Maximum Classifier Discrepancy for Unsupervised Domain Adaptation

Kuniaki Saito[1], Kohei Watanabe[1], Yoshitaka Ushiku[1], and Tatsuya Harada[1,2]

[1]The University of Tokyo, [2]RIKEN
{k-saito,watanabe,ushiku,harada}@mi.t.u-tokyo.ac.jp

Abstract

In this work, we present a method for unsupervised domain adaptation. Many adversarial learning methods train domain classifier networks to distinguish the features as either a source or target and train a feature generator network to mimic the discriminator. Two problems exist with these methods. First, the domain classifier only tries to distinguish the features as a source or target and thus does not consider task-specific decision boundaries between classes. Therefore, a trained generator can generate ambiguous features near class boundaries. Second, these methods aim to completely match the feature distributions between different domains, which is difficult because of each domain's characteristics.

To solve these problems, we introduce a new approach that attempts to align distributions of source and target by utilizing the task-specific decision boundaries. We propose to maximize the discrepancy between two classifiers' outputs to detect target samples that are far from the support of the source. A feature generator learns to generate target features near the support to minimize the discrepancy. Our method outperforms other methods on several datasets of image classification and semantic segmentation. The codes are available at `https://github.com/mil-tokyo/MCD_DA`

1. Introduction

The classification accuracy of images has improved substantially with the advent of deep convolutional neural networks (CNN) which utilize numerous labeled samples [16]. However, collecting numerous labeled samples in various domains is expensive and time-consuming.

Domain adaptation (DA) tackles this problem by transferring knowledge from a label-rich domain (i.e., source domain) to a label-scarce domain (i.e., target domain). DA aims to train a classifier using source samples that generalize well to the target domain. However, each domain's samples have different characteristics, which makes the problem difficult to solve. Consider neural networks trained on labeled source images collected from the Web. Although such neural networks perform well on the source images, correctly recognizing target images collected from a real camera is difficult for them. This is because the target images can have different characteristics from the source images, such as change of light, noise, and angle in which the image is captured. Furthermore, regarding unsupervised DA (UDA), we have access to labeled source samples and only unlabeled target samples. We must construct a model that works well on target samples despite the absence of their labels during training. UDA is the most challenging situation, and we propose a method for UDA in this study.

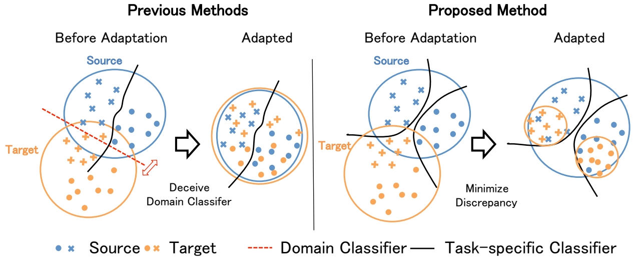


Figure 1. (Best viewed in color.) Comparison of previous and the proposed distribution matching methods.. **Left**: Previous methods try to match different distributions by mimicing the domain classifier. They do not consider the decision boundary. **Right**: Our proposed method attempts to detect target samples outside the support of the source distribution using task-specific classifiers.

Many UDA algorithms, particularly those for training neural networks, attempt to match the distribution of the source features with that of the target without considering the category of the samples [8, 37, 4, 40]. In particular, domain classifier-based adaptation algorithms have been applied to many tasks [8, 4]. The methods utilize two players to align distributions in an adversarial manner: domain classifier (i.e., a discriminator) and feature generator. Source and target samples are input to the same feature generator.

Features from the feature generator are shared by the discriminator and a task-specific classifier. The discriminator is trained to discriminate the domain labels of the features generated by the generator whereas the generator is trained to fool it. The generator aims to match distributions between the source and target because such distributions will mimic the discriminator. They assume that such target features are classified correctly by the task-specific classifier because they are aligned with the source samples.

However, this method should fail to extract discriminative features because it does not consider the relationship between target samples and the task-specific decision boundary when aligning distributions. As shown in the left side of Fig. 1, the generator can generate ambiguous features near the boundary because it simply tries to make the two distributions similar.

To overcome both problems, we propose to align distributions of features from source and target domain by using the classifier's output for the target samples.

We introduce a new adversarial learning method that utilizes two types of players: task-specific classifiers and a feature generator. *task-specific classifiers* denotes the classifiers trained for each task such as object classification or semantic segmentation. Two classifiers take features from the generator. Two classifiers try to classify source samples correctly and, simultaneously, are trained to detect the target samples that are far from the support of the source. The samples existing far from the support do not have discriminative features because they are not clearly categorized into some classes. Thus, our method utilizes the task-specific classifiers as a discriminator. Generator tries to fool the classifiers. In other words, it is trained to generate target features near the support while considering classifiers' output for target samples. Thus, our method allows the generator to generate discriminative features for target samples because it considers the relationship between the decision boundary and target samples. This training is achieved in an adversarial manner. In addition, please note that we do not use domain labels in our method.

We evaluate our method on image recognition and semantic segmentation. In many settings, our method outperforms other methods by a large margin. The contributions of our paper are summarized as follows:

- We propose a novel adversarial training method for domain adaptation that tries to align the distribution of a target domain by considering task-specific decision boundaries.
- We confirm the behavior of our method through a toy problem.
- We extensively evaluate our method on various tasks: digit classification, object classification, and semantic segmentation.

2. Related Work

Training CNN for DA can be realized through various strategies. Ghifary *et al*. proposed using an autoencoder for the target domain to obtain domain-invariant features [9]. Sener *et al*. proposed using clustering techniques and pseudo-labels to obtain discriminative features [33]. Taigman *et al*. proposed cross-domain image translation methods [38]. Matching distributions of the middle features in CNN is considered to be effective in realizing an accurate adaptation. To this end, numerous methods have been proposed [8, 37, 4, 29, 40, 36].

