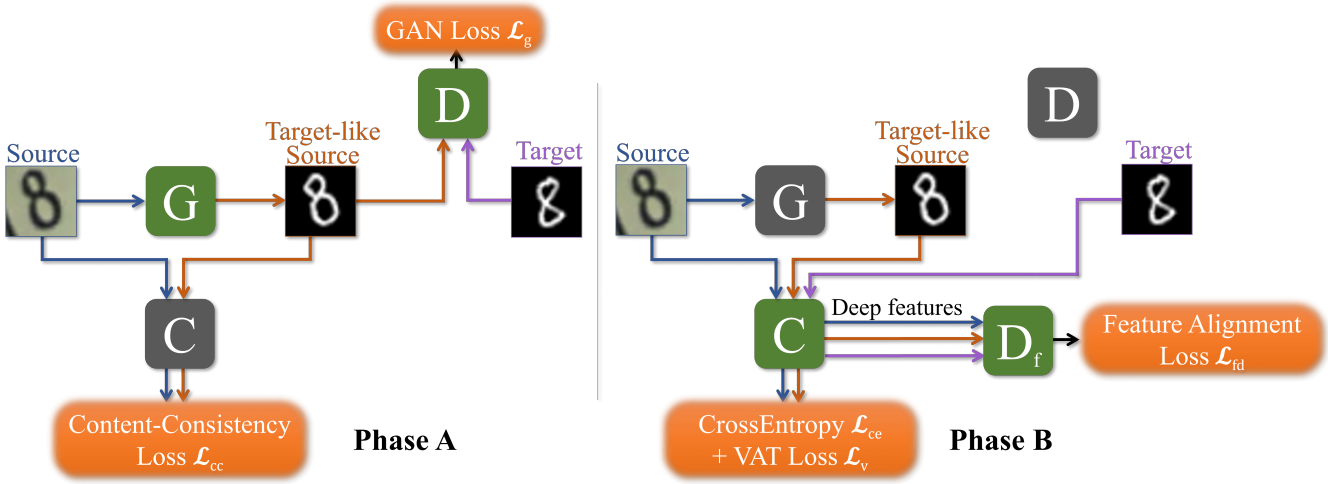

Figure 2: The proposed model. Green denotes network is trained and Gray denotes network is not trained during that phase. During Phase-A, Generator (G) is trained to minimize GAN loss $\mathcal{L}_g$ along with Content-Consistency loss $\mathcal{L}_{cc}$. During Phase-B, the source and the generated target-like source image with source labels are used to train the classifier (C) using cross-entropy loss $\mathcal{L}_{ce}$ and Virtual Adversarial Training loss $\mathcal{L}_v$. Additionally, the deep features of source, target-like source and target images are aligned using feature alignment loss $\mathcal{L}_{fd}$.

$$\mathcal{L}_{ce}^{s}(C) = -\mathbb{E}_{\{x_s,y_s\}\sim S}[y \cdot \log C(x_s)]. \quad (1)$$

We propose to retain content information from the source image using the principle of consistency regularization, which is a regularization technique from semi-supervised learning [29]. We would like the source image and its corresponding target-like transformed image to have the same content. Content in this context refers to the category of the image. Having extracted the source content from the classifier in the form of a probability distribution over the categories, $C(x)$, we train the generator $G(.)$ of our GAN to retain this content information in the generated target-like image with the content consistency loss,

$$\mathcal{L}_{cc}(G) = \mathbb{E}_{x_s\sim S}\left[||C(x_s) - C(G(x_s))||_2\right]. \quad (2)$$

Matching the probability distributions alone will not result in the identical content for the source image and its corresponding target-like image because the generated images can achieve the same probability distribution adversarially as well. To avoid this, we train the classifier using Virtual Adversarial Training [21] by minimizing the Kullback-Leibler divergence over the classifier predictions using a tiny perturbation $r < \epsilon$,

$$\mathcal{L}_{v}^{s}(C) = \mathbb{E}_{x_s\sim S}[\max_{||r||<\epsilon} D_{KL}(C(x_s)||C(x_s + r))]. \quad (3)$$

