Generative Adversarial Guided Learning for Domain Adaptation

Kai-Ya Wei
jakc4103@gmail.com
Chiou-Ting Hsu
cthsu@cs.nthu.edu.tw

Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan

Abstract

This paper focuses on unsupervised domain adaptation problem, which aims to learn a classification model on an unlabelled target domain by referring to a fully-labelled source domain. Our goal is twofold: bridging the gap between source-target domains, and deriving a discriminative model for the target domain. We propose a Generative Adversarial Guided Learning (GAGL) model to tackle the task. To minimize the source-target domain shift, we adopt the idea of domain adversarial training to build a classification network. Next, to derive a target discriminative classifier, we propose to include a generative network to guide the classifier so as to push its decision boundaries away from high density area of target domain. The proposed GAGL model is an end-to-end framework and thus can simultaneously learn the classification model and refine its decision boundary under the guidance of the generator. Our experimental results show that the proposed GAGL model not only outperforms the baseline domain adversarial model but also achieves competitive results with state-of-the-art methods on standard benchmarks.

1 Introduction

Although deep neural networks have brought impressive achievements in many tasks, their success heavily relies on large amounts of labelled data. In many real-world applications, collecting fully-labelled data for the task of interest is not only costly but also very time-consuming. Many efforts have been undertaken to alleviate the heavy burden of data labelling. One of the promising methods is through domain adaptation, which aims to transfer knowledge learnt from a fully-labelled source domain into a target domain. For example, we can synthesize labelled synthetic or semi-synthetic data (source domain) to learn the model, and then adopt the learnt model to another set of real-world data (target domain). Nevertheless, once the characteristics of source data are different from those of the target data, a model trained in the source domain can hardly perform well on the target domain. This problem is often referred to as the domain shift problem. Since the goal of domain adaptation is to learn a model that can generalize well on the target domain, the domain shift problem is indeed a key issue in domain adaptation.

Among different settings of domain adaptation, we focus on the most challenging case - unsupervised domain adaptation (UDA) problem. In the UDA setting, the target data are unlabelled; in other words, for the target domain data, we are provided with the data themselves but have absolutely no information about their ground-truth labels. Therefore, many

© 2018. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.

methods have been developed, e.g., to match distributions of the two domains or to learn a domain-invariant representation.

Several recent methods [8, 18, 31, 35] proposed to combine the feature distribution matching and the model learning in a unified framework. In particular, in [8], the authors proposed using domain adversarial training to learn domain-invariant representation; their results suggested that a successful adaptation tends to occur when both the source generalization error and the feature discrepancy distance are small. However, even under the two conditions, we argue that it is insufficient to guarantee a well-generalized model in the target domain. According to the theoretical analysis of domain adaptation [2], the expected loss of the target domain is bounded by three terms: (1) the expected loss in source domain; (2) the divergence between the source and target distributions; and (3) the expected difference in labelling functions across the two domains. These methods [8, 18, 31, 35] focused on minimizing the first and the second terms, while considering the third term of little account and can be ignored. Unfortunately, this assumption is only valid when the learnt representation is discriminative in the target domain. Once the representation is indiscriminative in the target domain, the decision boundary of classification model may locate across high density area of target-domain data. This undesirable situation usually results in significant loss in the third term and leads to poor performance in the target domain.

To tackle the above-mentioned problem, we need to ensure that the learnt model is also target discriminative in addition to domain-invariant. In [24], the authors also focused on learning a target discriminative model and proposed an asymmetric tri-training to conduct the task. Unlike [24], which can be seen as replacing the domain adversarial training [8] with the asymmetric tri-training, we mean to directly control the decision boundary of the classification model in domain adversarial training. The key idea underlying the proposed model comes from the cluster assumption [3], which assumes that the data distribution consists of several clusters and that data in the same cluster are likely to originate from the same class. Therefore, even without the target domain labels, we can still move the decision boundaries away from high density area and push them toward low density ones. In this paper, we propose to incorporate the cluster assumption with the GAN-based classification model [6, 15, 25] to relocate the decision boundaries. Note that, although the idea of moving decision boundaries is similar to [29], we define a novel loss term to conduct the task and will show that, through the GAN-based classification model, the generated fake data can serve as an effective guide to control the decision boundary in the UDA task.

To sum up, we propose an unsupervised Generative Adversarial Guided Learning (GAGL) model to learn a domain-invariant and target-discriminative classification model. The proposed GAGL has two main objectives. Firstly, we learn a domain-invariant classification model through domain adversarial training. Secondly, we propose to incorporate an additional generative model to further push the decision boundary of the classifier toward low density areas. The proposed GAGL is an end-to-end framework that simultaneously learns and refines the classification model via generative adversarial guided learning.

