Generative Pseudo-label Refinement for Unsupervised Domain Adaptation

Pietro Morerio[1], Riccardo Volpi[1], Ruggero Ragonesi[1,2], Vittorio Murino[1,3,4]
{pietro.morerio,riccardo.volpi,ruggero.ragonesi,vittorio.murino}@iit.it
[1]Pattern Analysis & Computer Vision - Istituto Italiano di Tecnologia
[2]Università degli Studi di Genova, Italy
[3]Computer Science Department - Università di Verona, Italy
[4]Huawei Technologies Ltd., Ireland Research Center

Abstract

We investigate and characterize the inherent resilience of conditional Generative Adversarial Networks (cGANs) against noise in their conditioning labels, and exploit this fact in the context of Unsupervised Domain Adaptation (UDA). In UDA, a classifier trained on the labelled source set can be used to infer pseudo-labels on the unlabelled target set. However, this will result in a significant amount of misclassified examples (due to the well-known domain shift issue), which can be interpreted as noise injection in the ground-truth labels for the target set. We show that cGANs are, to some extent, robust against such "shift noise". Indeed, cGANs trained with noisy pseudo-labels, are able to filter such noise and generate cleaner target samples. We exploit this finding in an iterative procedure where a generative model and a classifier are jointly trained: in turn, the generator allows to sample cleaner data from the target distribution, and the classifier allows to associate better labels to target samples, progressively refining target pseudo-labels. Results on common benchmarks show that our method performs better or comparably with the unsupervised domain adaptation state of the art.

1. Introduction

Unsupervised Domain Adaptation (UDA) addresses the problem of learning models that perform well on a *target* domain for which ground truth annotations are not provided. During the training phase, one can leverage unlabeled samples from this distribution and labelled samples from a *source* distribution, separated by the so-called *domain shift* [46], *i.e.*, drawn from two different data distributions. In this work, we address UDA from a novel perspective, by casting the problem in the setting of *learning with noisy labels* [32].

We start from the very simple realization that, given a model trained on the source domain, we can infer a set of (pseudo-)labels for the target domain. Typically, due to the domain shift, a consistent number of labels are wrongly inferred, resulting in a noisy-labelled target set, with an amount of noise proportional to the classification error. Previous work [32] has shown that deep learning-based classifiers are robust against label noise, provided a sufficiently large training set. For this reason, one might hope that such resilience could be exploited to train a classifier for the target domain with the supervision of the inferred, noisy pseudo-labels. The idea is that the classifier might disregard label noise to some extent, providing target accuracy higher than the original model trained on source samples.

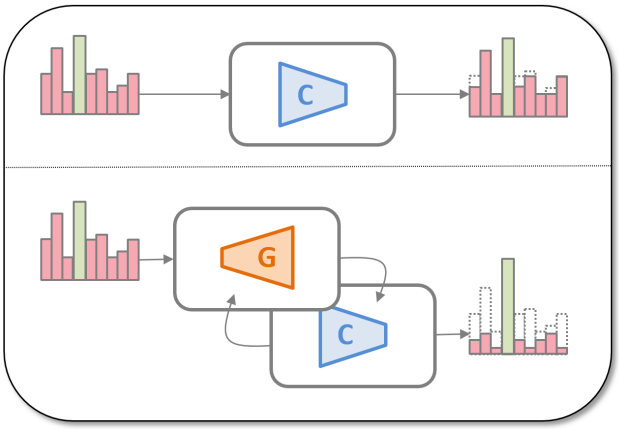


Figure 1. *Top*: a neural network classifier C is typically not robust against shift noise, here represented by an histogram. *Bottom*: a cGAN generator G is able to filter such structured noise, making it more uniform and thus tolerable from the classifier. By jointly training C and G, the former benefits from "cleaner" generated data, while the latter from more accurate inferred labels.

However, we empirically show that such strategies alone cannot compete with existing UDA methods in terms of accuracy on the target set. Indeed, while deep models' robustness against noisy labels is remarkable when the noise distribution is nearly uniform, we show that they are not robust

against the label noise resulting from the domain shift. Although *a priori* assumptions cannot be made on such noise, we empirically observe in a variety of adaptation problems heavy deviations from uniform noise, meaning that misclassications are not evenly distributed across classes. We term such highly structured noise *"shift noise"*.

While classifiers are not robust against such more structured kind on noise, we observe that conditional Generative Adversarial Networks [25] (cGAN) are. A cGAN model can be trained to generate samples conditioning on the desired classes of an arbitrary distribution. It was shown that this class of models can be *made* more resistant against label noise [45], but we provide empirical evidence that—to some extent—they are inherently robust against it, without any modification from the standard training procedure [10, 25]. This means, in practice, that training a cGAN on some noisy-labelled samples will result in a model that generates samples that are "cleaner" than the training ones. A natural idea that follows this finding, is trying to generate cleaner target samples to train better performing models for the target domain. However, although cGANs are to some extent resistant to noisy-labels, they are not robust enough to generate target samples that allow to train competitive models.

Interestingly though, we observe that, even if the noise reduction is not sufficient to train competitive target models, *the labels of the generated samples obey a noise distribution which is closer to the uniform than the shift noise one*.

In this work, we explore the two facts above (classifier robustness against uniform noise [32] and cGAN robustness to shift noise), and jointly exploit them for UDA. We devise a UDA strategy based on the properties of both classifiers and cGANs to filter out noise in the labels. We propose an iterative procedure, where we alternately optimize the losses associated with a cGAN and with a classifier. Throughout the training phase, the classifier can benefit from more and more reliable conditionally-generated data, while a cGAN can exploit more and more reliable pseudo-labels inferred by the classifier (see Figure 1). Source samples are only exploited to train an initial classifier. After this step, the problem is faced in a fully unsupervised fashion, reducing the noise on the labels of the empirical target distribution over iterations during training. Results on standard UDA benchmarks show the effectiveness of our approach.