3.3 Image Generation

For training the generator to produce real-looking target-like images, we use the least squared loss [20] GAN objective along with the content consistency loss. The discriminator D is trained to distinguish between fake target $G(x_s)$ images generated by the generator G and real target images x_t. The generator G inputs a source image along with a noise vector $z \in \mathcal{N}(0, I)$ to fool the discriminator D. The input noise vector is sampled from a random Gaussian distribution and is up-scaled to the image size using a linear layer. It is then added as a new channel in the input image. The noise vector z provides the seed for the variations the generator produces in the target-like images,

$$\mathcal{L}_g(G, D) = \mathbb{E}_{x_t\sim T}[(D(x_t) - 1)^2] + \mathbb{E}_{\substack{x_s\sim S,\\ z\in\mathcal{N}(0,I)}}[(D(G(x_s, z)))^2]. \quad (4)$$

3.4 Learning Target Domain

Although the generated target images are constrained to have the same content as the source images, all of the images might not have content preserved. Therefore, select a subset of the most content-preserving generated images using content-consistency loss as a filter with a threshold τ,

$$S' := \{x_s \ni \mathcal{L}_{cc}(x_s) < \tau\}. \quad (5)$$

Initially, the classifier was trained using only the source images. The classifier can now be trained to classify target images using the generated target-like images. Although the generated images are not the actual target images, they are the best representation of the target domain learned by the generator given the constraints. These images have the same content as that of the source and are the closest representation to a labeled target domain. We exploit this fact and further train the classifier on the filtered generated images along with source labels so that it learns to classify target-like images. We train the classifier on these subsets of generated samples along with source labels using cross-entropy loss. We add VAT loss to it as well for the same reason as the source dataset,

$$\mathcal{L}_{v}^{t}(C, G) = \mathbb{E}_{\substack{x_s\sim S',\\ z\in\mathcal{N}(0,1)}}[\max_{||r||<\epsilon} D_{KL}(C(G(x_s, z))||C(G(x_s) + r))], \quad (6)$$

$$\mathcal{L}_{ce}^{t}(C, G)) = -\mathbb{E}_{\{x_s,y_s\}\sim S'}[y_s \cdot \log C(G(x_s))]. \quad (7)$$

When the classifier is trained on the generated samples, it helps the generator to produce images that are even closer to the target domain. This way, we train the generator and the classifier alternatively and both the networks can learn from each other. With each update, the generator produces better target-like images and the classifier learns more about the target from those images.

3.5 Feature Alignment

The iterative pixel-level training does not guarantee that the classifier will have good accuracy on the actual target images because the generated images may not represent the entire target distribution [9]. The pixel-level alignment alone can be achieved easily by aligning the generator to only a subset of the target domain (partial mode collapse). Also, the content variations in the target may not be entirely represented in the source images. Hence, we integrate a feature-level alignment loss to align the deep features of the domains. We consider target-like source images as a separate domain and implement domain alignment for all three domains - source, target-like source, and target,

$$\mathcal{L}_{fd}(D_f, C) = -\mathbb{E}_{x_s \sim S} \log D_f^1(C'(x_s)) - \mathbb{E}_{x_t \sim T} \log D_f^3(C'(x_t)) - \mathbb{E}_{\substack{x_s \sim S' \\ z \in \mathcal{N}(0,I)}} \log D_f^2(C'(G(x_s, z))), \quad (8)$$

where $C'(x)$ are the deep features of x using classifier C. D_f^i is a 3-class classification network and $D_f^i(x)$ is the probability of x belonging to i^{th} class. Our ternary feature alignment is similar to [30]. While training the discriminator, we use the same loss but while training the feature extractor, we maximize the loss with respect to all the domains instead of aligning them all to one domain. Eq. 8 helps in the further alignment of the distributions. The classifier/content extractor starts with knowing only about the source domain and ends up becoming an expert predictor on the target domain. The overall objective function brings together the cross-entropy loss ($\mathcal{L}_{ce}$), the virtual adversarial loss ($\mathcal{L}_v$), feature alignment ($\mathcal{L}_{fd}$), content consistency ($\mathcal{L}_{cc}$) and GAN loss ($\mathcal{L}_g$) for training the network with corresponding λ hyper-parameters controlling relative importance of the terms,

$$\min_C \max_{D_f} \lambda_{sce}\mathcal{L}_{ce}^s(C) + \lambda_{tce}\mathcal{L}_{ce}^t(C) + \lambda_{vs}\mathcal{L}_v^s(C) + \lambda_{vt}\mathcal{L}_v^t(C) - \lambda_{fd}\mathcal{L}_{fd}(D_f, C),$$

$$\min_G \max_D \lambda_{cc}\mathcal{L}_{cc}(G) + \lambda_g\mathcal{L}_g(G, D). \quad (9)$$