The representative method of distribution matching involves training a domain classifier using the middle features and generating the features that deceive the domain classifier [8]. This method utilizes the techniques used in generative adversarial networks [10]. The domain classifier is trained to predict the domain of each input, and the category classifier is trained to predict the task-specific category labels. Feature extraction layers are shared by the two classifiers. The layers are trained to correctly predict the label of source samples as well as to deceive the domain classifier. Thus, the distributions of the middle features of the target and source samples are made similar. Some methods utilize maximum mean discrepancy (MMD) [22, 21], which can be applied to measure the divergence in high-dimensional space between different domains. This approach can train the CNN to simultaneously minimize both the divergence and category loss for the source domain. These methods are based on the theory proposed by [2], which states that the error on the target domain is bounded by the divergence of the distributions. To our understanding, these distribution aligning methods using GAN or MMD do not consider the relationship between target samples and decision boundaries. To tackle these problems, we propose a novel approach using task-specific classifiers as a discriminator.

Consensus regularization is a technique used in multi-source domain adaptation and multi-view learning, in which multiple classifiers are trained to maximize the consensus of their outputs [23]. In our method, we address a training step that minimizes the consensus of two classifiers, which is totally different from consensus regularization. Consensus regularization utilizes samples of multi-source domains to construct different classifiers as in [23]. In order to construct different classifiers, it relies on the different characteristics of samples in different source domains. By contrast, our method can construct different classifiers from only one source domain.

3. Method

In this section, we present the detail of our proposed method. First, we give the overall idea of our method in Section 3.1. Second, we explain about the loss function we

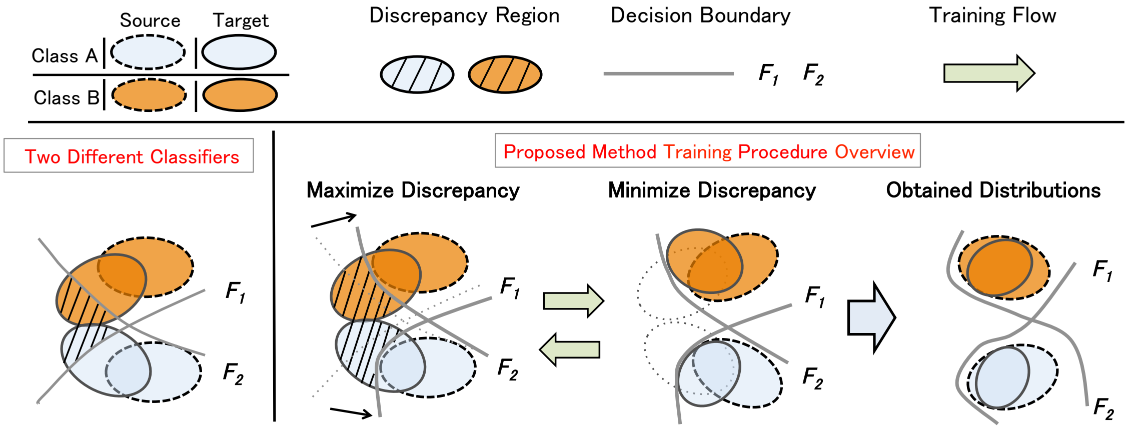


Figure 2. (Best viewed in color.) Example of two classifiers with an overview of the proposed method. Discrepancy refers to the disagreement between the predictions of two classifiers. First, we can see that the target samples outside the support of the source can be measured by two different classifiers (Leftmost, *Two different classifiers*). Second, regarding the training procedure, we solve a minimax problem in which we find two classifiers that *maximize* the discrepancy on the target sample, and then generate features that *minimize* this discrepancy.

used in experiments in Section 3.2. Finally, we explain the entire training procedure of our method in Section 3.3.

3.1. Overall Idea

We have access to a labeled source image $\mathbf{x_s}$ and a corresponding label y_s drawn from a set of labeled source images $\{X_s, Y_s\}$, as well as an unlabeled target image $\mathbf{x_t}$ drawn from unlabeled target images X_t. We train a feature generator network G, which takes inputs $\mathbf{x_s}$ or $\mathbf{x_t}$, and classifier networks F_1 and F_2, which take features from G. F_1 and F_2 classify them into K classes, that is, they output a K-dimensional vector of logits. We obtain class probabilities by applying the softmax function for the vector. We use the notation $p_1(\mathbf{y}|\mathbf{x})$, $p_2(\mathbf{y}|\mathbf{x})$ to denote the K-dimensional probabilistic outputs for input $\mathbf{x}$ obtained by F_1 and F_2 respectively.

The goal of our method is to align source and target features by utilizing the task-specific classifiers as a discriminator in order to consider the relationship between class boundaries and target samples. For this objective, we have to detect target samples far from the support of the source. The question is how to detect target samples far from the support. These target samples are likely to be misclassified by the classifier learned from source samples because they are near the class boundaries. Then, in order to detect these target samples, we propose to utilize the disagreement of the two classifiers on the prediction for target samples. Consider two classifiers (F_1 and F_2) that have different characteristics in the leftmost side of Fig. 2. We assume that the two classifiers can classify source samples correctly. This assumption is realistic because we have access to labeled source samples in the setting of UDA. In addition, please note that F_1 and F_2 are initialized differently to obtain different classifiers from the beginning of training. Here, we have the key intuition that target samples outside the support of the source are likely to be classified differently by the two distinct classifiers. This region is denoted by black lines in the leftmost side of Fig. 2 (*Discrepancy Region*). Conversely, if we can measure the disagreement between the two classifiers and train the generator to minimize the disagreement, the generator will avoid generating target features outside the support of the source. Here, we consider measuring the difference for a target sample using the following equation, $d(p_1(\mathbf{y}|\mathbf{x_t}), p_2(\mathbf{y}|\mathbf{x_t}))$ where d denotes the function measuring divergence between two probabilistic outputs. This term indicates how the two classifiers disagree on their predictions and, hereafter, we call the term as *discrepancy*. Our goal is to obtain a feature generator that can minimize the discrepancy on target samples.