Summary. The main contributions of this work are the following: **(i)** we characterize the concept of shift noise, and provide an analysis of the robustness of both discriminative and generative models against it; **(ii)** we design a novel training procedure that leverages the above findings in order to refine the predictions of a classifier over iterations; **(iii)** we apply the proposed algorithm in the unsupervised domain adaptation scenario, observing competitive performance with the state of the art on public benchmarks.

The remaining of the paper is organized as follows. In Section 2, we detail background and related work. In Section 3, we characterize shift noise and investigate the robustness of classifiers and cGANs against it. Section 4 describes how to exploit these findings to tackle UDA problems, and Section 5 reports the related experimental results. Finally, we draw the conclusions in Section 6.

2. Background and related work

Unsupervised Domain Adaptation. In UDA, we are given a set of samples from a source distribution in the form $\{x_s, y_s\} \sim p_{source}$, and a set of samples from a target distribution of interest in the form $\{x_t\} \sim p_{target}$ (no labels). The goal is to perform well on data from the target distribution. Different approaches allow to solve this problem efficiently in a plethora of tasks. Adversarial training has been effectively used to map source and target samples in a common feature space [7, 8, 47, 48]. Other works aim at aligning the second order statistics of source and target features [43, 27]. More recently, image-to-image translation methods, that learn the mapping from the source space to the target one and vice-versa, have been proposed [22, 44, 33, 3, 21, 38, 15]. In general, one can design models for UDA that leverage labeled source samples that are "rendered" with the style of target samples (and vice-versa). Other works propose different successful solutions to face the adaptation problem (*e.g.*, [4, 35, 39, 12, 36, 37]). Since the latter are only related to our work for the common goal, they are not detailed in this section. Our approach is somehow related to image-to-image translation methods, since we exploit generated samples to train a classifier for the target domain. In particular, PixelDA [3] is the most related method, since it leverages a training procedure where a GAN and a classifier are jointly trained. However, the latter makes a strong assumption on the relationship between source and target domains: *"the differences between the domains are primarily low-level (due to noise, resolution, illumination, color) rather than high-level (types of objects, geometric variations, etc)"*. In this work, we generate target images using a simple cGAN, namely mapping noise vectors from a latent space into the image space, merely conditioning on label codes. This difference comes with two main advantages: our architecture and loss functions are much simpler than the ones adopted for image-to-image translation, and we do not have to make such strong assumptions on the gap between the two domains.

Our method is substantially different from most UDA solutions also because we do not need source samples throughout the adaptation procedure, but only to pre-train the model M_{θ_s}, used to assign pseudo-labels to target samples. Indeed, solutions that align source/target feature statistics [43, 27], map samples from both distributions in a common feature space via adversarial training [7, 8], or translate

images between domains [22, 44, 33, 3, 21, 38, 15], are typically based on objectives that depend on both source and target samples. In our case, the independence from source samples during the adaptation procedure brings a number of advantages. The main one is that the training procedure designed for a certain target can be used *as is*, regardless of the source domain, the only difference being the model M_{θ_s} used for the first, initial label inference. Moreover, many adaptation methods require additional hyperparameters to balance different loss terms [7, 8, 21, 44, 33, 38, 3] that depend on both source and target samples. The latter is a huge drawback because in UDA we do not have target labels for hyperparameter cross-validation.

Learning with pseudo-labels. Our joint training procedure for UDA is related to the approach by Lee et al. [16]. In this work, a method for semi-supervised learning is proposed, where, as training proceeds, inference is performed on unlabeled samples, and the pseudo-labels obtained are interpreted as correct and used for training a classifier. Part of our method has similarities to this idea since, during our training procedure, we infer pseudo-labels for the target samples. However, we are different in that we use them to train a generative model.

Generative Adversarial Networks. The original formulation by Goodfellow et al. [10] is defined by the following minimax game between a network D (discriminator) and a network G (generator)

$$\min_{\theta_D}\max_{\theta_G}\mathcal{L}_{GAN} = \mathbb{E}_{x\sim p_x}[-\log D(x;\theta_D)] + \mathbb{E}_{z\sim p_z}[-\log(1-D(G(z;\theta_G);\theta_D))] \quad (1)$$

Solving such optimization problem makes D classify samples from the data distribution as *real* and samples generated by G as *fake*. Conversely, it makes G generate samples that D would classify as *real*.

A straightforward extension is the concatenation of label codes to the input before it is fed to G and D, in order to condition on the class from which data are generated. This extension is termed *conditional* GAN (cGAN, [25]), and represents the class of models this work focuses on, being our method based on class-conditioned image generation.

Several alternatives to the original GAN formulation [10] have been proposed. Two examples are substituting the cross-entropy loss with the least-squares loss [24] or with the Hinge loss [26]. Also more elaborated alternatives have been introduced [1, 18, 2]. To date, the superiority of one objective function over the others is not fully clear [23], and the main advancements on GAN research have been related to architectural choices [31] and different training procedures [49, 26, 17, 5].

3. Robustness against label noise

In this section we first formalize the problem and describe the concept of *shift noise*, as the noise resulting from inferring labels in a domain that is different from the training one. Armed with the formal definition, we explore the robustness of ConvNets [19] and cGANs against such peculiar and highly structured noise. Following standardized UDA benchmarks used by the main competing algorithms [44, 22, 3, 21, 33]—namely works that rely on GANs to perform adaptation—we train models on MNIST [20] and test on SVHN [28], MNIST-M [7] and USPS [6], we train on SVHN and test on MNIST, and we train on USPS and test on MNIST. For brevity, we define the procedure of training on source and testing on target as $source \rightarrow target$ (*e.g.*, MNIST $\rightarrow$ SVHN). The conclusions we draw in this section will motivate the algorithmic choices we will introduce in Sec. 4

3.1. Shift noise.

In the UDA setting, given a model $M_{\theta_s}(x)$, trained on a source domain $\mathcal{S} = \{x_s^{(i)}, y_s^{(i)}\}_{i=1}^n$, we can infer a pseudo-label $\tilde{y} = M_{\theta_s}(x_t)$ for each target sample x_t. Misclassification on the target set will result in a noisy set of pseudo-label associations $\mathcal{T} = \{x_t^{(i)}, \tilde{y}^{(i)}\}_{i=1}^m$.