3.6 Final Training Procedure & Algorithm

We start with training the classifier using Eq.1 and Eq.4. Following that, the generator is trained with the image generation loss (Eq.3) and content matching loss (Eq.2). During this time, the generator is warmed up to learn to generate target-like images and match the content information. Instead of using fixed λ_{ct}, we increase it gradually using the $e^{-5(1-p)^2}$ ramp-up function from [31], where p is the ratio of the iterations completed. Lastly, the network is trained using Eq.9 iteratively. Figure 2 depicts the overall framework of our approach and the complete algorithm is in Algorithm 1.

Algorithm 1: Iterative Image Translation

Input : Source dataset $S = \{(x_s^i, y_s^i)\}_{i=1}^{N_s}$, target dataset $T = \{(x_t^i)\}_{i=1}^{N_t}$ and networks C, G, D, D_f

Output: Target labels $\{y_t^i\}_{i=1}^{N_t}$

Train C using Cross Entropy (Eq. 1) + VAT loss (Eq. 4)

Further train G and D using GAN loss (Eq. 3) + Content-Consistency loss (Eq. 2) using the ramp up function.

for *M iterations* **do**

- m_s = $miniBatch(S)$
- m_t = $miniBatch(T)$
- $m_{st} = G(m_s)$
- Train D using Eq. 9
- Train G using Eq. 9
- $m_{st} = G(m_s)$
- m'_{st} = Filter m_{st} using Eq. 5
- Train F_d using Eq. 9
- Train C using Eq. 9

end

Predict target labels y_t using C

Return $\{y_t^i\}_{i=1}^{N_t}$

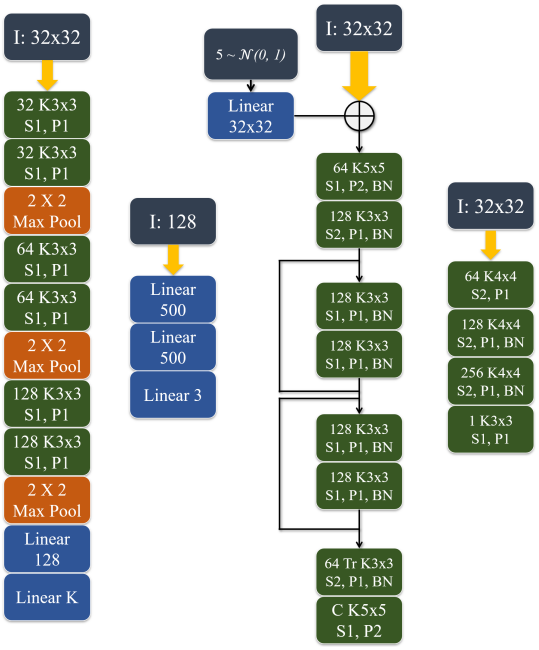

Figure 3: Network Architectures used in our experiments. From left to right: Classifier, Feature discriminator, Generator and Pixel-level discriminator. All input images are resized to 32 × 32. K 3×3 refers to 3 × 3 convolution with K feature maps and Tr K refers to transposed convolution. I, S, P, BN and C stands for Input, Stride, Pad, Batch normalization and channels respectively. All the layers use ReLU activation except Pixel-level discriminator which uses a lReLU with α=0.2.

Method	MNIST→USPS	USPS→MNIST	MNIST→MNISTM	SVHN→MNIST	MNIST→SVHN	SyDig→SVHN	SySigns→GTSRB
Source only	83.1	67.2	63.3	60.9	30.8	88.7	94.2
DANN[5]	85.1	73.0	76.7	73.9	35.7	91.1	88.7
MMD[17]	81.1	-	76.9	71.1	-	-	91.1
ADDA[32]	89.4	90.1	-	76.0	-	-	-
ATT[28]	-	-	94.2	86.2	52.8	93.1	96.2
DRCN[7]	91.8	73.7	-	82.0	40.0	-	-
PixelDA[2]	95.9	-	98.2	-	-	-	-
DSN[3]	-	-	83.2	82.7	-	91.2	93.1
I2I Adapt[23]	95.1	92.2	-	92.1	-	-	-
UNIT[16]	96.0	93.6	-	90.5	-	-	-
PLR[22]	90.7	91.8	94.3	97.3	63.4	-	-
CYCADA[9]	95.6	96.5	-	90.4	-	-	-
DupGAN[11]	96.0	98.8	-	92.5	62.7	-	-
SBADA[26]	97.6	95.0	**99.4**	76.1	61.1	-	96.7
IIT (Ours)	**97.8**	**99.1**	**99.4**	**97.4**	**66.5**	**95.4**	**97.2**

