Bidirectional Adversarial Training for Semi-Supervised Domain Adaptation

Pin Jiang[1,2*], **Aming Wu**[1,2*], **Yahong Han**[1,2†], **Yunfeng Shao**[3], **Meiyu Qi**[3] and **Bingshuai Li**[3]
[1]College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]Tianjin Key Lab of Machine Learning, Tianjin University, Tianjin, China
[3]Huawei Noah's Ark Lab
{jpin, tjwam, yahong}@tju.edu.cn, {shaoyunfeng, qimeiyu, libingshuai}@huawei.com

Abstract

Semi-supervised domain adaptation (SSDA) is a novel branch of machine learning that scarce labeled target examples are available, compared with unsupervised domain adaptation. To make effective use of these additional data so as to bridge the domain gap, one possible way is to generate adversarial examples, which are images with additional perturbations, between the two domains and fill the domain gap. Adversarial training has been proven to be a powerful method for this purpose. However, the traditional adversarial training adds noises in arbitrary directions, which is inefficient to migrate between domains, or generate directional noises from the source to target domain and reverse. In this work, we devise a general bidirectional adversarial training method and employ gradient to guide adversarial examples across the domain gap, i.e., the Adaptive Adversarial Training (AAT) for source to target domain and Entropy-penalized Virtual Adversarial Training (E-VAT) for target to source domain. Particularly, we devise a Bidirectional Adversarial Training (BiAT) network to perform diverse adversarial trainings jointly. We evaluate the effectiveness of BiAT on three benchmark datasets and experimental results demonstrate the proposed method achieves the state-of-the-art.

1 Introduction

When Semi-Supervised Learning (SSL) [Rasmus *et al.*, 2015; Laine and Aila, 2016; Tarvainen and Valpola, 2017] meets Domain Adaptation (DA) [Ganin and Lempitsky, 2014; Long *et al.*, 2015; Long *et al.*, 2017], there recently highlights an appealing task of Semi-Supervised Domain Adaptation (SSDA) [Saito *et al.*, 2019]. SSDA refers to the problem that we not only have plentiful labeled source domain data and unlabeled target domain data, but a small amount of labeled target domain data, typically one or three per class. SSDA methods usually use labeled examples (from both source and target domain) to jointly train a network and use unlabeled examples to regularize it. However, the importance of the labeled target data is downplayed. Although the labeled target data are still scarce, they may provide effective clues to explore the connection between the source and target domain and bridge the gap between them, such as generating perturbed auxiliary examples between two domains. As the triangle class in Fig. 1, the BiAT examples ▲ are generated from both the source domain ▲→▲ and the target domain ▲, △→▲, and fill the domain gap.

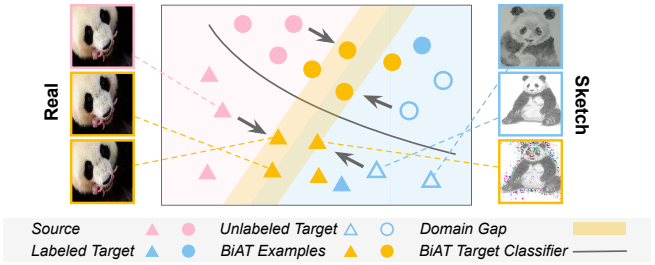

Figure 1: Our BiAT method generates bidirectional adversarial examples between two domains and fill the domain gap, i.e., source to target and target to source. The arrows indicate the directions in which adversarial examples are generated.

Corrupting training data with noises have been well-known to be helpful in stabilizing prediction [Park *et al.*, 2018]. Recently, gradient-based perturbation generation methods, i.e., Adversarial Training (AT) [Goodfellow *et al.*, 2014] as well as Virtual Adversarial Training (VAT) [Miyato *et al.*, 2018], are proposed to generate adversarial examples, which are images with additional perturbations. Adversarial examples have been proven to be effective on supervised and semi-supervised tasks [Miyato *et al.*, 2016; Miyato *et al.*, 2018]. The effectiveness owns to their ability of local smoothness around data points, and pushing decision boundary away from data points, which leads to an improvement of the model robustness for raw examples. However, it is not straightforward to use AT or VAT in SSDA. The noise direction of AT and VAT is arbitrary, which cannot make the adversarial examples cross the domain gap.