Table 1 (first column) provides an example, associated with the split MNIST $\rightarrow$ SVHN. First of all, we can observe that misclassification noise in the confusion matrices, which we term *shift noise*, is not uniformly distributed across classes. Moreover, shift noise is also different from the structured noise analyzed by Rolnick et al. [32], where the correct label is always assumed to have the highest probability. The only *a priori* assumption we can make concerns the amount of shift noise, which must at least guarantee an accuracy higher than random chance for $M_{\theta_s}(x)$ on $\mathcal{T}$. Note that given an accuracy a for $M_{\theta_s}(x_t)$, the same accuracy can be obtained by injecting *uniform* noise in a fraction n of the labels:

$$n = (1-a)\frac{c}{c-1}, \quad (2)$$

where c is the number of classes. Vice-versa, randomizing a fraction n of the labels, one would get a fraction of correct predictions (i.e. accuracy) equal to

$$a = 1 - n\frac{c-1}{c}. \quad (3)$$

Note that randomizing a fraction n of labels does not imply they are all wrong, since, on average, $1/c$ of them will be assigned the correct class.

As already mentioned, no hypotheses on the shape of the distribution of shift noise can be put forward, making it difficult to characterize it. However, a useful estimate of its amount of structure is given by the asymmetry of the

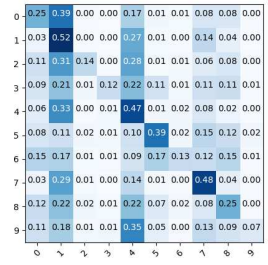
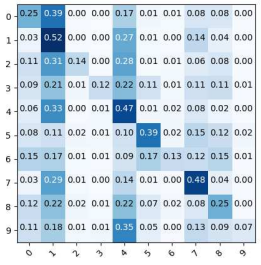
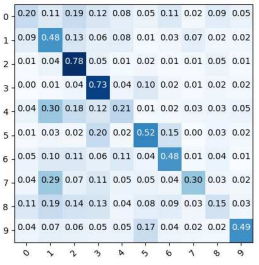
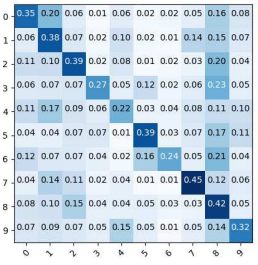
Shift noise	Classifier	GAN-test [40]	GAN-train [40]
			
$a = 0.300$ $\delta_A = 0.374$	$a = 0.321 \pm 0.002$ $\delta_A = 0.374 \pm 0.0001$	$a = 0.419 \pm 0.012$ $\delta_A = 0.252 \pm 0.012$	$a = 0.337 \pm 0.010$ $\delta_A = 0.270 \pm 0.006$

Table 1. *Left:* shift noise for the split MNIST→SVHN: only 30% of the labels are correctly inferred on SVHN after training a classifier on MNIST; high degree of asymmetry (values refer to the training sets). *Mid-left*: the confusion matrix, accuracy and δ_A for a classifier trained on noisy SVHN almost reflect the initial ones, meaning that shift noise was overfitted. *Mid-right* Oracle performance on samples generated by a cGAN trained with shift noise. Not only generated images are classified better than training samples, but also residual noise is less structured (lower δ_A). *Right*: A classifier is trained on (cleaner) cGAN-generated samples: its accuracy is slightly higher than the one of the classier directly trained on shift noise, but more importantly inferred labels show a consistently lower amount of structure in their noise. Results are averages over 5 runs, starting from the same shift noise and accuracies refers to the training sets.

	Shift noise		Classifier		Equiv. unif. noise		Classifier	
Split	a	δ_A	a	δ_A	a	δ_A	a	δ_A
SVHN → MNIST	0.669	0.208	0.660 ± 0.001	0.241 ± 0.003	0.669	0.017	0.879 ± 0.043	0.029 ± 0.006
MNIST → SVHN	0.300	0.374	0.321 ± 0.002	0.374 ± 0.000	0.300	0.157	0.293 ± 0.006	0.161 ± 0.008
MNIST → MNIST-M	0.550	0.153	0.557 ± 0.003	0.153 ± 0.000	0.550	0.014	0.619 ± 0.002	0.023 ± 0.001
USPS → MNIST	0.608	0.273	0.619 ± 0.001	0.273 ± 0.000	0.608	0.013	0.881 ± 0.036	0.026 ± 0.006
MNIST → USPS	0.819	0.150	0.807 ± 0.002	0.150 ± 0.000	0.819	0.024	0.919 ± 0.006	0.025 ± 0.001

Table 2. Classifiers tend to overfit shift noise, reflecting initial accuracy and asymmetry of the shift noise itself. They are instead significantly robust to an equivalent (in term of number of corrupted labels) amount of uniform noise n (eq. 2).

confusion matrix M, defined as:

$$\delta_A(M) = \frac{||M - M^T||_F}{2||M||_F}. \tag{4}$$

In general, $0 \leq \delta_A(M) \leq 1$. We have $\delta_A(M) = 0$ for symmetric matrices, thus $\delta_A(M) \approx 0$ for uniform noise, since M would be approximately symmetric. For shift noise, $0 < \delta_A(M) < 1$. The lower δ_A the more uniform the noise. Values for δ_A are given for all the considered benchmarks in Table 2, together with the amount of correctly inferred labels (*i.e.* accuracy).