Table 1: Comparison of classification accuracy of our approach IIT (Iterative Image Translation) with different domain adaptation methods. Pixel-based approaches are below the dashed line.

4 EXPERIMENTS AND RESULTS

4.1 Dataset

We test our approach on the following tasks: MNIST ↔ USPS, MNIST → MNIST-M, SVHN ↔ MNIST, SynDigits → SVHN and SynSigns → GTSRB. MNIST is a handwritten digits dataset with a black background [13]. SVHN contains RGB digits dataset extracted from real-world house number images [25]. MNIST-M was created by combining MNIST images with patches randomly extracted from color photos of BSDS500. USPS is a digits dataset developed by recognizing the digits on the envelopes. Synthetic Digits (SynDigits) is a synthetically created digits dataset consisting of various English fonts on random backgrounds. Synthetic Signs (SynSigns) and GTSRB are traffic signs datasets where SynSigns contains images from Wikipedia whereas GTSRB has real-world traffic sign images [10]. All the digit datasets have 10 classes and present different visual variations in the domains. The traffic signs dataset provides a larger classification task of 43 classes.

4.2 Training details

We follow the standard unsupervised domain adaptation protocol i.e. train using the labeled source and unlabelled target training sets and evaluate on the target test set. All the input images are resized to 32 × 32. The classifier has a generic architecture of six 3 × 3 convolutional layers containing 32, 32, 64, 64, 128, and 128 feature maps followed by one fully-connected layer of 128 hidden units and a classification layer of size K. The Feature discriminator uses the output of the first fully-connected layer of the classifier as the input. The generator and the pixel-level discriminator architecture are inspired from [12, 26, 37]. For MNIST and USPS image generation, we used a smaller image discriminator by excluding the 256 feature map layer. The generator uses a 5-dimensional noise vector sampled from $\mathcal{N}(0, 1)$ which is then scaled up to the size of the image using a linear layer and concatenated channel-wise with the input image. The network architecture is displayed in Figure 3.

All the networks are trained using a batch size of 128 and Adam optimizer with β_1 = 0.5 and β_2 = 0.99. We use a learning rate of $1e^{-4}$ to train the classifier, $2e^{-4}$ for training the generator and discriminator (default from CycleGAN[37]). For feature discriminator, we used a lower learning rate of 10^{-5} to prioritize pixel-level adaptation over feature-level adaptation.

Figure 4: Sample generated images. Top to Bottom: USPS ↔ MNIST, MNIST ↔ MNIST-M, SyDigits ↔ SVHN, SVHN ↔ MNIST, SySigns ↔ GTSRB. We show images for each combination on every row. Upper row: Original source images and Lower row: Generated target images.

We set the hyperparameters λ_{sce}, λ_{tce}, λ_g to 1. the The hyperparameters λ_{vs}, λ_{vt}, λ_{fd}, ϵ, λ_{cc} are set to 0.1, 0.1, 0.01, 3.5 and K^2s respectively based on existing literature which has used similar loss functions. τ is the only hyperparameter tuned by use and is empirically set to 10^{-6}.

4.3 Results

The results of our experiments are in Table 1. We compare our approach with Image translation approaches like CYCADA [9], SBADA [26] as well as feature alignment approaches like DANN

CC	VAT	FA	TOT	M→U	U→M	M→MM	S→M
✗	✗	✗	✗	83.1	67.2	63.3	60.9
✗	✓	✓	✓	94.2	97.4	98.3	15.5
✓	✗	✓	✓	96.5	98.1	97.7	77.4
✓	✓	✗	✓	96.2	95.1	91.3	77.8
✓	✓	✓	✓	**97.8**	**99.1**	**99.4**	**97.4**

Table 2: Ablation study of our method on MNIST(M) → USPS(U), MNIST(M) → MNIST-M(MM) and SVHN(S) → MNIST(M). CC: Content-consistency loss; VAT: Virtual adversarial training for source and target; FA: feature-level alignment; TOT: Train on target. The first row represents source-only.