Our contributions are summarized as follows:

- We propose to incorporate a generative model to guide the classification model in domain adversarial training and obtain better classification performance.
- The proposed GAGL drives a classification model that is not only invariant to source-target domain shift but also generalizes well in the target domain.
- Our experimental results verify that the proposed GAGL model achieves competitive results to state-of-the-arts on standard UDA benchmarks.

2 Related Work

Learning a domain-invariant representation is crucial to UDA task. The idea of matching feature representation of two distributions has been extensively studied and has shown promising results. For example, the methods in [8, 18, 28, 31, 32, 35] minimized the domain divergence in terms of either maximum mean discrepancy (MMD) [10, 18, 31], domain discriminator [8, 32], central moment discrepancy (CMD) [35], or Wasserstein distance [28]. However, these distribution-matching-based methods did not explicitly incorporate the target discriminative issue into the objective functions.

Several methods [12, 23, 26, 34] proposed to incorporate generative adversarial network (GAN) [9] to tackle the UDA task. [12, 23] proposed using two GAN models to learn the mapping functions between source and target domains. For each round trip mapping, e.g., $source \rightarrow target \rightarrow source$, a cyclic-consistency loss is enforced to ensure a correct mapping. In [26], a domain-invariant embedding space is learnt by an additional GAN model. In [34], the authors further proposed to learn the domain-invariant representation by adversarial feature augmentation. The aforementioned GAN-based UDA methods adopt the generative model as a mapping function to tackle the domain shift problem. However, as these methods include no explicit constraint on the discriminative capability of task-specific classifier, the learnt representation does not necessarily guarantee to be target discriminative.

Meanwhile, [11, 24, 27] proposed to replace distribution matching by either similarity-based embedding [11], asymmetric tri-training [24], or nearest-neighbor-based classifier [27]. These methods have shown that both domain-invariant and target discriminative representation are crucial to a well-performed classifier in target domain.

Recently, the cluster assumption [3], which indicates that the decision boundaries of classifier should lie on low density region in data manifold, has led to considerable success on semi-supervised learning (SSL) [5, 15, 16, 19, 25] and has also been applied to domain adaptation [6, 7, 29]. In [29], the locally-Lipschitz constrained conditional entropy loss is proposed to encourage the model to be discriminative on target task. In [6, 7], extensive data augmentation is applied to learn a more stable decision boundary in classification model. Instead of relying on conditional entropy loss [29] or applying various data augmentation [6, 7], we aim to extend the idea from GAN-based SSL [5, 15, 25] into UDA task.

3 Proposed Method

We first define the notations and describe the unsupervised domain adaption problem in Sec.3.1. Next, in Sec.3.2, we briefly review the idea of domain adversarial training [8]. In Sec.3.3, we present our proposed method by incorporating a generative network into the domain adversarial training so as to derive a target discriminative model.

3.1 Problem Statement and Notations

Let X be the input space (e.g. images) and Y be the output space (e.g. image categories) with K categories. In the unsupervised domain adaption (UDA) scenario, the source distribution $S(x,y)$ and target distribution $T(x,y)$ on $X \otimes Y$, where $x \in X$ and $y \in Y$, are assumed to be complex, unknown, and differed with certain domain shift. During the training stage, in the source domain, a set of n data and their labels $(X^S, Y^S) = \{(x_1^s, y_1^s), (x_2^s, y_2^s), ..., (x_n^s, y_n^s)\}$ are given; whereas, in the target domain, only m data $X^T = \{x_1^t, x_2^t, ..., x_m^t\}$ are given but

without their ground-truth labels. The goal of UDA is to learn a classification model D that generalizes well on the target data X^T in the testing phase.

Figure 1 shows the flowchart of the proposed Generative Adversarial Guided Learning (GAGL) model, where D denotes the classification model and G denotes the generative network that takes a noisy vector $z \in \mathbb{R}^w$ as its input and outputs the generated data X^G.