In order to effectively detect target samples outside the support of the source, we propose to train discriminators (F_1 and F_2) to maximize the discrepancy given target features (*Maximize Discrepancy* in Fig. 2). Without this operation, the two classifiers can be very similar ones and cannot detect target samples outside the support of the source. We then train the generator to fool the discriminator, that is, by minimizing the discrepancy (*Minimize Discrepancy* in Fig. 2). This operation encourages the target samples to be generated inside the support of the source. This adversarial learning steps are repeated in our method. Our goal is to obtain the features, in which the support of the target is included by that of the source (*Obtained Distributions* in Fig. 2). We show the loss function used for discrepancy loss in the next section. Then, we detail the training procedure.

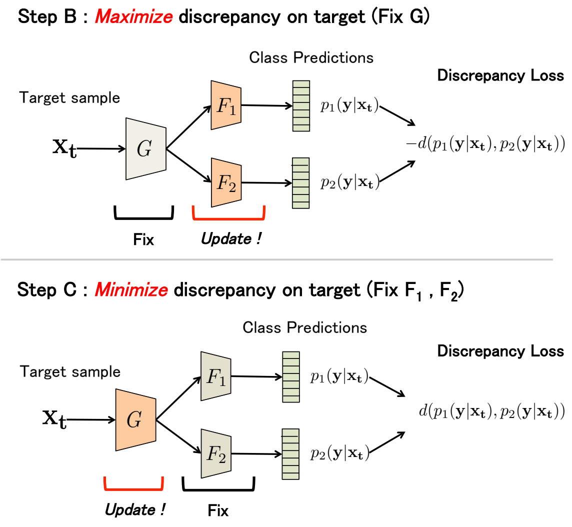

Step B : Maximize discrepancy on target (Fix G)
Class Predictions
Discrepancy Loss
Target sample
$\mathbf{x_t}$
G
F_1
F_2
$p_1(\mathbf{y}|\mathbf{x_t})$
$p_2(\mathbf{y}|\mathbf{x_t})$
$-d(p_1(\mathbf{y}|\mathbf{x_t}), p_2(\mathbf{y}|\mathbf{x_t}))$
Fix
Update !
Step C : Minimize discrepancy on target (Fix F1 , F2)
Class Predictions
Discrepancy Loss
Target sample
$\mathbf{x_t}$
G
F_1
F_2
$p_1(\mathbf{y}|\mathbf{x_t})$
$p_2(\mathbf{y}|\mathbf{x_t})$
$d(p_1(\mathbf{y}|\mathbf{x_t}), p_2(\mathbf{y}|\mathbf{x_t}))$
Update !
Fix

Figure 3. Adversarial training steps of our method. We separate the network into two modules: generator (G) and classifiers (F_1, F_2). The classifiers learn to maximize the discrepancy **Step B** on the target samples, and the generator learns to minimize the discrepancy **Step C**. Please note that we employ a training **Step A** to ensure the discriminative features for source samples.

3.2. Discrepancy Loss

In this study, we utilize the absolute values of the difference between the two classifiers' probabilistic outputs as discrepancy loss:

$$d(p_1, p_2) = \frac{1}{K}\sum_{k=1}^{K} |{p_1}_k - {p_2}_k|, \tag{1}$$

where the ${p_1}_k$ and ${p_2}_k$ denote probability output of p_1 and p_2 for class k respectively. The choice for L1-distance is based on the Theorem . Additionally, we experimentally found that L2-distance does not work well.

3.3. Training Steps

To sum up the previous discussion in Section 3.1, we need to train two classifiers, which take inputs from the generator and maximize $d(p_1(\mathbf{y}|\mathbf{x_t}), p_2(\mathbf{y}|\mathbf{x_t}))$, and the generator which tries to mimic the classifiers. Both the classifiers and generator must classify source samples correctly. We will show the manner in which to achieve this. We solve this problem in three steps.

Step A First, we train both classifiers and generator to classify the source samples correctly. In order to make classifiers and generator obtain task-specific discriminative features, this step is crucial. We train the networks to minimize softmax cross entropy. The objective is as follows:

$$\min_{G, F_1, F_2} \mathcal{L}(X_s, Y_s). \tag{2}$$

$$\mathcal{L}(X_s, Y_s) = -\mathbb{E}_{(\mathbf{x_s}, y_s)\sim(X_s, Y_s)} \sum_{k=1}^{K} \mathbb{1}_{[k=y_s]} \log p(\mathbf{y}|\mathbf{x}_s) \tag{3}$$

Step B In this step, we train the classifiers (F_1, F_2) as a discriminator for a fixed generator (G). By training the classifiers to increase the discrepancy, they can detect the target samples excluded by the support of the source. This step corresponds to **Step B** in Fig. 3. We add a classification loss on the source samples. Without this loss, we experimentally found that our algorithm's performance drops significantly. We use the same number of source and target samples to update the model. The objective is as follows:

$$\min_{F_1, F_2} \mathcal{L}(X_s, Y_s) - \mathcal{L}_{\text{adv}}(X_t). \tag{4}$$

$$\mathcal{L}_{\text{adv}}(X_t) = \mathbb{E}_{\mathbf{x_t}\sim X_t}[d(p_1(\mathbf{y}|\mathbf{x_t}), p_2(\mathbf{y}|\mathbf{x_t}))] \tag{5}$$

Step C We train the generator to minimize the discrepancy for fixed classifiers. This step corresponds to **Step C** in Fig. 3. The number n indicates the number of times we repeat this for the same mini-batch. This number is a hyperparameter of our method. This term denotes the trade-off between the generator and the classifiers. The objective is as follows:

$$\min_{G} \mathcal{L}_{\text{adv}}(X_t). \tag{6}$$

These three steps are repeated in our method. To our understanding, the order of the three steps is not important. Instead, our major concern is to train the classifiers and generator in an adversarial manner under the condition that they can classify source samples correctly.

3.4. Theoretical Insight

Since our method is motivated by the theory proposed by Ben-David *et al.* [1], we want to show the relationship between our method and the theory in this section.