[5], MMD [17], ADDA [32] and our approach outperforms all the compared methods on all the combinations. Even for the difficult combination of MNIST → SVHN, our approach is effective and beats all the compared baselines. The generated target-like images can represent the target variations and is an effective way to learn target domain. The sample translated images can be found in Fig. 4.

5 ANALYSIS

5.1 Ablation Study

For analyzing the significance of each of the loss components, we perform an ablation study by removing them, one loss component at a time, from our approach. We use MNIST ↔ USPS, SVHN → MNIST and MNIST → MNIST-M combinations for this study. The results are presented in Table 2. We perform the following three experiments

(1) No content-consistency loss: We replace the content consistency loss with the cycle consistency loss (used for training CycleGAN) by adding another image generator [37]. Though this produces decent results for small variation combinations - MNIST ↔ USPS, MNIST → MNIST-M, but results in shuffling of labels for the high variation combinations like SVHN → MNIST.
(2) No virtual adversarial training losses: No adversarial training allows the generator to fool the consistency loss without matching the content and results in negative transfer, thereby impacting the final accuracy.
(3) No feature alignment: It is clear from the results that pixel-level alignment is not sufficient to train the classifier on target domain.

5.2 Feature Visualization

We use t-SNE plots to visualize the deep features generated by the classifier before and after the adaptation in Figure 5 and Figure 7. The deep features are spread across for source, target-like and target domains before the adaptation. After the adaptation, all three of them are aligned together to get an indistinguishable representation across domains.

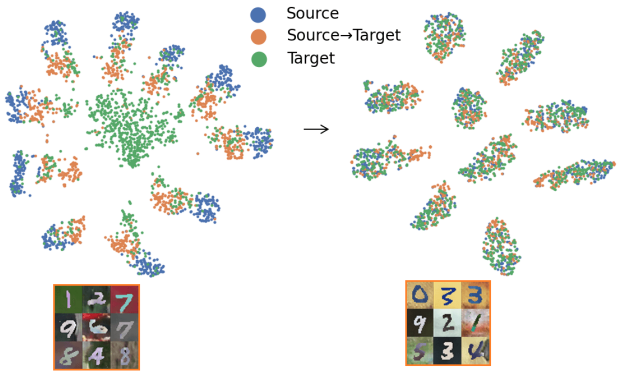

Figure 5: t-SNE plots of the deep features of source, target-like generated and target images along with generated target-like images for the MNIST → MNIST-M task. Left: depicts image features and generated target-like images before convergence. Right: depicts image features and generated target-like images after convergence.

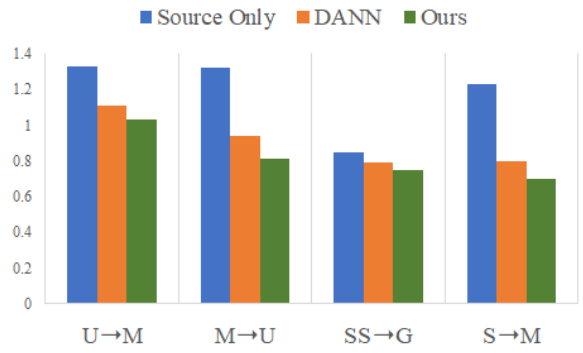

Figure 6: The domain discrepancy $\mathcal{A}$-distance between deep features for the MNIST(M) ↔ USPS(U), SySigns(SS) → GTSRB(G) and SVHN(S) → MNIST(M). Smaller is better.

5.3 Generated Images Spectrum

We use the task of MNIST→MNIST-M transfer to understand the variations in generated images. Fig.5 shows the generated images before and after the adaptation. Before adaptation, the generated images look similar to MNIST-M but they are only restricted to having a darker background and lighter foreground, resembling MNIST images. We believe it is because the classifier is trained on MNIST images only and can only detect dark to light edges. Hence, the generator needs to generate such images to keep the content loss low. The deep features of these generated images lie between those of the source (MNIST) and the target (MNIST-M) samples. They act as a bridge between the source and target domain. After adaptation, the features are aligned and the generated images cover the full spectrum of the target images.