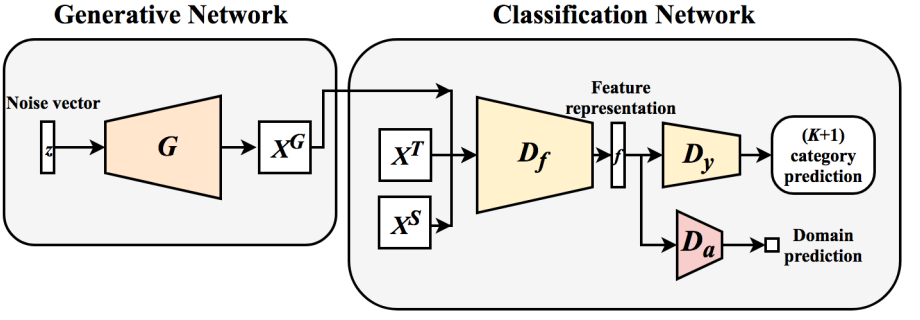


Figure 1: Flowchart of the Generative Adversarial Guided Learning model.

3.2 Review of Domain Adversarial Training

We now give a brief review of domain adversarial training [8]. In [8], the authors built the classification model D upon a deep feed-forward architecture and decomposed D into three parts: a feature extractor D_f, a label predictor D_y, and a domain classifier D_a, as illustrated in Figure 2 (a). The feature extractor D_f, parameterized by θ_f, maps an input x into a high-dimensional feature representation $\boldsymbol{f} \in \mathbb{R}^{h \times w \times c}$, i.e. $\boldsymbol{f} = D_f(x, \theta_f)$. The label predictor D_y, parameterized by θ_y, then maps the feature $\boldsymbol{f}$ into the output category label. The domain classifier D_a, parameterized by θ_a, maps $\boldsymbol{f}$ into the domain label $d \in \{0, 1\}$ so as to distinguish whether the input x is drawn from the the target domain X^T or the source domain X^S.

Thus, to learn domain-invariant representation, one needs to optimize the label prediction loss L_y as well as the domain prediction loss L_a. In other words, the label prediction for the source domain should be as accurate as possible; whereas the domain prediction should try to match the distributions of the two domains to be as similar as possible. The adversarial loss is then defined by combining the two loss functions with a hyper-parameter λ_a by

$$L_{adv}(X^S, Y^S, X^T) = L_y(X^S, Y^S) + \lambda_a L_a(X^S, X^T). \tag{1}$$

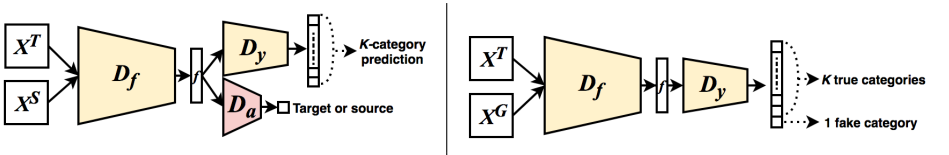


Figure 2: (a) The domain adversarial training. Here, the gradient reversal layer $R(\cdot)$ [8] is omitted for simplicity. (b) The (K+1)-category prediction, with an additional "fake" category, in the proposed GAGL .

3.3 Generative Adversarial Guided Learning

As mentioned before, because domain adversarial training [8] includes no explicit formulation for the task-discriminative issue, there is no guarantee that the learnt model can well generalize to the target domain. In this subsection, we describe how we incorporate a generative model G to derive a target discriminative model D.

Similar to [8], we also decompose the classification model D into a feature extractor D_f, a label predictor D_y, and a domain classifier D_a. Let $P_y(x^s) = D_y(D_f(x^s,\theta_f),\theta_y)$ denote the function that outputs the K-dimensional softmax prediction for a given input x^s, and $P_a(x^*) = D_a(R(D_f(x^*,\theta_f)),\theta_a)$ denote the function that outputs the domain prediction for an input x^*, where $* \in \{s,t\}$ and $R(\cdot)$ is the gradient reversal layer [8]. We then define the label prediction loss L_y and the domain prediction loss L_a as follows,

$$L_y(X^S,Y^S) = -\frac{1}{n}\sum_{i=1}^{n} y_i^s \log[P_y(x_i^s)], \tag{2}$$

$$L_a(X^S,X^T) = -\Big\{\frac{1}{n}\sum_{i=1}^{n}\log[P_a(x_i^s)] + \frac{1}{m}\sum_{j=1}^{m}\log[1-P_a(x_i^t)]\Big\}. \tag{3}$$