To solve the SSDA problem, an intuitive idea is to force AT and VAT to generate adversarial examples between the two domains and fill the domain gap. [Liu *et al.*, 2019] introduced

*Equal contributions
†Corresponding author

adversarial training at the feature layer to enhance the unsupervised domain adaptation. However, they use traditional AT on the source and target examples. The generated examples surround the original examples, which has only limited ability to fill the domain gap.

In order to guide the direction of generating adversarial examples from one domain towards another, we devise two opposing adversarial training methods which form a bidirectional strategy. We firstly propose Adaptive Adversarial Training (AAT), a novel adversarial training notion for specific SSDA scenarios that generates adaptive adversarial examples from the source to target domain. AAT treats a source example x_s as a optimizable object, which is fed into a well-trained target domain classifier, then the target domain classifier will optimize x_s along the direction of gradient descent and force x_s to cross into the target domain. Moreover, we introduce Entropy-penalized Virtual Adversarial Training (E-VAT) which improves VAT and generates adversarial examples from the target to source domain. Finally, we propose a uniform Bidirectional Adversarial Training (BiAT) network to perform AT, AAT, and E-VAT jointly. We evaluate BiAT on three benchmark datasets and conduct extensive ablation study. Experiments show the BiAT network advances all the state-of-the-arts.

Our contributions can be summarized into threefold. (1) We devise a novel Adaptive Adversarial Training for specific SSDA setting that generates adversarial examples from the source to target domain. (2) We introduce the Entropy-penalized Virtual Adversarial Training to generate more accurate adversarial examples from the target to source domain. (3) We propose a uniform Bidirectional Adversarial Training network to integrate three adversarial training methods and demonstrate the effectiveness of them in SSDA scenarios.

2 Related Work

Semi-Supervised Domain Adaptation. SSDA was highlighted by [Saito *et al.*, 2019] recently, which is a combination of semi-supervised learning (SSL) and domain adaptation (DA). The MME method they proposed maximize the entropy of unlabeled target data to optimize classifier, and minimize the entropy with respect to the feature extractor to cluster features. Recently, some effective arts of SSL have emerged. [Grandvalet and Bengio, 2005] considered entropy minimization as a regularizer to incorporate unlabeled data. [Laine and Aila, 2016] introduced $\prod$-model and temporal-ensembling, a consensus prediction of the unknown labels using an exponential moving average of outputs. [Berthelot *et al.*, 2019] proposed MixMatch data-augment to generate low-entropy pseudo-labels. On the other hand, previous works have addressed the unsupervised DA problem which generalizes a learner across different domains, by either matching the marginal distributions or the conditional distributions [Long *et al.*, 2018].

Adversarial Training. AT was proposed by [Szegedy *et al.*, 2013] originally. They discovered that deep networks are particularly vulnerable to minor adversarial perturbations applied to the input. [Goodfellow *et al.*, 2014] proposed the definition of adversarial examples and pointed out that adversarial training can restraint the impact of adversarial perturbations to some extent. Not just defense against adversarial attacks, adversarial training can also promise a variety of machine learning tasks. [Miyato *et al.*, 2018] introduced virtual adversarial training in supervised and semi-supervised scenarios to promote local smoothness around data points.

3 Preliminaries

3.1 Adversarial Training

Adversarial Training is recognized as an effective method to improve the robustness of models [Szegedy *et al.*, 2013; Goodfellow *et al.*, 2014], which promotes the local smoothness and mitigates over-fitting by regularizing the network.