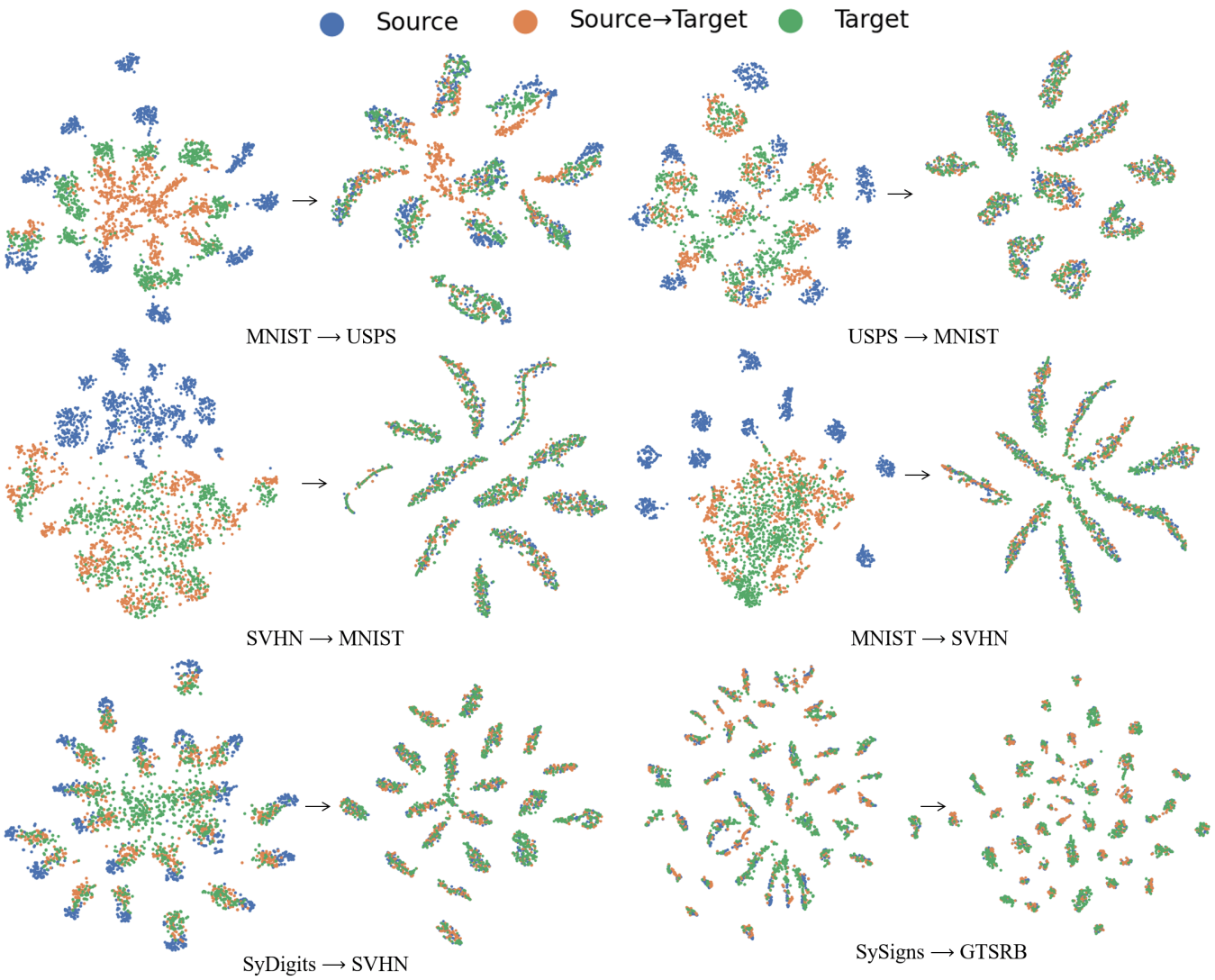

Figure 7: t-SNE plots for MNIST ⟶ USPS, USPS ⟶ MNIST, SVHN ⟶ MNIST, MNIST ⟶ SVHN, SyDigits ⟶ SVHN and SySigns ⟶ GTSRB in English reading order. For every combination, left is before adaptation and right is after adaptation. • denotes source domain, • denotes source to target-like source translated images, and • denotes Target domain samples.