Next, we propose to include a generator G to guide the decision boundary of D. The goal of G is to generate realistic data X^G that the discriminator cannot distinguish from real ones X^T. As suggested in [25], we combine the real-fake discriminator into the K-category classification model D, and have the (K+1)-category classification model D with an additional "fake" class. Figure 2 (b) illustrates this (K+1)-category classification model. When given the unlabelled target domain data X^T and the generated data X^G, we expect the classification model D to be able to determine whether an input sample is real or fake. We thus define the unlabelled guide loss L_u between unlabelled target data and the generated data as

$$L_u(X^T,X^G) = -\Big\{\frac{1}{m}\sum_{i=1}^{m}\log[P_y^{K+1}(x_i^g)] + \frac{1}{m}\sum_{j=1}^{m}\log[1-P_y^{K+1}(x_j^t)]\Big\}, \tag{4}$$

where $P_y^{K+1}(x^*) = D_y(D_f(x^*,\theta_f),\theta_y)$ is the function that outputs the prediction of x^*, $* \in \{t,g\}$, on the (K+1)-th category (i.e., the fake class). Minimizing the unlabelled guide loss L_u will consequently enforce X^G and X^T to be predicted as the first K categories (i.e., the true classes) to be low and high, respectively. Therefore, inclusion of L_u encourages the decision boundary of classification model to lie between X^G and X^T in data manifold and to be away from X^T. During the GAN-based learning process, the randomly generated data X^G keep guiding the classification model to refine its decision boundary. The classification model, whose decision boundary becomes smoother and is gradually pushed away from the high density area of target data, finally leads to a target discriminative model. Note that, because the (K+1) form of classifier is over-parameterized [25], we thus set the (K+1)-th output before soft-max operation to be a zero vector and lead back to a K-category classification model. With two hyper-parameters λ_a and λ_u, we formulate the loss function of the classification model D by

$$L_D(X^S,Y^S,X^T,X^G) = L_y(X^S,Y^S) + \lambda_a L_a(X^S,X^T) + \lambda_u L_u(X^T,X^G). \tag{5}$$

As to the loss function L_G of the generator G, instead of directly maximizing L_u, we use a more stable optimization method introduced in [25] to define L_G. Let $\phi(x^*)$ be the output

feature representation before the last dense layer of D for a given input x^*, $* \in \{t, g\}$. The loss function L_G is defined as

$$L_G(z) = Dist(\phi(X^T), \phi(X^G)), \tag{6}$$

where $Dist(\cdot,\cdot)$ is a distribution matching criterion. In this paper, we use CMD [35] to calculate $Dist(\cdot,\cdot)$.

To train the network, we alternatively update the classifier D and the generator G by optimizing Eq.5 and Eq.6. Note that, our experiments show that the domain adversarial training of D using gradient reversal layer [8] is unstable and hyper-parameter sensitive. Hence, we follow [9, 29] to update D_f and D_a alternatively instead.

4 Experiments

4.1 Domain adaptation benchmarks

We evaluate and compare with existing methods on the following benchmarks, under different settings, where **source** $\rightarrow$ **target** indicates that we use the dataset **source** as the source domain for training and use the dataset **target** as the target domain for testing.

SVHN $\leftrightarrow$ **MNIST.** The street view house number (SVHN) dataset [21] consists of 73257 training images and 26032 testing images. We use the variant *Format 2*, where each image, though cropped, still contains multiple digits. MNIST [17] contains 55000 training images and 10000 testing images. Because MNIST images are all in grey scale, we extend the MNIST images to three color channels by simply replicating the gray-scale values to three channels to match color images in other datasets. We conduct the domain adaptation on both directions (i.e., SVHN $\rightarrow$ MNIST and MNIST $\rightarrow$ SVHN). Note that, because images in SVHN are more diverse, the task of using MNIST as the source domain (i.e., MNIST $\rightarrow$ SVHN) is far more challenging than using SVHN as the source domain (i.e., SVHN $\rightarrow$ MNIST).

MNIST $\rightarrow$ **MNIST-M.** In this task, we use MNIST [17] as the source data for training and test on MNIST-M [8]. The MNIST-M [8] is generated by blending grey scale MNIST images with color background patches from BSD500 dataset [1]. The resulting color images in MNIST-M thus enlarge the domain shift and are much difficult for automatic classification.

Syn-Num $\rightarrow$ **SVHN.** The synthetic number (Syn-Num) dataset, which consists of 500000 training images, was introduced in [8]. In Syn-Num, the images are generated by varying the fonts and properties (including position, orientation, background, stroke colors, and the amount of blur) of digits to stimulate SVHN.