3.2. Classifiers

As already mentioned, a study on the robustness of classifiers against label noise is proposed by Rolnick et al. [32]. We integrate their findings by training a set of classifiers with labels corrupted by shift noise. In practice, we train a first ancillary classifier $M_{\theta_s}(x)$ on a source domain $\mathcal{S}$, and then use it to assign labels $\tilde{y} = M_{\theta_s}(x_t)$ for the samples of a target domain $\mathcal{T}$. Eventually, we train a new classifier from scratch on the noisy set $\mathcal{T} = \{x_t^{(i)}, \tilde{y}^{(i)}\}_{i=1...m}$, where labels $\tilde{y}$ are corrupted by shift noise deriving from misclassifications produced by M_{θ_s}.

Classifiers tend to tolerate uniform noise in training labels to a good extent [32]. Differently, as reported in Table 2 - columns 2 and 3 - we notice that a classifier trained with shift noise in the training labels is perfectly able to (over)fit it, being accuracies on the training set and noise asymmetry in the confusion matrices nearly the same. This can also be noticed when comparing the first and second columns of Table 1. Note that this behavior does not depend on the *amount* of noise, but only on its nature. In fact, training the same classifiers with the equivalent amount of uniform noise (eq. 2) produces higher accuracies (together with, of course, nearly null δ_A), as shown in Table 2, columns 4 and 5. Note also that we train all classifiers till convergence, since we have no means for early stopping. This reflect the UDA setting were no target validation labels are provided.

Since deeper architectures, and Residual Networks [13] in particular, are more robust against uniform noise [32], one could wonder whether they could prove more resilience against shift noise than shallow models: this does not hap-

Split	Shift noise		GAN-test		GAN-train	
	a	δ_A	a	δ_A	a	δ_A
SVHN $\rightarrow$ MNIST	0.669	0.208	0.737 ± 0.018	0.134 ± 0.006	0.725 ± 0.014	0.219 ± 0.007
MNIST $\rightarrow$ SVHN	0.300	0.374	0.419 ± 0.012	0.252 ± 0.012	0.337 ± 0.010	0.270 ± 0.006
MNIST $\rightarrow$ MNIST-M	0.550	0.153	0.565 ± 0.057	0.189 ± 0.067	0.536 ± 0.092	0.174 ± 0.068
USPS $\rightarrow$ MNIST	0.608	0.273	0.772 ± 0.006	0.114 ± 0.005	0.692 ± 0.018	0.239 ± 0.009
MNIST $\rightarrow$ USPS	0.819	0.150	0.810 ± 0.005	0.135 ± 0.010	0.824 ± 0.005	0.143 ± 0.004

Table 3. cGANs trained with shift noise show robustness in generating clean samples (GAN-test), according to an oracle classifier. At the same time, they produce samples with enough variability and quality: in fact a classifier trained on generated samples outperforms a classifier trained directly on shift noise (Table 2) both in term of accuracy and noise uniformity.

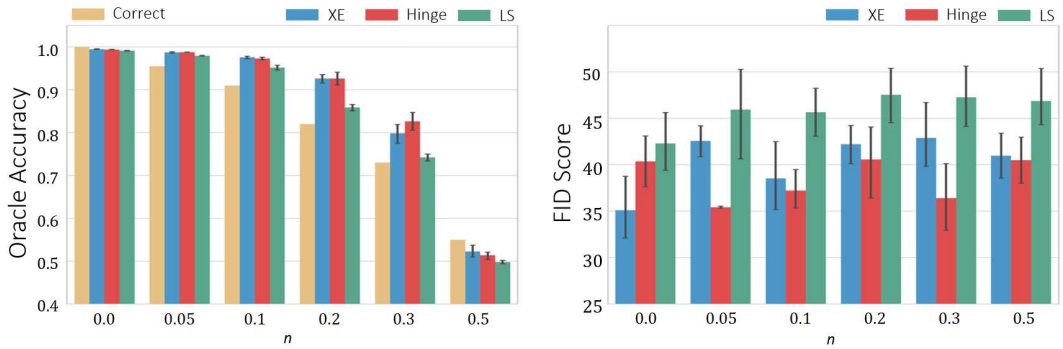


Figure 2. *Left* panel plots the fraction of images correctly generated by cGANs with different levels n of uniform noise and different objectives (*blue*: cross-entropy [10], *red*: Hinge [26], *green*: least-squares [24]), evaluated through the oracle. *Yellow* bars indicate the percentage of images with the correct label in the training set. *Right* panel shows the FID scores achieved with different levels of noise and different GAN objectives (same as *left*).

pen as we show in the experiments provided in the Supplementary Material.

3.3. cGANs

Given the success of GANs in UDA, we investigate their properties in term of robustness against both uniform and shift noise, since never investigated before.

Uniform noise. We consider the MNIST dataset [20] and assume to have an oracle to classify which class a sample belongs to. In practice, this oracle is a ConvNet trained on MNIST, that achieves $> 99\%$ accuracy on both the training and the test sets. Accuracy of such oracle on GAN-generated samples is referred to as the *GAN-test* metric [40].

One might genuinely expect that training a cGAN with, *e.g.*, a fraction of noisy labels $n = 0.1$ will result in $\sim 10\%$ of mis-generated samples. We show in the following that this does not occurr. We train the cGAN with different levels of uniform noise n and evaluate the output of the generator through the oracle, by comparing the label code given in input to the cGAN and the output of the oracle fed with the generated image.