Ben-David *et al.* [1] proposed the theory that bounds the expected error on the target samples, $R_{\mathcal{T}}(h)$, by using three terms: (i) expected error on the source domain, $R_{\mathcal{S}}(h)$; (ii) $\mathcal{H}\Delta\mathcal{H}$-distance ($d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S}, \mathcal{T})$), which is measured as the discrepancy between two classifiers; and (iii) the shared error of the ideal joint hypothesis, λ. $\mathcal{S}$ and $\mathcal{T}$ denote source and target domain respectively. Another theory [2] bounds the error on the target domain, which introduced $\mathcal{H}$-distance ($d_{\mathcal{H}}(\mathcal{S}, \mathcal{T})$) for domain divergence. The two theories and their relationships can be explained as follows.

Theorem 1 *Let H be the hypothesis class. Given two domains $\mathcal{S}$ and $\mathcal{T}$, we have*

$$\begin{aligned} \forall h \in H, R_{\mathcal{T}}(h) &\leq R_{\mathcal{S}}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S}, \mathcal{T}) + \lambda \\ &\leq R_{\mathcal{S}}(h) + \frac{1}{2} d_{\mathcal{H}}(\mathcal{S}, \mathcal{T}) + \lambda \end{aligned} \tag{7}$$

where

$$d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S},\mathcal{T}) = 2 \sup_{(h,h')\in\mathcal{H}^2} \left| \mathop{\mathbf{E}}_{\mathbf{x}\sim\mathcal{S}} \mathrm{I}\big[h(\mathbf{x}) \neq h^{'}(\mathbf{x})\big] - \mathop{\mathbf{E}}_{\mathbf{x}\sim\mathcal{T}} \mathrm{I}\big[h(\mathbf{x}) \neq h^{'}(\mathbf{x})\big] \right|$$

$$d_{\mathcal{H}}(\mathcal{S},\mathcal{T}) = 2 \sup_{h\in\mathcal{H}} \left| \mathop{\mathbf{E}}_{\mathbf{x}\sim\mathcal{S}} \mathrm{I}\big[h(\mathbf{x}) \neq 1\big] - \mathop{\mathbf{E}}_{\mathbf{x}\sim\mathcal{T}} \mathrm{I}\big[h(\mathbf{x}) \neq 1\big] \right|,$$

$$\lambda = \min\left[R_{\mathcal{S}}(h) + R_{\mathcal{T}}(h)\right]$$

Here, $R_{\mathcal{T}}(h)$ *is the error of hypothesis* h *on the target domain, and* $R_{\mathcal{S}}(h)$ *is the corresponding error on the source domain.* $\mathrm{I}[a]$ *is the indicator function, which is 1 if predicate a is true and 0 otherwise.*

$\mathcal{H}$-distance is shown to be empirically measured by the error of the domain classifier, which is trained to discriminate the domain of features. λ is a constant—the shared error of the ideal joint hypothesis—which is considered sufficiently low to achieve an accurate adaptation. Earlier studies [8, 37, 4, 29, 40] attempted to measure and minimize $\mathcal{H}$-distance in order to realize the adaptation. As this inequality suggests, $\mathcal{H}$-distance upper-bounds the $\mathcal{H}\Delta\mathcal{H}$-distance. We will show the relationship between our method and $\mathcal{H}\Delta\mathcal{H}$-distance.

Regarding $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S},\mathcal{T})$, if we consider that h and h' can classify source samples correctly, the term $\mathop{\mathbf{E}}_{\mathbf{x}\sim\mathcal{S}} \mathrm{I}\big[h(\mathbf{x}) \neq h^{'}(\mathbf{x})\big]$ is assumed to be very low. h and h' should agree on their predictions on source samples. Thus, $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S},\mathcal{T})$ is approximately calculated as $\sup_{(h,h')\in\mathcal{H}^2} \mathop{\mathbf{E}}_{\mathbf{x}\sim\mathcal{T}} \mathrm{I}\big[h(\mathbf{x}) \neq h^{'}(\mathbf{x})\big]$, which denotes the supremum of the expected disagreement of two classifiers' predictions on target samples.

We assume that h and $h^{'}$ share the feature extraction part. Then, we decompose the hypothesis h into G and F_1, and $h^{'}$ into G and F_2. G, F_1 and F_2 correspond to the network in our method. If we substitute these notations into the $\sup_{(h,h')\in\mathcal{H}^2} \mathop{\mathbf{E}}_{\mathbf{x}\sim\mathcal{T}} \mathrm{I}\big[h(\mathbf{x}) \neq h^{'}(\mathbf{x})\big]$ and for fixed G, the term will become

$$\sup_{F_1,F_2} \mathop{\mathbf{E}}_{\mathbf{x}\sim\mathcal{T}} \mathrm{I}\left[F_1 \circ G(\mathbf{x}) \neq F_2 \circ G(\mathbf{x})\right]. \tag{8}$$

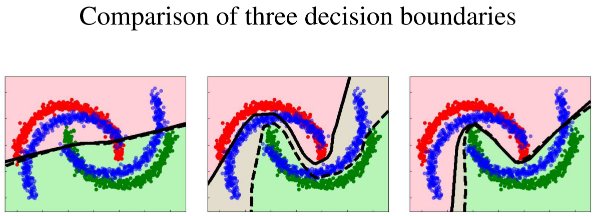
Furthermore, if we replace sup with max and minimize the term with respect to G, we obtain

$$\min_{G} \max_{F_1,F_2} \mathop{\mathbf{E}}_{\mathbf{x}\sim\mathcal{T}} \mathrm{I}\left[F_1 \circ G(\mathbf{x}) \neq F_2 \circ G(\mathbf{x})\right]. \tag{9}$$

This equation is very similar to the mini-max problem we solve in our method, in which classifiers are trained to maximize their discrepancy on target samples and generator tries to minimize it. Although we must train all networks to minimize the classification loss on source samples, we can see the connection to the theory proposed by [1].

4. Experiments on Classification

First, we observed the behavior of our model on toy problem. Then, we performed an extensive evaluation of the proposed methods on the following datasets: digits, traffic signs, and object classification.