Syn-Signs $\rightarrow$ **GTSRB.** The synthetic signs (Syn-Signs) dataset is provided by [20] and contains 100000 training images generated by applying various manual transformations to traffic sign images. The German Traffic Signs Recognition Benchmark (GTSRB) [30] consists of 39209 training images and 12630 testing images. Because the GTSRB was originally provided for recognition task, we follow the same setting in most previous work to resize the cropped region of interest (RoI) in each image into the same size as Syn-Signs images. Both the two tasks Syn-Signs $\rightarrow$ GTSRB and Syn-Num $\rightarrow$ SVHN use synthetic data as the source data and real data as the target domain; but the adaptation of Syn-Signs $\rightarrow$ GTSRB is more complex because it involves 43 categories rather than 10 categories.

STL $\leftrightarrow$ **CIFAR.** Both STL-10 [4] and CIFAR-10 [14] are 10-category image datasets. Here,

Loss function	Source / Target	SVHN / MNIST	MNIST / SVHN	Syn-Num / SVHN	MNIST / MNIST-M	CIFAR / STL	STL / CIFAR	Syn-Signs / GTSRB
L_y (DNN-SO)		85.7	41.0	90.1	64.4	75.3	56.7	95.5
$L_y + L_a$ (DANN-O)		86.0	61.5	88.7	92.9	76.2	58.0	94.0
$L_y + L_a + Cond.$ (DANN-C)		**97.1**	*64.2*	91.5	*93.5*	76.3	58.5	*96.9*
L_D/L_{G-FM} (GAGL-FM)		95.3	63.3	*92.5*	90.3	**78.0**	**65.4**	*96.9*
L_D/L_G (GAGL)		*96.7*	**74.6**	**93.1**	**94.9**	*77.0*	*61.5*	**97.6**

Table 1: Evaluation of loss functions. We show the classification accuracy of test set on seven domain adaptation benchmarks.

we follow the setting in previous work to remove the non-overlapping "monkey" and "frog" categories from STL and CIFAR, respectively, and have a 9-category domain adaptation task.

4.2 Implementation Details

CNN architecture. To have a fair comparison with [29], we adopt the same architecture of classification model D as in [29]. We use large CNN for STL $\leftrightarrow$ CIFAR tasks and use small CNN for the other adaptation tasks. All the input images are resized to 32×32 using bilinear interpolation. As to the generative model G, we use 1 dense layer followed by 3 transpose convolution layers. We use leaky-ReLU as non-linearity and apply batch normalization before non-linearity on each layer (including convolution and dense layers) of D and G before non-linearity, except for the domain discriminator D_a, where we use ReLU as non-linearity and apply no batch normalization. More details of our CNN architecture is provided in the supplementary file.

Data pre-processing. The only pre-processing we apply is instance normalization. As suggested in [29], by introducing two learnable parameters γ and β, instance normalization is served as a learnable layer before the first convolution layer in the classification model.

Hyper-parameters. For each task, we randomly select 1000 samples from the target domain to form a validation set, and then use the validation set to tune the hyper-parameters (λ_a, λ_u). The ranges of hyper-parameters are set as in $(10^{-2}, 0)$ and $(10^{-1}, 1)$ for λ_a and λ_u, respectively. We apply Adam optimizer [13] for both the generative and classification models with learning rate equal to 10^{-3}, $\beta_1 = 0.5$, and $\beta_2 = 0.999$. All tasks are trained for no more than 80000 iterations.

Exponential moving average of model parameters. As suggested in [22], using the averaged model parameters over the training process tends to result in a better model than directly using the parameters obtained at the last iteration. We thus keep the model parameters θ_t of D at each iteration and then use their exponential moving average (EMA) θ_{EMA} in the final classification model $D(x, \theta_{EMA})$. We calculate the EMA parameters by $\theta_{EMA} = \alpha\theta_{EMA} + (1-\alpha)\theta_t$, where θ_t refers to the model parameters of the t^{th} iteration and α is set as 0.998 in our experiments. In general, EMA models provide more stable and slightly better results than non-EMA models.