Figure 2 *(left)* reports our findings. Yellow bars indicate the percentage of correct labels in the training set. Blue, red and green bars indicate the percentage of samples correctly generated by cGANs trained with cross-entropy [10], Hinge [26] and least-squares [24] losses, respectively. As it can be observed, when the level of noise α is reasonably below some threshold, the amount of images correctly generated (*i.e.*, correctly classified by the oracle) is always consistently higher than the amount of clean training samples, meaning that the cGAN can ignore noisy labels to some extent.

Several objectives have been proposed for the GAN formulation, which theoretically minimize different divergences between the data distribution $p_d(x)$ and the generated one $p_g(x)$. For instance, (i) the original GAN, that uses the cross-entropy loss, is proven to minimize the Jensen-Shannon divergence $D_{JS}(p_d||p_g)$ [10]; (ii) the least-squares GAN [24] is proven to minimize the Pearson [29] divergence $D_P(p_d + p_g||2p_g)$; (iii) a GAN with a Hinge loss is proved to minimize the reverse KL divergence $D_{KL}(p_g||p_d)$ [26]. In principle, being the KL divergence not symmetric, minimizing $D_{KL}(p_d||p_g)$ would place high probability everywhere the data occurs, while $D_{KL}(p_g||p_d)$ should enforce low probability wherever the data does *not* occur [9], thus yielding models more prone to mode collapse [11]. Furthermore, it has been suggested that the Pearson divergence is more resistant to outliers than the KL divergence [41, 42], and we can interpret samples with noisy label as outliers in the conditional distributions. We are thus interested in understanding how the different objectives, theoretically associated with different divergences,

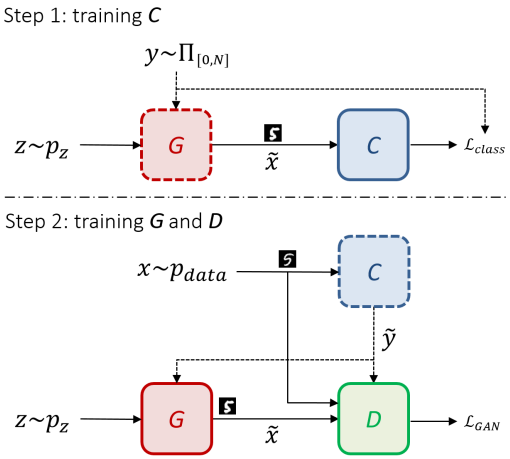


Figure 3. Graphical view of Algorithm 1. Step 1 *(top)* and step 2 *(bottom)* refer to lines $3-5$ and $6-9$, respectively. The module G and the module D are the generator and the discriminator of the cGAN. The module C is the classifier. Dashed boxes indicate frozen modules (not trained). Solid and dashed wires indicate image and label flows, respectively. $\Pi_{[0,N]}$ is the discrete uniform distribution.

behave in presence of noisy labels

There seem to be some differences between the three objectives: the least-squares GAN [24] appears to be less resistant to noisy samples. However, since there is no substantial gap between a Hinge GAN and a cross-entropy GAN, we will only use the latter from now on in this section.

Figure 2 *(right)* reports the FID scores (Fréchet Inception Distance [14]) for the same models. The FID is an indirect measure of image quality, accounting for the distance between the training and the generated distributions (the lower, the better). Interestingly, there seem to be no correlation between the amount of noisy labels and the overall quality of the images.

Shift Noise. As for classifiers, we train several cGANs on $\mathcal{T} = \{x_t^{(i)}, \tilde{y}^{(i)}\}_{i=1\dots m}$, *i.e.* we try to generate image sets starting from shift-noisy labels. In order to assess model performances, we exploit the metrics proposed by [40], *GAN-test* and *GAN-train*, which specifically deal with classifiers. As already mentioned, the GAN-test is the accuracy of an "oracle" classifier trained on real images and evaluated on generated images. This metric tries to capture the precision (i.e., image quality) of GANs. We thus train an oracle classifier for each target set and use it to test the corresponding cGAN trained with shift noise. Ground-truth labels for evaluating the oracle are those fed into the cGAN for class-conditional image generation. The GAN-train is instead the accuracy of a classifier trained on generated images and evaluated on real test images. This metric tries to capture the recall (i.e., diversity) of samples generated [40].

Accuracies and asymmetries of such classifiers are reported in Table 3, and confusion matrices are shown in Table 1. Interestingly, the samples generated by the CGANs not only induce a better accuracy on the oracle classifier, but also *significantly reduce the amount of asymmetry of the confusion matrices*. This also happens for a classifier trained on generated samples, although to a lower amount.

Algorithm 1 Pseudo-Label Refinement (PLR)

Input: target data distribution p_{target}, noise distribution p_z, pre-trained $\theta_D^0, \theta_G^0, \theta_C^0$, step sizes η, δ

Output: learned weights $\theta_D, \theta_G, \theta_C$

1: **Initialize:** $\theta_D \leftarrow \theta_D^0$, $\theta_G \leftarrow \theta_G^0$, $\theta_C \leftarrow \theta_C^0$
2: **while** not done **do**
3: Sample $z \sim p_z$ and $y \sim \Pi_{[0,N]}$
4: Generate $\tilde{x} = G(z|y)$
5: $\theta_C \leftarrow \theta_C - \eta \nabla_{\theta_C} \mathcal{L}_{class}(\theta_C; \tilde{x}, y)$
6: Sample $x \sim p_{target}$ and $z \sim p_z$
7: Infer $\tilde{y} = C(x)$
8: $\theta_D \leftarrow \theta_D - \delta \nabla_{\theta_D} \mathcal{L}_{GAN}(\theta_D; z, x, \tilde{y})$
9: $\theta_G \leftarrow \theta_G - \delta \nabla_{\theta_G} \mathcal{L}_{GAN}(\theta_G; z, \tilde{y})$

Summary. In conclusion, training a cGAN on the set $\mathcal{T} = \{x_t^{(i)}, \tilde{y}^{(i)}\}_{i=1\dots m}$ allows to "filter" noise in $\tilde{y}^{(i)}$ in two respects: *i)* by reducing the amount of shift noise in the generated data and *ii)* by reducing the asymmetry of shift noise, making its distribution more alike uniform noise and thus more tolerable for classifiers [32]. As a matter of fact, Tables 1 and 3 show that classifiers trained on generated samples (GAN-train column) perform better than classifiers trained with shift noise (Classifier column).