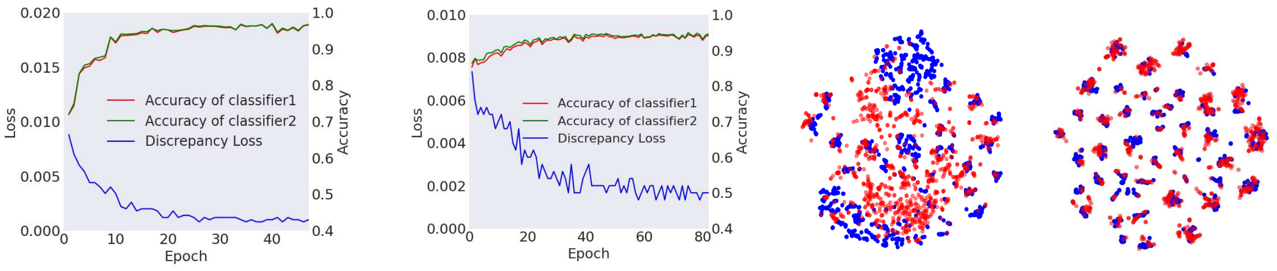


(a) Source Only (b) No Step C (c) Proposed

Figure 4. (Best viewed in color.) Red and green points indicate the source samples of class 0 and 1, respectively. Blue points are target samples generated by rotating source samples. The dashed and normal lines are two decision boundaries in our method. The pink and light green regions are where the results of both classifiers are class 0 and 1, respectively. Fig. 4(a) is the model trained only on source samples. Fig. 4(b) is the model trained to increase discrepancy of the two classifiers on target samples without using Step C. Fig. 4(c) shows our proposed method.

4.1. Experiments on Toy Datasets

In the first experiment, we observed the behavior of the proposed method on *inter twinning moons* 2D problems, in which we used *scikit-learn* [27] to generate the target samples by rotating the source samples. The goal of the experiment was to observe the learned classifiers' boundary. For the source samples, we generated a lower moon and an upper moon, labeled 0 and 1, respectively. Target samples were generated by rotating the angle of the distribution of the source samples. We generated 300 source and target samples per class as the training samples. In this experiment, we compared the decision boundary obtained from our method with that obtained from both the model trained only on source samples and from that trained only to increase the discrepancy. In order to train the second comparable model, we simply skipped Step C in Section 3.3 during training. We tested the method on 1000 target samples and visualized the learned decision boundary with source and target samples. Other details including the network architecture used in this experiment are provided in our supplementary material. As we expected, when we trained the two classifiers to increase the discrepancy on the target samples, two classifiers largely disagreed on their predictions on target samples (Fig. 4(b)). This is clear when compared to the source only model (Fig. 4(a)). Two classifiers were trained on the source samples without adaptation, and the boundaries seemed to be nearly the same. Then, our proposed method attempted to generate target samples that reduce the discrepancy. Therefore, we could expect that the two classifiers will be similar. Fig. 4(c) demonstrates the assumption. The decision boundaries are drawn considering the target samples. The two classifiers output nearly the same prediction for target samples, and they classified most target samples correctly.



(a) SVHN to MNIST

(b) SYN SIGN to GTSRB

(c) Source Only

(d) Adapted (Ours)

Figure 5. (Best viewed in color.) **Left**: Relationship between discrepancy loss (**blue** line) and accuracy (**red** and **green** lines) during training. As discrepancy loss decreased, accuracy improved. **Right**: Visualization of features obtained from last pooling layer of the generator in adaptation from SYN SIGNS to GTSRB using t-SNE [24]. **Red** and **blue** points indicate the target and source samples, respectively. All samples are testing samples. We can see that applying our method makes the target samples discriminative.

METHOD	**SVHN** to **MNIST**	**SYNSIG** to **GTSRB**	**MNIST** to **USPS**	**MNIST*** to **USPS***	**USPS** to **MNIST**
Source Only	67.1	85.1	76.7	79.4	63.4
Distribution Matching based Methods					
MMD † [21]	71.1	91.1	-	81.1	-
DANN † [7]	71.1	88.7	77.1±1.8	85.1	73.0±0.2
DSN † [4]	82.7	93.1	91.3	-	-
ADDA [39]	76.0±1.8	-	89.4±0.2	-	90.1±0.8
CoGAN [19]	-	-	91.2±0.8	-	89.1±0.8
PixelDA [3]	-	-	-	95.9	-
Ours ($n = 2$)	94.2±2.6	93.5±0.4	92.1±0.8	93.1±1.9	90.0±1.4
Ours ($n = 3$)	95.9±0.5	94.0±0.4	93.8±0.8	95.6±0.9	91.8±0.9
Ours ($n = 4$)	**96.2**±0.4	**94.4**±0.3	**94.2**±0.7	**96.5**±0.3	**94.1**±0.3
Other Methods					
ATDA † [32]	86.2	96.2	-	-	-
ASSC [11]	95.7±1.5	82.8±1.3	-	-	-
DRCN [9]	82.0±0.1	-	91.8±0.09	-	73.7±0.04

Table 1. Results of the visual DA experiment on the digits and traffic signs datasets. The results are cited from each study. The score of MMD is cited from DSN [4]. Please note that † means that the method used a few labeled target samples as validation, which is different from our setting. We repeated each experiment 5 times and report the average and the standard deviation of the accuracy. The accuracy was obtained from classifier F_1. Including the methods that used the labeled target samples for validation, our method achieved good performance. MNIST* and USPS* mean that we used all of the training samples to train the model.

4.2. Experiments on Digits Datasets

In this experiment, we evaluate the adaptation of the model on three scenarios. The example datasets are presented in the supplementary material.