4.3 Experimental Results and Discussion

Evaluation of loss functions. Table 1 shows the performance of using different combinations or variants of loss terms in the proposed model. In Table 1, DNN-SO is the model trained on supervised source data X^S only, DANN-O is our implementation of domain adversarial training, DANN-C is the model which includes the conditional entropy loss of target data X^T on DANN-O model, GAGL-FM is the model using the feature-matching GAN [25]

Source / Target	SVHN / MNIST	MNIST / SVHN	Syn-Num / SVHN	MNIST / MNIST-M	CIFAR / STL	STL / CIFAR	Syn-Signs / GTSRB
DNN-SO	85.7	41.0	90.1	64.4	75.3	56.7	95.5
DAN [18]	71.1	-	88.0	76.9	-	-	91.1
DANN [8]	71.1	35.7	90.3	81.5	-	-	88.7
ADDA [32]	76.0	-	-	-	-	-	-
SBADA-GAN [23]	76.1	61.1	-	**99.4**	-	-	96.7
DIFA [34]	89.7	-	93.0	-	-	-	-
ATT [22]	92.0	52.8	94.2	-	-	-	96.2
Assoc. [11]	97.6	-	91.9	89.5	-	-	97.7
VADA [25]	94.5	73.3	94.9	95.7	*78.3*	*71.4*	99.2
DIRT-T [25]	*99.4*	**76.5**	*96.2*	*98.7*	-	**73.3**	**99.6**
Self-ensemble [7]	**99.5**	37.5	**97.1**	-	**80.0**	69.9	*99.4*
GAGL	96.7	*74.6*	93.1	94.9	77.0	61.5	97.6

Table 2: Comparison with existing methods on UDA benchmarks.

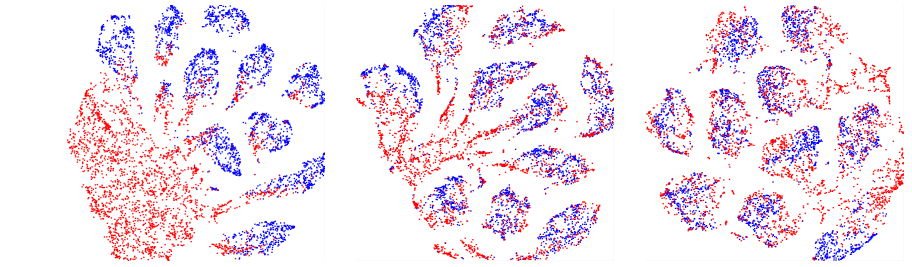
(i.e., matching the statistic mean values of two distributions) as the distribution criterion $Dist(\cdot,\cdot)$ in the generator loss L_G, and GAGL is the proposed model, which use CMD [35] as the criterion $Dist(\cdot,\cdot)$.

We first investigate the loss terms in domain adversarial training. In the first two rows of Table 1, although including the domain prediction loss L_a to label prediction loss L_y improves the classification performance in most cases, we observe that, in Syn-Num $\rightarrow$ SVHN and Syn-Signs $\rightarrow$ GTSRB tasks, including L_a even deteriorates the performance. Because the two tasks involve adaptation from synthetic data to real data, the domain adversarial learning alone is insufficient to generalize the learnt model to a much complex target domain. This also verifies our claim that a domain-invariant model does not necessarily lead to a target discriminative model.

Next, when including the generator model, we see that both GAGL-FM and GAGL improve the performance over DANN-O with a considerable margin. These results support our argument that both domain-invariant and target discriminative issues are essential to domain adaptation tasks. Moreover, we observe that GAGL generally performs better than GAGL-FM. One possible reason is that GAGL learns a better generator model through using a more effective distribution matching measurement CMD [35] in the generator loss L_G. The results show that a generative adversarial learning guided by an effective generator really affects the performance of GAGL model.

In Table 1, we also show the results of DANN-C, where we add the conditional entropy loss of the target data X^T to DANN-O model. The goal of conditional entropy loss is to learn discriminative model by enforcing confident prediction on X^T; however, learning directly from noisy or wrong predictions of X^T may lead to undesirable model, especially when there exists great diversity in X^T. Therefore, the proposed GAGL outperforms DANN-C model in most tasks, especially in MNIST $\rightarrow$ SVHN by more than 10%.

Comparisons. Table 2 shows the comparisons with existing methods on seven UDA benchmarks. The proposed GAGL achieves promising performance on most of the tasks. In comparison with ADDA [32], which did not explicitly address the target discriminative issue, the proposed GAGL not only learns a source discriminative model but also enforces the model to be target discriminative under the guidance of the generator. Thus, the learnt classification model can better generalize to the target domain. Note that, Self-ensemble [7] introduced additional data augmentation, including translation, flipping, and affine augmentation, to the task; whereas we only apply learnable instance normalization layer to our model. In DIRT-



(a) (b) (c)

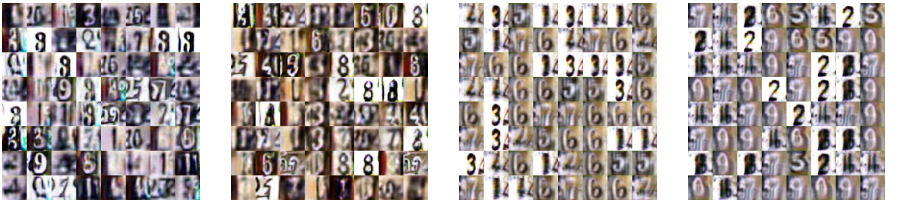
Figure 3: Visualization of feature representation from MNIST (blue) → MNIST-M (red) task via t-SNE. (a): DNN-SO; (b): DANN-O; (c): GAGL.