For labeled input pair (x_l, y_l) and unlabeled input x_u, we denote the output distribution parameterized by Θ as $p(y|x, \Theta)$. The loss function of adversarial training in [Goodfellow *et al.*, 2014] can be written as

$$L_{\text{adv}}(x_l, \Theta) = D\left[q(y_l), p(y|x_l + r_{\text{adv}}, \Theta)\right], \quad (1)$$

$$\text{where } r_{\text{adv}} := \underset{r;\|r\|\leq\epsilon}{\arg\max} D\left[q(y_l), p(y|x_l + r, \Theta)\right]. \quad (2)$$

$D[\cdot, \cdot]$ is a function that measures the divergence between two distributions. The $q(y_l)$ is the true distribution of label which is generally approximated by one-hot vector of y_l.

When the norm is ℓ_2, [Miyato *et al.*, 2016] proposed to approximate the adversarial perturbation r_{adv} by

$$r_{\text{adv}} \approx \epsilon \frac{g}{\|g\|_2}, \text{ where } g = \nabla_{x_l} D\left[q(y_l), p(y|x_l, \Theta)\right]. \quad (3)$$

g is the gradient that can be efficiently computed by back-propagation, ϵ is the perturbation amplitude.

3.2 Virtual Adversarial Training

When facing unsupervised or semi-supervised learning tasks, [Miyato *et al.*, 2018] proposed Virtual Adversarial Training which replaces the true label $q(y_l)$ in Eq.(1) with its current approximation $p(y|x_l, \hat{\Theta})$. Let $x_\star$ represents either x_l or x_u. The objective function is given by

$$L_{\text{vadv}}(x_*, \Theta) = D\left[p(y|x_*, \hat{\Theta}), p(y|x_* + r_{\text{vadv}}, \Theta)\right], \quad (4)$$

$$\text{where } r_{\text{vadv}} := \underset{r;\|r\|\leq\epsilon}{\arg\max} D\left[p(y|x_*, \hat{\Theta}), p(y|x_* + r)\right], \quad (5)$$

Then $L_{\text{vadv}}(x_*, \Theta)$ can be considered as a regularization term over all input examples.

4 Bidirectional Adversarial Training

In this section, we propose a novel Bidirectional Adversarial Training (BiAT) method for SSDA task, illustrated in Fig. 2. We devise two opposing adversarial training which form a bidirectional strategy of adversarial example generation. AAT produces adversarial examples from the source to target domain in Section 4.2. E-VAT generates examples from the target to source domain, elaborated in Section 4.3. In addition, we combine the AT, AAT, E-VAT, and the minimax entropy method [Saito *et al.*, 2019] to jointly construct an end-to-end network in Section 4.4.