5.4 Domain Gap

[1] defined the $\mathcal{A}$-distance metric for evaluating testing the domain discrepancy as $2\times(1-\varepsilon)$ where ε is the generalization error of a classifier trained to classify deep features. We use 5-fold cross-validation using a linear SVM to compute the A-distances. Fig. 6 shows the $\mathcal{A}$-distance between three domains on MNIST ↔ USPS, SySigns ⟶ GTSRB and SVHN(S) ⟶ MNIST(M) using Source only, DANN-alignment [5] and our method. Our approach brings the domains closer while achieving superior performance.

5.5 Content Extraction

We perform linear interpolation between two random input noise vectors for 3 pairs of source-target combinations and the results are in Fig. 8. For each pair, we use the same noise vectors, and the linear interpolation results in the same changes in style. For MNIST ⟶ MNIST-M combination, the color variations are similar across the two inputs. For SynDigits ⟶ SVHN, we can see 1 appearing on the left and slowly transforming the background to a box. Similarly, for USPS ⟶ MNIST, we observe equivalent angle rotations for the inputs. This confirms that our model can extract the content and uses the noise vector as the style component.

6 CONCLUSION

Pixel-based unsupervised domain adaptation approaches are fascinating as we can visualize the domain adaptation process. Most of these approaches require two sets of generators, classifiers, and

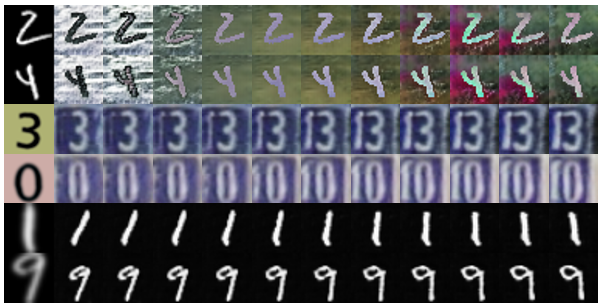

Figure 8: Linear Interpolation between two random noise vectors on MNIST → MNIST-M (Top two rows), SynDigits → SVHN (Middle two rows), and USPS → MNIST (Bottom two rows) pairs. Each pair uses the same input noise vector. The first column is the source image and other columns are generated target images. The images in the middle are generated by linear interpolating the random noise vector used for the first and the last column of the generated images.

discriminators. In this paper, we present an iterative approach that trains a classifier and image generator in tandem and keeps the size of the model to a minimum. Our approach can accurately translate a source image to a target domain while keeping the content preserved and classifies the source and target domain using a single classifier. Its three-way feature alignment ensures that the deep features are domain-invariant. The combined pixel and feature-level alignment establish a successful adaptation to the target domain. The translated images are of superior quality and our approach outperforms existing pixel and feature-level adaptation approaches for digits and traffic signs datasets.

ACKNOWLEDGMENTS

The work was supported in part by a grant from ONR. Any opinions expressed in this material are those of the authors and do not necessarily reflect the views of ONR.