We assessed four types of adaptation scenarios by using the digits datasets, namely MNIST [17], Street View House Numbers (SVHN) [26], and USPS [14]. We further evaluated our method on the traffic sign datasets, Synthetic Traffic Signs (SYN SIGNS) [25] and the German Traffic Signs Recognition Benchmark [35] (GTSRB). In this experiment, we employed the CNN architecture used in [7] and [3]. We added batch normalization to each layer in these models. We used Adam [15] to optimize our model and set the learning rate as 2.0×10^{-4} in all experiments. We set the batch size to 128 in all experiments. The hyper-parameter peculiar to our method was n, which denotes the number of times we update the feature generator to mimic classifiers. We varied the value of n from 2 to 4 in our experiment and observed the sensitivity to the hyper-parameter. We followed the protocol of unsupervised domain adaptation and did not use validation samples to tune hyper-parameters. The other details are provided in our supplementary material due to a limit of space.

SVHN→MNIST SVHN [26] and MNIST [17] have distinct properties because SVHN datasets contain images with a colored background, multiple digits, and extremely blurred digits, meaning that the domain divergence is very large between these datasets.

SYN SIGNS→GTSRB In this experiment, we evaluated the adaptation from synthesized traffic signs datasets (SYN SIGNS dataset [7]) to real-world signs datasets (GTSRB dataset [35]). These datasets contain 43 types of classes.

MNIST↔USPS We also evaluate our method on MNIST and USPS datasets [17] to compare our method with other methods. We followed the different protocols provided by the paper, ADDA [39] and PixelDA [3].

Results Table 1 lists the accuracies for the target samples, and Fig. 5(a) and 5(b) show the relationship between the discrepancy loss and accuracy during training. For the *source only* model, we used the same network architecture as used in our method. Details are provided in the supplementary material. We extensively compared our methods with distribution matching-based methods as shown in Table 1. The proposed method outperformed these methods in all settings. The performance improved as we increased the value of n. Although other methods such as ATDA [32] performed better than our method in some situations, the method utilized a few labeled target samples to decide hyper-parameters for each dataset. The performance of our method will improve too if we can choose the best hyper-parameters for each dataset. As Fig. 5(a) and

Method	plane	bcycl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	truck	mean
Source Only	55.1	53.3	61.9	59.1	80.6	17.9	79.7	31.2	81.0	26.5	73.5	8.5	52.4
MMD [21]	87.1	63.0	76.5	42.0	90.3	42.9	**85.9**	53.1	49.7	36.3	**85.8**	20.7	61.1
DANN [7]	81.9	**77.7**	82.8	44.3	81.2	29.5	65.1	28.6	51.9	**54.6**	82.8	7.8	57.4
Ours ($n = 2$)	81.1	55.3	83.6	**65.7**	87.6	72.7	83.1	73.9	85.3	47.7	73.2	27.1	69.7
Ours ($n = 3$)	**90.3**	49.3	82.1	62.9	**91.8**	69.4	83.8	72.8	79.8	53.3	81.5	**29.7**	70.6
Ours ($n = 4$)	87.0	60.9	**83.7**	64.0	88.9	**79.6**	84.7	**76.9**	**88.6**	40.3	83.0	25.8	**71.9**

Table 2. Accuracy of ResNet101 model fine-tuned on the VisDA dataset. The reported accuracy was obtained after 10 epoch updates.

5(b) show, as the discrepancy loss diminishes, the accuracy improves, confirming that minimizing the discrepancy for target samples can result in accurate adaptation. We visualized learned features as shown in Fig. 5(c) and 5(d). Our method did not match the distributions of source and target completely as shown in Fig. 5(d). However, the target samples seemed to be aligned with each class of source samples. Although the target samples did not separate well in the non-adapted situation, they did separate clearly as do source samples in the adapted situation.

4.3. Experiments on VisDA Classification Dataset

We further evaluated our method on an object classification setting. The VisDA dataset [28] was used in this experiment, which evaluated adaptation from synthetic-object to real-object images. To date, this dataset represents the largest for cross-domain object classification, with over 280K images across 12 categories in the combined training, validation, and testing domains. The source images were generated by rendering 3D models of the same object categories as in the real data from different angles and under different lighting conditions. It contains 152,397 synthetic images. The validation images were collected from MSCOCO [18] and they amount to 55,388 in total. In our experiment, we considered the images of validation splits as the target domain and trained models in unsupervised domain adaptation settings. We evaluate the performance of ResNet101 [12] model pre-trained on Imagenet [6]. The final fully-connected layer was removed and all layers were updated with the same learning rate because this dataset has abundant source and target samples. We regarded the pre-trained model as a generator network and we used three-layered fully-connected networks for classification networks. The batch size was set to 32 and we used SGD with learning rate 1.0×10^{-3} to optimize the model. We report the accuracy after 10 epochs. The training details for baseline methods are written in our supplementary material due to the limit of space.

Results Our method achieved an accuracy much better than other distribution matching based methods (Table 2). In addition, our method performed better than the source only model in all classes, whereas MMD and DANN perform worse than the source only model in some classes such as car and plant. We can clearly see the clear effectiveness of our method in this regard. In this experiment, as the value of n increase, the performance improved. We think that it was because of the large domain difference between synthetic objects and real images. The generator had to be updated many times to align such distributions.

5. Experiments on Semantic Segmentation

We further applied our method to semantic segmentation. Considering a huge annotation cost for semantic segmentation datasets, adaptation between different domains is an important problem in semantic segmentation.

Implementation Detail We used the publicly available synthetic dataset GTA5 [30] or Synthia [31] as the source domain dataset and real dataset Cityscapes [5] as the target domain dataset. Following the work [13, 42], the Cityscapes validation set was used as our test set. As our training set, the Cityscapes train set was used. During training, we randomly sampled just a single sample (setting the batch size to 1 because of the GPU memory limit) from both the images (and their labels) of the source dataset and the remaining images of the target dataset but with no labels.

We applied our method to VGG-16 [34] based FCN-8s [20] and DRN-D-105 [41] to evaluate our method. The details of models, including their architecture and other hyper-parameters, are described in the supplementary material.

We used Momentum SGD to optimize our model and set the momentum rate to 0.9 and the learning rate to 1.0×10^{-3} in all experiments. The image size was resized to 1024×512. Here, we report the output of F_1 after 50,000 iterations.