4. Application to UDA

In this section, we detail the method designed to face UDA, based on the insights and the findings reported so far.

As already mentioned, we can interpret data from the target distribution p_{target}, with pseudo-labels (inferred through a classifier trained on data from the source distribution p_{source}), as a dataset polluted with label noise. From this perspective, training a cGAN on such empirical, noisy distribution should allow us to generate cleaner samples, as suggested by the findings reported in Section 3. In turn, a classifier trained on generated data will perform better than the one trained on source data, since noise in the target labels has been reduced in both amount and asymmetry. Starting from these two insights, we define a training procedure where we simultaneously train a classifier C and the modules G and D that define a cGAN (see Figure 1).

The pre-train step of our method consists in training a model M_{θ_s} on labeled data from the source distribution

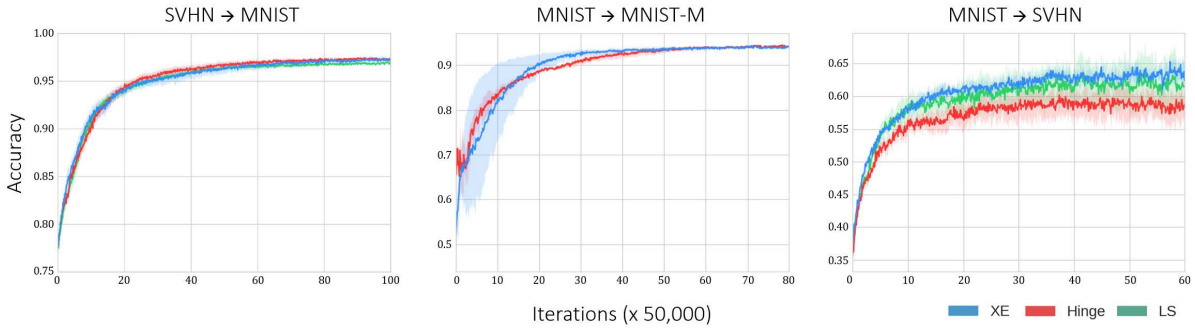


Figure 4. Evolution of the accuracy on target test sets for SVHN → MNIST, MNIST → MNIST-M and MNIST → SVHN (from *left* to *right*), computed throughout the training procedure described in Algorithm 1. *Blue*, *red* and *green* curves are associated with GANs trained with the cross-entropy loss [10], the Hinge loss [26] and the least-squares loss [24], respectively. Results obtained with the least-squares loss are not reported for the MNIST → MNIST-M as they are significantly worse than the ones achieved with the other options. Curves are averaged over three different runs, shades represent the confidence bands.

$$\min_{\theta_C} \mathcal{L}_{class} \coloneqq \mathbb{E}_{x,y\sim p_{source}} H(x, y; \theta_s), \tag{5}$$

where H is the cross-entropy loss. Equipped with this classifier, we can straightforwardly infer pseudo-labels for each target sample as $\tilde{y}_t = C(x_t)$. Typically, an unknown percentage of these labels will be wrong, due to the domain shift between p_{source} and p_{target}. We obviously do not know which labels are correct and which are not, but this is irrelevant for the devised strategy.

Before starting the joint training procedure, we also need to train a cGAN on the noisy target distribution, as in the previous section. This is necessary because we will train C on generated data, and thus starting with randomly-initialized G and D would result in a random classifier C, and consequently in non-informative pseudo-labels.

We train the cGAN in a standard fashion, alternating between the following minimax game. Note that we report here the objective as defined in Goodfellow et al. [10], but in our experiments we also test least-squares GANs [24] and Hinge-GANs [26].

$$\begin{aligned}\min_{\theta_D}\max_{\theta_G} \mathcal{L}_{GAN} \coloneqq \mathbb{E}_{x,\tilde{y}\sim p_{target}}[-\log D(x|\tilde{y})] \\ + \mathbb{E}_{z\sim p_z}[-\log(1 - D(G(z|\tilde{y})|\tilde{y}))]\end{aligned} \tag{6}$$

Armed with the pre-trained modules C, G and D, we can start the training procedure, which is defined by Algorithm 1. In short, we alternate until convergence between (i) updating the weights of the classifier θ_c via stochastic gradient descent, with labeled target samples uniformly generated via G (lines $3-5$), and (ii) training the weights of the discriminator θ_D and of the generator θ_G via stochastic gradient descent, with target samples from p_{target}, with pseudo-labels inferred via the classifier C (lines $6-9$). Alternating the two steps will progressively reduce the amount and asymmetry of the initial shift noise in the target set.

In Figure 3, we show the computation flow of the proposed system: *top* (step 1) and *bottom* (step 2) panels represent modules corresponding to lines $3-5$ and $6-9$ of Algorithm 1, respectively.

The output of Algorithm 1 is twofold: (i) the trained modules of the cGAN (G and D), and (ii) the trained classifier C, which is the module finally used to classify target samples. In the next section, we report performances obtained using C in UDA benchmarks.

5. Experiments

We test Algorithm 1 on a variety of UDA benchmarks. In every experiment, we run Algorithm 1 until convergence, intended as convergence of the cGAN minimax game, and use accuracy on target dataset test sets (fed to the classifier C) as a metric to evaluate our models and compare them with other adaptation approaches.