REFERENCES

[1] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine learning* 79, 1-2 (2010), 151–175.
[2] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. 2017. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *CVPR*. 3722–3731.
[3] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. 2016. Domain Separation Networks. In *NeurIPS*. 343–351.
[4] Chaoqi Chen, Weiping Xie, Wenbing Huang, Yu Rong, Xinghao Ding, Yue Huang, Tingyang Xu, and Junzhou Huang. 2019. Progressive Feature Alignment for unsupervised domain adaptation. In *CVPR*. 627–636.
[5] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-Adversarial Training of Neural Networks. *JMLR* 17, 1 (2016), 2096–2030.
[6] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. 2016. Image style transfer using convolutional neural networks. In *CVPR*. 2414–2423.
[7] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. 2016. Deep reconstruction-classification networks for unsupervised domain adaptation. In *ECCV*. 597–613.
[8] Rui Gong, Wen Li, Yuhua Chen, and Luc Van Gool. 2019. DLOW: Domain Flow for adaptation and generalization. In *CVPR*. 2477–2486.
[9] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. 2018. Cycada: Cycle-Consistent Adversarial Domain Adaptation. In *ICML*. 1989–1998.
[10] Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. 2013. Detection of Traffic Signs in Real-World Images: The German Traffic Sign Detection Benchmark. In *International Joint Conference on Neural Networks*.
[11] Lanqing Hu, Meina Kan, Shiguang Shan, and Xilin Chen. 2018. Duplex generative adversarial network for unsupervised domain adaptation. In *CVPR*. 1498–1507.
[12] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *CVPR*. 1125–1134.
[13] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.
[14] Chen-Yu Lee, Tanmay Batra, Mohammad Haris Baig, and Daniel Ulbricht. 2019. Sliced Wasserstein Discrepancy for Unsupervised Domain Adaptation. In *CVPR*. 10285–10295.
[15] Yanghao Li, Naiyan Wang, Jiaying Liu, and Xiaodi Hou. 2017. Demystifying Neural Style Transfer. In *IJCAI*, Carles Sierra (Ed.). 2230–2236.
[16] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. 2017. Unsupervised image-to-image translation networks. In *NeurIPS*. 700–708.
[17] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. 2015. Learning transferable features with deep adaptation networks. In *ICML*. 97–105.
[18] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. 2018. Conditional Adversarial Domain Adaptation. In *NeurIPS*. 1640–1650.
[19] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. 2017. Deep Transfer Learning with Joint Adaptation Networks. In *ICML*. JMLR.org, 2208–2217.
[20] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. 2017. Least squares generative adversarial networks. In *ICCV*. 2794–2802.
[21] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE PAMI* 41, 8 (2018), 1979–1993.
[22] Pietro Morerio, Riccardo Volpi, Ruggero Ragonesi, and Vittorio Murino. 2020. Generative Pseudo-label Refinement for Unsupervised Domain Adaptation. In *IEEE WACV*. 3130–3139.
[23] Zak Murez, Soheil Kolouri, David Kriegman, Ravi Ramamoorthi, and Kyungnam Kim. 2018. Image to Image Translation for domain adaptation. In *CVPR*. 4500–4509.
[24] Luigi Musto and Andrea Zinelli. 2020. Semantically Adaptive Image-to-image Translation for Domain Adaptation of Semantic Segmentation. In *BMVC*.
[25] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. 2011. Reading digits in natural images with unsupervised feature learning. In *NeurIPS Workshop on Deep Learning and Unsupervised Feature Learning*.
[26] Paolo Russo, Fabio M Carlucci, Tatiana Tommasi, and Barbara Caputo. 2018. From Source to Target and Back: Symmetric Bi-directional Adaptive GAN. In *CVPR*. 8099–8108.
[27] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. 2010. Adapting visual category models to new domains. In *ECCV*. 213–226.
[28] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. 2017. Asymmetric Tri-training for unsupervised domain adaptation. In *ICML*. JMLR.org, 2988–2997.
[29] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. 2016. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. *NeurIPS* 29 (2016), 1163–1171.
[30] Yaniv Taigman, Adam Polyak, and Lior Wolf. 2016. Unsupervised cross-domain image generation. *arXiv preprint arXiv:1611.02200* (2016).
[31] Antti Tarvainen and Harri Valpola. 2017. Mean Teachers are Better Role Models: Weight-averaged Consistency Targets Improve Semi-supervised Deep Learning Results. In *NeurIPS*. 1195–1204.
[32] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. 2017. Adversarial discriminative domain adaptation. In *CVPR*. 7167–7176.
[33] Hemanth Venkateswara, Shayok Chakraborty, and Sethuraman Panchanathan. 2017. Deep-learning systems for domain adaptation in computer vision: Learning transferable feature representations. *IEEE Signal Processing Magazine* 34, 6 (2017), 117–129.
[34] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. 2017. Deep Hashing Network for Unsupervised Domain Adaptation. In *CVPR*. 5018–5027.
[35] Garrett Wilson and Diane J Cook. 2020. A survey of unsupervised deep domain adaptation. *ACM Transactions on Intelligent Systems and Technology (TIST)* 11, 5 (2020), 1–46.
[36] Xinpeng Xie, Jiawei Chen, Yuexiang Li, Linlin Shen, Kai Ma, and Yefeng Zheng. 2020. Self-Supervised CycleGAN for Object-Preserving Image-to-Image Domain Adaptation. In *(ECCV)*. 498–513.
[37] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*. 2223–2232.