Results Table 3, Table 4, and Fig. 6 show quantitative and qualitative results, respectively. These results illustrate that even with a large domain difference between synthetic to real images, our method is capable of improving the performance. Considering the mIoU of the model trained only on source samples, we can see the clear effectiveness of our adaptation method. Also, compared to the score of DANN, our method shows clearly better performance.

6. Conclusion

In this paper, we proposed a new approach for UDA, which utilizes task-specific classifiers to align distributions.

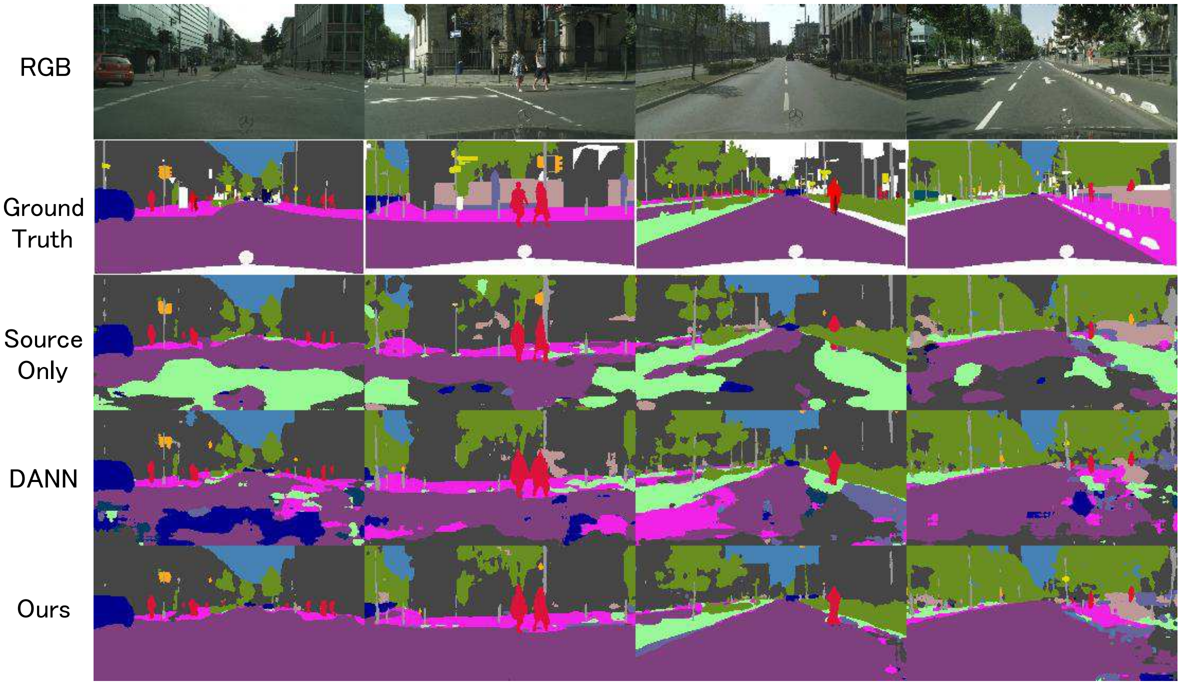


Figure 6. Qualitative results on adaptation from GTA5 to Cityscapes. DRN-105 is used to obtain these results.

Network	method	mIoU	road	sdwk	bldng	wall	fence	pole	light	sign	vgttn	trrn	sky	person	rider	car	truck	bus	train	mcycl	bcycl
VGG-16	Source Only	24.9	25.9	10.9	50.5	3.3	12.2	25.4	28.6	13.0	78.3	7.3	63.9	52.1	7.9	66.3	5.2	7.8	0.9	13.7	0.7
	FCN Wld [13]	27.1	70.4	32.4	62.1	14.9	5.4	10.9	14.2	2.7	79.2	21.3	64.6	44.1	4.2	70.4	8.0	7.3	0.0	3.5	0.0
	CDA (I) [42]	23.1	26.4	10.8	69.7	10.2	9.4	20.2	13.6	14.0	56.9	2.8	63.8	31.8	10.6	60.5	10.9	3.4	**10.9**	3.8	9.5
	Ours (k=2)	28.0	87.4	15.4	75.5	17.4	9.9	16.2	11.9	0.6	80.6	28.1	60.2	32.5	0.9	75.4	13.6	4.8	0.1	0.7	0.0
	Ours (k=3)	27.3	86.0	10.5	75.1	20.0	2.9	19.4	8.4	0.7	78.4	19.4	74.8	23.2	0.3	74.1	14.3	10.4	0.2	0.1	0.0
	Ours (k=4)	28.8	86.4	8.5	76.1	18.6	9.7	14.9	7.8	0.6	82.8	32.7	71.4	25.2	1.1	76.3	16.1	17.1	1.4	0.2	0.0
DRN-105	Source Only	22.2	36.4	14.2	67.4	16.4	12.0	20.1	8.7	0.7	69.8	13.3	56.9	37.0	0.4	53.6	10.6	3.2	0.2	0.9	0.0
	DANN [7]	32.8	64.3	23.2	73.4	11.3	**18.6**	**29.0**	**31.8**	14.9	82.0	16.8	73.2	53.9	12.4	53.3	20.4	11.0	5.0	18.7	9.8
	Ours (k=2)	**39.7**	90.3	31.0	78.5	19.7	17.3	28.6	30.9	**16.1**	83.7	30.0	69.1	**58.5**	**19.6**	81.5	23.8	**30.0**	5.7	**25.7**	**14.3**
	Ours (k=3)	38.9	**90.8**	**35.6**	**80.5**	22.9	15.5	27.5	24.9	15.1	84.2	31.8	77.4	54.6	17.2	82.0	21.6	29.0	1.3	21.8	5.3
	Ours (k=4)	38.1	89.2	23.2	80.2	**23.6**	18.1	**27.7**	25.0	9.3	**84.4**	**34.6**	**79.5**	53.2	16.0	**84.1**	**26.0**	22.5	5.2	16.7	4.8

Table 3. Adaptation results on the semantic segmentation. We evaluate adaptation from GTA5 to Cityscapes dataset.