Benchmarks. We test our method on the following cross-dataset digit classification problems: SVHN ↔ MNIST, MNIST → MNIST-M and USPS ↔ MNIST, following protocols on which UDA algorithms based on GANs are tested [22, 44, 3, 33, 38]. In order to work with comparable sizes, we resized all images to 32×32. For each experiment, we use a ConvNet with architecture *conv-pool-conv-pool-fc-fc-softmax*. For the GAN architectures, we draw inspiration from DCGAN [31], though considering different objectives (cross-entropy, least-squares, Hinge). All details regarding architectures and training procedures are reported in the Supplementary Material.

5.1. Results

We report in Figure 4 the plots showing the evolution of test accuracy in different experiments throughout the training procedure defined by Algorithm 1. As it can be observed, the performance on target domain of the classifier trained on target samples generated via the cGAN is improved over iterations. It is worth highlighting the mono-

	SVHN → MNIST	MNIST → SVHN	MNIST → MNIST-M	USPS → MNIST	MNIST → USPS
Train on source	0.682	0.314	0.548	0.612	0.783
DANN [7]	0.739	-	0.767	-	-
ADDA [47]	0.760 ± 0.018	-	-	0.901 ± 0.008	0.894 ± 0.002
DIFA [48]	0.897 ± 0.020	-	-	0.897 ± 0.005	0.962 ± 0.002
MECA [27]	0.952	-	-	-	-
ATT [35]	0.862	0.528	0.942	-	-
AD [36]	0.950 ± 0.187	-	-	0.931 ± 0.127	0.961 ± 0.029
MCD [37]	0.962 ± 0.004	-	-	0.941 ± 0.003	0.965 ± 0.003
CoGAN [22]	-	-	-	0.931 [21]	0.957 [21]
DTN* [44]	0.849	-	-	-	-
UNIT* [21]	0.905	-	-	0.936	0.960
PixelDA** [3]	-	-	0.982	-	0.959
SBADA** [33]	0.761	0.611	0.994	0.950	0.976
GenToAd [38]	0.924 ± 0.009	-	-	0.908 ± 0.013	0.953 ± 0.007
CycADA [15]	0.904 ± 0.004	-	-	0.965 ± 0.001	0.956 ± 0.002
Ours (PLR)					
Cross-entropy	0.973 ± 0.006	0.634 ± 0.026	0.943 ± 0.002	0.918 ± 0.013	0.893 ± 0.019
Least-squares	0.969 ± 0.003	0.618 ± 0.060	-	0.916 ± 0.019	0.903 ± 0.013
Hinge	0.973 ± 0.003	0.586 ± 0.041	0.938 ± 0.002	0.891 ± 0.010	0.907 ± 0.022
Train on target	0.992	0.913	0.964	0.992	0.999

Table 4. Comparison between our method (Pseudo-Label Refinement - PLR) with different GAN objectives and competing algorithms. Test-set accuracies are the results of averaging over 3 different runs. (*) Uses extra SVHN data ($531,131$ images). (**) Uses $1,000$ target samples for cross-validation.

tonic increase of performance: early stopping is not feasible in UDA, thus an unstable algorithm is of scarce utility.

A particularly important result in the one related to the MNIST → SVHN split. The large gap between the two domains, and the fact that labels are provided for the easier, more biased dataset makes this split particularly difficult to tackle [7, 8]. Our method allows to generate SVHN samples that make the classifier C – trained on them – better generalizing to the target distribution, improving performance of $\sim 30\%$ with respect to the baseline. The complete analysis of the obtained results, also in comparison with the state-of-the-art methods, is illustrated in the following.

Comparison with other methods. Table 4 compares the proposed method performance (Pseudo-Label Refinement, PLR) with the results obtained by several works in the literature. It is worth to note that, nowadays, research in UDA reached a point where it is difficult to state the superiority of a method over the others. Indeed, Table 4 shows that there is not a single method that performs better than the others in *every* benchmark.

First, our method shows performance comparable with the state of the art in the SVHN → MNIST split benchmark, significantly outperforming more complex image-to-image translation methods [44, 21, 33, 38] that not only rely on more complicated architectures, but also present a training procedure where the objective is weighted by different hyperparameters (which, as previously mentioned, is a significant drawback in UDA).

Next, an important result is the one related to MNIST → SVHN. As discussed above, this is a rather challenging split, and several methods (*e.g.*, [7, 47, 48, 27, 22, 21, 44]) do not show results on this benchmark. Furthermore, we tested the implementation of PixelDA [3] provided by the authors and could not observe any sign of adaptation. Our algorithm, with the cross-entropy loss as GAN objective, is the best performing method by a statistically significant margin. We also stress that Russo et al. [33], the second best performing method on this split, use $1,000$ samples from SVHN to cross-validate the hyperparameters, thus making the working setup much easier than ours.

On MNIST → MNIST-M, the performance achieved with our method is comparable with Saito et al. [35] and below the one achieved by methods that perform hyperparameter cross-validation [3, 33].

6. Conclusion

We introduce the concept of shift noise and analyze the robustness of classifiers and cGANs against such highly structured noise. We empirically show that, while classifiers are generally not robust against this kind of label noise, cGANs are more resilient against it, and furthermore generate samples with a more uniform noise distribution. Inspired by these findings, we design a training procedure that progressively allows to generate cleaner samples from the target distributions, and in turn to train better classifiers.

For future work, we hope to extend the devised algorithm to more realistic UDA benchmarks, such as Office-31 [34] and VisDA [30]. The limitation towards this goal is the current computational expense in trainig GANs that generate high-resolution samples [17, 5].