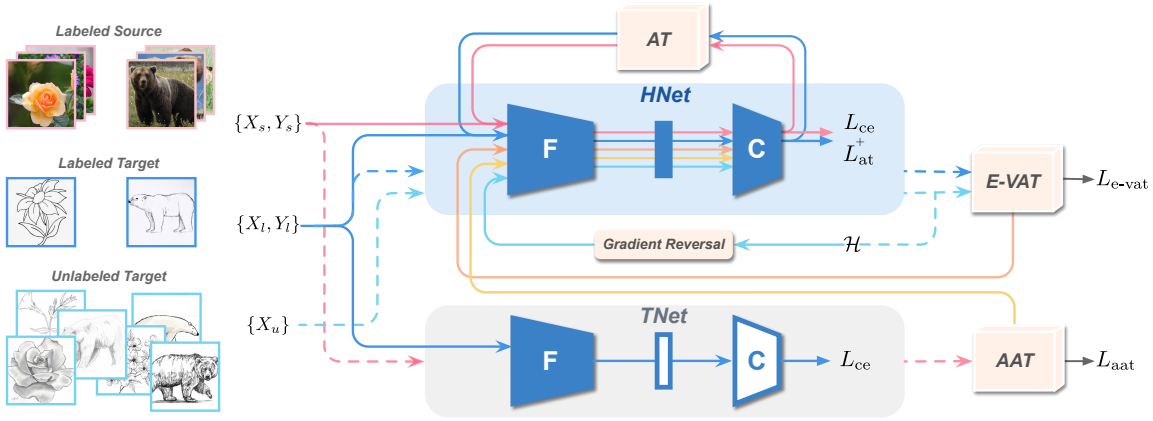

Figure 2: The architecture of Bidirectional Adversarial Network (BiAT). BiAT consists of two parallel networks *HNet* and *TNet* which share the feature extractor F. The solid lines indicate that data forward go through and backward optimize the net, the dotted lines indicate that only forward propagation is used to obtain the gradients for generating adversarial perturbations. *HNet* takes both source and target data as input and is updated by AT, AAT, and E-VAT. *TNet* is employed as the auxiliary net for AAT and is only trained with labeled target data $\{X_l, Y_l\}$. Moreover, there is a gradient reversal layer between F and C to conduct the minimax entropy penalty.

4.1 Semi-Supervised Domain Adaptation

Under the SSDA setting, the data comes from two different domains, the source domain $\mathcal{S}$ and the target domain $\mathcal{T}$. Data from the source domain $\mathcal{S} = \{X_s, Y_s\}$ are labeled, and each data point $x_s \in X_s$ has an associated label $y_s \in Y_s$. A small number of target domain data are labeled (one or three per class), formalized as $\mathcal{T}_l = \{X_l, Y_l\}$ that data point $x_l \in X_l$ has an associated label $y_l \in Y_l$. The rest is unlabeled target domain data $\mathcal{T}_u = \{X_u\}$. Our goal is to find a model that performs well on unlabeled target domain test data.

Similar to the previous domain adaptation models [Ganin and Lempitsky, 2014; Saito *et al.*, 2018], our backbone network consists of two modules, the feature extractor F and the classifier C. We can employ commonly used deep convolutional neural networks as feature extractors, such as AlexNet [Krizhevsky *et al.*, 2012] and ResNet [He *et al.*, 2016]. We denote the network outputs as $\mathbf{h} = C(F(x)), \in \mathbb{R}^c$.

Previously, [Saito *et al.*, 2019] has shown that performing ℓ_2 normalization on the outputs of F achieves good performance. We further conduct weight normalization on the classifier. We write the weight matrix $\mathbf{W}_C \in \mathbb{R}^{d\times c}$ as $[\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_c]$, where each class corresponds to a d-dimensional weight vector. Then C takes normalized features as input and calculate the cosine similarity scores to weight vectors as $\mathbf{h} = \frac{1}{T}\frac{F(x)^{\mathrm{T}}\mathbf{W}_C}{\|F(x)\|\|\mathbf{W}_C\|}$, where T is the temperature parameter. Consequently, the outputs is the prediction scores of corresponding classes. Then the weight vectors of $\mathbf{W}_C$ can be interpreted as prototypes of classes, which may be effective due to reducing intra-class variations.

4.2 Adaptive Adversarial Training

Here, we propose Adaptive Adversarial Training (AAT), a novel adversarial training notion for SSDA task that can generate adaptive adversarial examples to fill the domain gap from the source to target domain ($\mathcal{S} \to \mathcal{T}$). Unlike traditional adversarial training, which adds arbitrary perturbation to examples in a single domain, our AAT adds perturbation in a direction and can indicate the direction of domain adaptation.

In AAT, we can access to plentiful labeled source data, and other small amount of labeled target data. Suppose we can find a clear direction to guide the source examples towards the target domain, then we use these adversarial examples to train the network may effectively improve the performance.