T [29], the authors assumed that a well-trained model on the source domain may degrade the performance on the target domain; thus, DIRT-T [29] is trained specifically for the target domain and thus usually achieve the best performance for each specific task. In the MNIST → SVHN task, our method especially achieves promising performance. In comparison with VADA [29], which includes conditional entropy loss to explicitly enforce the prediction confidence of target data X^T, our model is less prone to the noisy or wrong predictions of X^T. Moreover, the proposed GAGL directly learns a target discriminative model through generative adversarial guided learning and thus achieve comparable results on the UDA benchmarks.

Visualization. Figure 3 shows the t-SNE visualization [33] of feature representation in MNIST → MNIST-M task before the last dense layer of classification model D. We randomly select 3000 samples from the testing sets of both datasets, and show the visualization of 3 different models: DNN-SO, DANN-O, and GAGL. In the DNN-SO, the target samples spread farther away from source clusters and are widely scattered. The two models DANN-O and GAGL better preserve the clustering structures on the target samples. DANN-O produced several small clusters between large ones; these small clusters may contain ambiguous samples (e.g., domain-invariant but not discriminative) resulted from domain adversarial training. On the other hand, GAGL generally produced clusters with clear separability on the target samples. More t-SNE visualization data are provided in the supplementary file.

Figure 4 shows some random samples generated by the generative network. Although the images do not look visually plausible, these results meets the theoretical analysis in [5, 15] that the generated samples, which influence most to the classification model, are neither very realistic nor unreal ones. These moderate samples contribute the most to guide the decision boundary away from high density areas. Moreover, as the generative model in GAGL is meant to guide the classification model to generalize to the target domain, the generator trained by GAGL may not fit for synthesizing realistic images. On the other hand, we observe some slight mode collapsing (e.g., Figure 4 (c)&(d)). Nevertheless, because our goal is not to generate realistic images with high diversity, our experimental results verify that the guided learning from generated samples indeed improve the classification model in UDA tasks.

Discussion. Figure 5 shows a failure case in the MNIST → SVHN task. As shown in Figure 5 (a), the confusion matrix shows that all the other categories are classified with around 75% accuracy, except for the category of number 6. When examining this category of number 6, we found that many samples are misclassified as 0, 5, or 8 and thus have only



(a) (b) (c) (d)

Figure 4: Samples generated from MNIST ↔ SVHN tasks. (a)&(b): MNIST → SVHN; (c)&(d): SVHN → MNIST.

(a) (b) (c)

Figure 5: A failure case in the MNIST → SVHN task. (a): The confusion matrix; (b): the correctly classified samples; and (c): the wrongly classified samples.

37% classification accuracy. Nevertheless, as shown in Figure 5 (b) and (c), the correctly and wrongly classified samples actually look very similar. We suspect that, the poor performance is attributed to the noisy guided learning in the beginning of the training stage. That is, some target samples in this category are guided into wrong clusters. Although we did not observe similar situation in our other experiments, this problem seems more likely to occur when both the source-target domain shift and the target domain intra-class variance are comparatively large, such as in this MNIST → SVHN task.

5 Conclusion

This paper proposed a Generative Adversarial Guided Learning (GAGL) model to tackle the unsupervised domain adaptation problem, especially focus on the source-target domain shift problem and on learning a target discriminative model. We adopt the concept of domain adversarial training to build a domain-invariant classification model. By incorporating a generative network, we further push the decision boundary of the classification model away from high-density area of the target data. The proposed end-to-end GAGL model simultaneously learns the classification model and refines its decision boundary under the guidance of the generative network. Experimental results verify the effectiveness of the proposed GAGL model and show that GAGL achieves competitive results with state-of-the-art methods on standard UDA benchmarks. In the future, we will investigate the relation between mode collapsing and GAGL and will verify the effectiveness of GAGL on higher resolution images.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 33, pages 898–916, 2011.