References

[1] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. 3

[2] D. Berthelot, T. Schumm, and L. Metz. BEGAN: boundary equilibrium generative adversarial networks. *CoRR*, abs/1703.10717, 2017. 3

[3] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2, 3, 7, 8

[4] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. Domain separation networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 343–351. Curran Associates, Inc., 2016. 2

[5] A. Brock, J. Donahue, and K. Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019. 3, 8

[6] J. S. Denker, W. R. Gardner, H. P. Graf, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, H. S. Baird, and I. Guyon. Advances in neural information processing systems 1. chapter Neural Network Recognizer for Handwritten Zip Code Digits, pages 323–331. 1989. 3

[7] Y. Ganin and V. S. Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 1180–1189, 2015. 2, 3, 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. *J. Mach. Learn. Res.*, 17(1), Jan. 2016. 2, 3, 8

[9] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org. 5

[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, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014. 2, 3, 5, 7

[11] I. J. Goodfellow. NIPS 2016 tutorial: Generative adversarial networks. *CoRR*, abs/1701.00160, 2017. 5

[12] P. Haeusser, T. Frerix, A. Mordvintsev, and D. Cremers. Associative domain adaptation. In *International Conference on Computer Vision (ICCV)*, 2017. 2

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 4

[14] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6626–6637. Curran Associates, Inc., 2017. 6

[15] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell. CyCADA: Cycle-consistent adversarial domain adaptation. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1989–1998, Stockholmsmssan, Stockholm Sweden, 10–15 Jul 2018. PMLR. 2, 3, 8

[16] D. hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*, 2013. 3

[17] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018. 3, 8

[18] N. Kodali, J. D. Abernethy, J. Hays, and Z. Kira. How to train your DRAGAN. *CoRR*, abs/1705.07215, 2017. 3

[19] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, 1(4):541–551, Dec. 1989. 3

[20] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998. 3, 5

[21] M. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. *CoRR*, abs/1703.00848, 2017. 2, 3, 8

[22] M.-Y. Liu and O. Tuzel. Coupled generative adversarial networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 469–477. Curran Associates, Inc., 2016. 2, 3, 7, 8

[23] M. Lucic, K. Kurach, M. Michalski, S. Gelly, and O. Bousquet. Are gans created equal? A large-scale study. *CoRR*, abs/1711.10337, 2017. 3

[24] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, and Z. Wang. Multi-class generative adversarial networks with the L2 loss function. *CoRR*, abs/1611.04076, 2016. 3, 5, 6, 7

[25] M. Mirza and S. Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014. 2, 3

[26] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018. 3, 5, 7

[27] P. Morerio, J. Cavazza, and V. Murino. Minimal-entropy correlation alignment for unsupervised deep domain adaptation. *International Conference on Learning Representations*, 2018. 2, 8

[28] 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*, 2011. 3

[29] K. Pearson. *On the Criterion that a Given System of Deviations from the Probable in the Case of a Correlated System of Variables is Such that it Can be Reasonably Supposed to have Arisen from Random Sampling*, pages 11–28. Springer New York, New York, NY, 1992. 5

[30] X. Peng, B. Usman, N. Kaushik, J. Hoffman, D. Wang, and K. Saenko. Visda: The visual domain adaptation challenge. *CoRR*, abs/1710.06924, 2017. 8

[31] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016. 3, 7

[32] D. Rolnick, A. Veit, S. J. Belongie, and N. Shavit. Deep learning is robust to massive label noise. *CoRR*, abs/1705.10694, 2017. 1, 2, 3, 4, 6

[33] P. Russo, F. M. Carlucci, T. Tommasi, and B. Caputo. From source to target and back: Symmetric bi-directional adaptive gan. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2, 3, 7, 8

[34] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *Proceedings of the 11th European Conference on Computer Vision: Part IV*, ECCV'10, pages 213–226, Berlin, Heidelberg, 2010. Springer-Verlag. 8

[35] K. Saito, Y. Ushiku, and T. Harada. Asymmetric tri-training for unsupervised domain adaptation. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 2988–2997, 2017. 2, 8

[36] K. Saito, Y. Ushiku, T. Harada, and K. Saenko. Adversarial dropout regularization. In *International Conference on Learning Representations*, 2018. 2, 8

[37] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2, 8

[38] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2, 3, 7, 8

[39] O. Sener, H. O. Song, A. Saxena, and S. Savarese. Learning transferrable representations for unsupervised domain adaptation. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2110–2118. Curran Associates, Inc., 2016. 2

[40] K. Shmelkov, C. Schmid, and K. Alahari. How good is my gan? In *The European Conference on Computer Vision (ECCV)*, September 2018. 4, 5, 6

[41] M. Sugiyama, S. Liu, M. C. du Plessis, M. Yamanaka, M. Yamada, T. Suzuki, and T. Kanamori. Direct divergence approximation between probability distributions and its applications in machine learning. *JCSE*, 7(2):99–111, 2013. 5

[42] M. Sugiyama, T. Suzuki, and T. Kanamori. Density-ratio matching under the bregman divergence: a unified framework of density-ratio estimation. *Annals of the Institute of Statistical Mathematics*, 64(5):1009–1044, Oct 2012. 5

[43] B. Sun and K. Saenko. Deep CORAL: correlation alignment for deep domain adaptation. In *Computer Vision - ECCV 2016 Workshops - Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III*, pages 443–450, 2016. 2

[44] Y. Taigman, A. Polyak, and L. Wolf. Unsupervised cross-domain image generation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017. 2, 3, 7, 8

[45] K. K. Thekumparampil, A. Khetan, Z. Lin, and S. Oh. Robustness of conditional gans to noisy labels. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 10271–10282. Curran Associates, Inc., 2018. 2

[46] A. Torralba and A. A. Efros. Unbiased look at dataset bias. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, pages 1521–1528, Washington, DC, USA, 2011. IEEE Computer Society. 1

[47] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2, 8

[48] R. Volpi, P. Morerio, S. Savarese, and V. Murino. Adversarial feature augmentation for unsupervised domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2, 8

[49] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017. 3