Main Idea. The goal of AAT is to generate directional perturbation on a source example, and more importantly, the direction should point to the target examples of the same class. Following the idea of adversarial training, we employ the gradients to do this work. The novel idea is to treat a source example x_s as an optimizable object, which is fed into a frozen well-trained target domain network *TNet*. Since the *TNet* can only recognize target data, then *TNet* will optimize x_s along **the direction of gradient descent** and force x_s to cross into the target domain. Note that, this optimization process only occurs in the example space and the optimized examples are regarded as adaptive adversarial examples. The AAT is inspired by targeted adversarial attack which aims to generate adversarial examples targeting a specific class from a given image and regards the direction of gradient descent as the direction of noises [Carlini and Wagner, 2017].

We first train a capable *TNet* as Θ_T using all the labeled target data $\mathcal{T}_l$. Then we froze Θ_T and feed the source data $\mathcal{S} = \{X_s, Y_s\}$ into the network and generate adaptive adversarial examples by the direction of gradient descent. With the ℓ_2 norm, the adversarial perturbation can be approximated by

$$r_{\text{aat}} \approx \epsilon \frac{g}{\|g\|_2}, \text{ where } g = -\nabla_{x_s} D\left[q(y_s), p(y|x_s, \Theta_T)\right]. \tag{6}$$

The only difference from traditional AT is that the network for gradients becomes Θ_T. The loss function of AAT is

$$L_{\text{aat}}(x_s, \Theta) = D\left[q(y_s), p(y_s|x_s + r_{\text{aat}}, \Theta)\right], \tag{7}$$

where Θ denotes the task-specific network.

4.3 Entropy-penalized Virtual Adversarial Training

In contrast to AAT, we introduce Entropy-penalized Virtual Adversarial Training (E-VAT) to generate adversarial examples from the target to source domain ($\mathcal{T} \rightarrow \mathcal{S}$), following the philosophy of VAT. As mentioned in Section 3.2, VAT uses approximation method to generate perturbations on unlabeled examples and ensures consistency of network predictions. This makes the adversarial examples towards the source domain to some extent. However, limited by the lack of clear labels, traditional VAT is insufficient in SSDA task.

We make three improvements to the traditional VAT: (1) We use multi-step iteration to obtain more accurate adversarial perturbations; (2) In order to obtain clear supervision, we use hard pseudo-labels, i.e. one-hot labels, instead of ambiguous approximate predictions $p(y|x^*, \hat{\Theta})$ in VAT; (3) We introduce two entropy-based consistency penalization to obtain more consistent pseudo-labels and prioritize the high-confidence examples, respectively.

We first start E-VAT from a small random noise, and then follow **the direction of gradient ascent** to iteratively obtain multiple small perturbations. Although the noise direction of E-VAT is arbitrary, it is responsible for smoothing the local space and filling the domain gap to some extent. Each iteration of E-VAT corresponds to a classifier prediction. These outputs form a prediction matrix $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_N] \in \mathbb{R}^{N \times d}$, where $\mathbf{h}_n$ denotes the prediction of n-th iteration and N is the number of iterations.

Row-wise and column-wise entropy penalty. For unlabeled examples, the pseudo-labels of N iterations may be inconsistent, especially in the early stage of training, which will confuse the pseudo-labels, e.g. the one-hot labels of $\mathbf{h}_1$ and $\mathbf{h}_N$ may be different. For a vector, entropy is maximized if the distribution is uniform. We propose to penalize $\mathbf{H}$ that we want low row-wise entropy for clear predictions with high confidence, and high column-wise entropy to encourage consistent predictions in N iterations, so as to obtain an explicit pseudo-label. We quantify the row-wise and column-wise entropy penalty as weight scores.

Example-specific entropy penalty. Owing to the VAT sometimes gives a multi-peak prediction. This phenomenon suggests that such examples near the prototypes of two or more classes, and the wrong pseudo-label will mislead the model. Therefore, we introduce the example-specific entropy penalty to suppress these examples corresponding to small entropy $\mathcal{H}^e = -\sum^d p(\mathbf{h}') \log p(\mathbf{h}')$ and the example-specific weight scores denote as $\mathbf{W}^e = e^{-\mathcal{H}^e}$.