Network	method	mIoU	road	sdwlk	bldng	wall	fence	pole	light	sign	vgttn	sky	prsn	ridr	car	bus	mcycl	bcycl
VGG-16	Source Only [42]	22.0	5.6	11.2	59.6	0.8	0.5	21.5	8.0	5.3	72.4	75.6	35.1	9.0	23.6	4.5	0.5	**18.0**
	FCN Wld [13]	20.2	11.5	19.6	30.8	4.4	0.0	20.3	0.1	**11.7**	42.3	68.7	51.2	3.8	54.0	3.2	0.2	0.6
	CDA (I+SP) [42]	29.0	65.2	26.1	74.9	0.1	**0.5**	10.7	3.7	3.0	76.1	70.6	47.1	8.2	43.2	20.7	0.7	13.1
DRN_105	Source Only	23.4	14.9	11.4	58.7	1.9	0.0	24.1	1.2	6.0	68.8	76.0	**54.3**	7.1	34.2	15.0	0.8	0.0
	DANN [7]	32.5	67.0	29.1	71.5	**14.3**	0.1	28.1	**12.6**	10.3	72.7	76.7	48.3	**12.7**	62.5	11.3	2.7	0.0
	Ours (k=2)	36.3	83.5	40.9	77.6	6.0	0.1	27.9	6.2	6.0	83.1	**83.5**	51.5	11.8	78.9	19.8	4.6	0.0
	Ours (k=3)	**37.3**	84.8	**43.6**	**79.0**	3.9	0.2	**29.1**	7.2	5.5	**83.8**	83.1	51.0	11.7	79.9	27.2	**6.2**	0.0
	Ours (k=4)	37.2	**88.1**	43.2	79.1	2.4	0.1	27.3	7.4	4.9	83.4	81.1	51.3	10.9	**82.1**	**29.0**	5.7	0.0

Table 4. Adaptation results on the semantic segmentation. We evaluate adaptation from Synthia to Cityscapes dataset.

We propose to utilize task-specific classifiers as discriminators that try to detect target samples that are far from the support of the source. A feature generator learns to generate target features near the support to fool the classifiers. Since the generator uses feedback from task-specific classifiers, it will avoid generating target features near class boundaries. We extensively evaluated our method on image classification and semantic segmentation datasets. In almost all experiments, our method outperformed state-of-the-art methods. We provide the results when applying gradient reversal layer [7] in the supplementary material, which enables to update parameters of the model in one step.

7. Acknowledgements

The work was partially supported by CREST, JST, and was partially funded by the ImPACT Program of the Council for Science, Technology, and Innovation (Cabinet Office, Government of Japan).

References

[1] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010. 4, 5

[2] S. Ben-David, J. Blitzer, K. Crammer, F. Pereira, et al. Analysis of representations for domain adaptation. In *NIPS*, 2007. 2, 4

[3] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *CVPR*, 2017. 6

[4] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. Domain separation networks. In *NIPS*, 2016. 1, 2, 5, 6

[5] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 7

[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

[7] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, 2014. 6, 7, 8

[8] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 17(59):1–35, 2016. 1, 2, 5

[9] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *ECCV*, 2016. 2, 6

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *NIPS*, 2014. 2

[11] P. Haeusser, T. Frerix, A. Mordvintsev, and D. Cremers. Associative domain adaptation. In *ICCV*, 2017. 6

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 7

[13] J. Hoffman, D. Wang, F. Yu, and T. Darrell. Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. *arXiv:1612.02649*, 2016. 7, 8

[14] J. J. Hull. A database for handwritten text recognition research. *PAMI*, 16(5):550–554, 1994. 6

[15] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014. 6

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 6

[18] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 7

[19] M.-Y. Liu and O. Tuzel. Coupled generative adversarial networks. In *NIPS*, 2016. 6

[20] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 7

[21] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2015. 2, 6, 7

[22] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Unsupervised domain adaptation with residual transfer networks. In *NIPS*, 2016. 2

[23] P. Luo, F. Zhuang, H. Xiong, Y. Xiong, and Q. He. Transfer learning from multiple source domains via consensus regularization. In *CIKM*, 2008. 2

[24] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *JMLR*, 9(11):2579–2605, 2008. 6

[25] B. Moiseev, A. Konev, A. Chigorin, and A. Konushin. Evaluation of traffic sign recognition methods trained on synthetically generated data. In *ACIVS*, 2013. 6

[26] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, 2011. 6

[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *JMLR*, 12(10):2825–2830, 2011. 5

[28] X. Peng, B. Usman, N. Kaushik, J. Hoffman, D. Wang, and K. Saenko. Visda: The visual domain adaptation challenge. *arXiv:1710.06924*, 2017. 7

[29] S. Purushotham, W. Carvalho, T. Nilanon, and Y. Liu. Variational recurrent adversarial deep domain adaptation. In *ICLR*, 2017. 2, 5

[30] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016. 7

[31] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016. 7

[32] K. Saito, Y. Ushiku, and T. Harada. Asymmetric tri-training for unsupervised domain adaptation. In *ICML*, 2017. 6

[33] O. Sener, H. O. Song, A. Saxena, and S. Savarese. Learning transferrable representations for unsupervised domain adaptation. In *NIPS*, 2016. 2

[34] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014. 7

[35] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *IJCNN*, 2011. 6

[36] B. Sun, J. Feng, and K. Saenko. Return of frustratingly easy domain adaptation. In *AAAI*, 2016. 2

[37] B. Sun and K. Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *ECCV Workshops*, 2016. 1, 2, 5

[38] Y. Taigman, A. Polyak, and L. Wolf. Unsupervised cross-domain image generation. In *ICLR*, 2017. 2

[39] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *CVPR*, 2017. 6

[40] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv:1412.3474*, 2014. 1, 2, 5

[41] F. Yu, V. Koltun, and T. Funkhouser. Dilated residual networks. In *CVPR*, 2017. 7

[42] Y. Zhang, P. David, and B. Gong. Curriculum domain adaptation for semantic segmentation of urban scenes. In *ICCV*, 2017. 7, 8