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

[3] O. Chapelle and A. Zien. Semi-supervised classification by low density separation. In *Proc. International Workshop on Artificial Intelligence and Statistics*, pages 57–64, 2005.

[4] A. Coates, A. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proc. International Conference on Artificial Intelligence and Statistics*, volume 15, pages 215–226, 2011.

[5] Z. Dai, Z. Yang, F. Yang, W. W. Cohen, and R. Salakhutdinov. Good semi-supervised learning that requires a bad GAN. In *Advances in Neural Information Processing Systems*, 2017.

[6] G. French, M. Mackiewicz, and M. Fisher. Self-ensembling for visual domain adaptation. *arXiv:1706.05208v1*, 2017.

[7] G. French, M. Mackiewicz, and M. Fisher. Self-ensembling for visual domain adaptation. In *Proc. International Conference on Learning Representation*, 2018.

[8] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proc. International Conference on Machine Learning*, volume 37, pages 1180–1189, 2015.

[9] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. In *Advances in Neural Information Processing Systems*, 2014.

[10] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Scholkopf, and A. Smola. A kernel two-sample test. In *Proc. International Conference on Machine Learning*, volume 13, pages 723–773, 2012.

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

[12] J. Hoffman, E. Tzeng, T. Park, J. Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell. CyCADA: Cycle-consistent adversarial domain adaptation. *arXiv:1711.03213v3*, 2017.

[13] D. P. Kingma and J. L. Ba. Adam: a method for stochastic optimization. In *Proc. International Conference on Learning Representation*, 2015.

[14] A. Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[15] A. Kumar, P. Sattigeri, and P. T. Fletcher. Semi-supervised learning with GANs: Manifold invariance with improved inference. In *Advances in Neural Information Processing Systems*, 2017.

[16] S. Laine and T. Aila. Temporal ensembling for semi-supervised learning. In *Proc. International Conference on Learning Representation*, 2016.

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proc. of the IEEE*, volume 86, pages 2278–2324, 1998.

[18] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. In *Proc. International Conference on Machine Learning*, volume 37, pages 97–105, 2015.

[19] T. Miyato, S. Maeda, M. Koyama, and S. Ishii. Virtual adversarial training : a regularization method for supervised and semi-supervised learning. *arXiv:1704.03976*, 2017.

[20] B. Moiseyev, A. Konev, A. Chigorin, and A. Konushin. Evaluation of traffic sign recognition methods trained on synthetically generated data. *Advanced Concepts for Intelligent Vision Systems*, pages 576–583, 2013.

[21] 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 Fearture Learning*, 2011.

[22] B. T. Polyak and A. B. Juditsky. Acceleration of stochastic approximation by averaging. In *SIAM Journal on Control and Optimization*, volume 30, pages 838–855, 1992.

[23] P. Russo, F. M. Carlucci, T. Tommasi, and B. Caputo. From source to target and back: Symmetric bi-directional adaptive GAN. *arXiv:1705.08824v2*, 2017.

[24] K. Saito, Y. Ushiku, and T. Harada. Asymmetric tri-training for unsupervised domain adaptation. In *Proc. International Conference on Machine Learning*, volume 70, pages 2988–2997, 2017.

[25] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems*, 2016.

[26] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa. Generate to adapt: aligning domains using generative adversarial networks. *arXiv:1704.01705v3*, 2017.

[27] O. Sener, H. O. Song, A. Saxena, and S. Savarese. Learning transferrable representations for unsupervised domain adaptation. In *Advances in Neural Information Processing Systems*, 2016.

[28] J. Shen, Y. Qu, W. Zhang, and Y. Yu. Wasserstein distance guided representation learning for domain adaptation. In *AAAI Conference on Artificial Intelligence*, 2018.

[29] R. Shu, H. H. Bui, H. Narui, and S. Ermon. A DIRT-T approach to unsupervised domain adaptation. In *Proc. International Conference on Learning Representation*, 2018.

[30] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. The German traffic sign recognition benchmark: a multi-class classification competition. *IEEE International Joint Conference on Neural Networks*, pages 1453–1460, 2011.

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

[32] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2962–2971, 2017.

[33] L. v. d. Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[34] R. Volpi, P. Morerio, S. Savarese, and V. Murino. Adversarial feature augmentation for unsupervised domain adaptation. *arXiv:1711.08561v1*, 2017.

[35] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschlager, and S. Saminger-Platz. Central moment discrepancy (CMD) for domain-invariant representation learning. In *Proc. International Conference on Learning Representation*, 2017.