4.4 Bidirectional Adversarial Training Network

As shown in Fig. 2, we design two parallel networks, hybrid domain network *HNet* and target domain network *TNet*. We first use the labeled data $\mathcal{S} \cup \mathcal{T}_l$ to jointly train the *HNet* and use only the labeled target data $\mathcal{T}_l$ to train the *TNet*, used to assist AAT according to Section 4.2. We denote the parameters of feature extractor F as Θ_F, classifier C as Θ_C. To avoid overfitting, we freeze the F of *TNet* and share the parameters with *HNet*. Then we have $\Theta_T = \{\Theta_F, \Theta_C^T\}$ and $\Theta_H = \{\Theta_F, \Theta_C^H\}$.

Algorithm 1 Bidirectional Adversarial Training (BiAT)

Input: Source data $\mathcal{S}$; Target labeled data $\mathcal{T}_l$; Target unlabeled data $\mathcal{T}_u$; Parameters of *HNet* Θ_H; Parameters of *TNet* Θ_T; Batch size N.

while not converged **do**
  Sample a mini-batch of N examples from $\mathcal{S}$ and $\mathcal{T}_l$ separately, $2N$ examples from $\mathcal{T}_u$
  // optimize *HNet* with cross entropy, AT, AAT, and E-VAT
  $\Theta_H \leftarrow L_{\text{ce}}(X_s, X_l, Y_s, Y_l, \Theta_H)$
  $\Theta_H \leftarrow L_{\text{at}}(X_s, X_l, Y_s, Y_l, \Theta_H)$
  $\Theta_H \leftarrow L_{\text{aat}}(X_s, Y_s, \Theta_T, \Theta_H)$
  $\Theta_H \leftarrow L_{\text{e-vat}}(X_l, X_u, \Theta_H)$
  // optimize *TNet* with cross entropy
  $\Theta_T \leftarrow L_{\text{ce}}(X_l, Y_l, \Theta_T)$
  // optimize *HNet* with minimax entropy
  $\Theta_H \leftarrow L_{\text{mme}}(X_u, \Theta_H)$
end while
return solution

In order to generate adversarial examples between the source and target domain, we propose the Bidirectional Adversarial Training mechanism to perform AT, AAT, and E-VAT on the *HNet* jointly. In addition, we use the minimax entropy as an regular term because of the good performance [Saito *et al.*, 2019]. We devise a three-stage training strategy in a mini-batch, shown in Algorithm 1. At first, we optimize the *HNet* with the cross entropy, AT, AAT, and E-VAT. Secondly, we optimize the classifier of *TNet*. Thirdly, we apply the minimax entropy regularization to *HNet*.

Task-specific objective. We define the task-specific objective function regarding the labeled data $\{X, Y\} = \{X_s \cup X_l, Y_s \cup Y_l\}$. As for classification task, the training objective of *HNet* is typical defined as cross entropy loss

$$L_{\text{ce}}(X, Y, \Theta_H) = -\mathbb{E}_{x,y \sim X,Y} \log p(y|x, \Theta_H). \tag{8}$$

AT objective. According to the Section 3.1, we use the labeled data $\mathcal{S} \cup \mathcal{T}_l$ to conduct traditional AT, the training objective can be written as $L_{\text{at}}(X, Y, \Theta_H)$.

AAT objective. We use the labeled source data for AAT to generate adaptive adversarial examples from the source to target domain. The training objective is $L_{\text{aat}}(X_s, Y_s, \Theta_T, \Theta_H)$.

E-VAT objective. We employ the E-VAT with unlabeled data on *HNet* to generate adversarial examples from the target to source domain. The VAT objective can be written as $L_{\text{e-vat}}(X_l, X_u, \Theta_H)$.

Minimax Entropy objective. We apply the previous minimax entropy objective on classifier to maximize the entropy $\mathcal{H}$ of unlabeled target data, and minimize the entropy with respect to the feature extractor to cluster features [Saito *et al.*, 2019], which form an adversarial minimax process. We use the gradient reversal layer to simplify the training process.

$$L_{\text{mme}}^F = \underset{\Theta_F}{\text{argmin}} \mathcal{H}, \text{ and } L_{\text{mme}}^C = \underset{\Theta_C}{\text{argmin}} - \mathcal{H}. \tag{9}$$

Net	Methods	R → C		R → P		P → C		C → S		S → P		R → S		P → R		Mean	
		1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot
AlexNet	S+T	43.3	47.1	42.4	45.0	40.1	44.9	33.6	36.4	35.7	38.4	29.1	33.3	55.8	58.7	40.0	43.4
	DANN	43.3	46.1	41.6	43.8	39.1	41.0	35.9	36.5	36.9	38.9	32.5	33.4	53.6	57.3	40.4	42.4
	ADR	43.1	46.2	41.4	44.4	39.3	43.6	32.8	36.4	33.1	38.9	29.1	32.4	55.9	57.3	39.2	42.7
	CDAN	46.3	46.8	45.7	45.0	38.3	42.3	27.5	29.5	30.2	33.7	28.8	31.3	56.7	58.7	39.1	41.0
	ENT	37.0	45.5	35.6	42.6	26.8	40.4	18.9	31.1	15.1	29.6	18.0	29.6	52.2	60.0	29.1	39.8
	MME	48.9	55.6	48.0	49.0	**46.7**	51.7	36.3	39.4	39.4	**43.0**	33.3	37.9	56.8	**60.7**	44.2	48.2
	BiAT	**54.2**	**58.6**	**49.2**	**50.6**	44.0	**52.0**	**37.7**	**41.9**	**39.6**	42.1	**37.2**	**42.0**	**56.9**	58.8	**45.5**	**49.4**
ResNet	S+T	55.6	60.0	60.6	62.2	56.8	59.4	50.8	55.0	56.0	59.5	46.3	50.1	71.8	73.9	56.9	60.0
	DANN	58.2	59.8	61.4	62.8	56.3	59.6	52.8	55.4	57.4	59.9	52.2	54.9	70.3	72.2	58.4	60.7
	ADR	57.1	60.7	61.3	61.9	57.0	60.7	51.0	54.4	56.0	59.9	49.0	51.1	72.0	74.2	57.6	60.4
	CDAN	65.0	69.0	64.9	67.3	63.7	68.4	53.1	57.8	63.4	65.3	54.5	59.0	73.2	78.5	62.5	66.5
	ENT	65.2	71.0	65.9	69.2	65.4	71.1	54.6	60.0	59.7	62.1	52.1	61.1	75.0	78.6	62.6	67.6
	MME	70.0	72.2	67.7	**69.7**	69.0	71.7	56.3	**61.8**	**64.8**	66.8	**61.0**	61.9	76.1	78.5	66.4	68.9
	BiAT	**73.0**	**74.9**	**68.0**	68.8	**71.6**	**74.6**	**57.9**	61.5	63.9	**67.5**	58.5	**62.1**	**77.0**	**78.6**	**67.1**	**69.7**

Table 1: Accuracy on the DomainNet dataset (%).

Methods	Office (W→A)		Office (D→A)		Office-Home	
	1-shot	3-shot	1-shot	3-shot	1-shot	3-shot
S+T	50.4	61.2	50.0	62.4	44.1	50.0
DANN	57.0	64.4	54.5	65.2	45.1	50.3
ADR	50.2	61.2	50.9	61.4	44.5	49.5
CDAN	50.4	60.3	48.5	61.4	41.2	46.2
ENT	50.7	64.0	50.0	66.2	38.8	50.9
MME	57.2	67.3	**55.8**	67.8	49.2	55.2
BiAT	**57.9**	**68.2**	54.6	**68.5**	**49.6**	**56.4**

Table 2: Accuracy on Office and Office-Home (%) with AlexNet.

Methods	R→C		R→S	
	1-shot	3-shot	1-shot	3-shot
Baseline (S+T)	43.3	47.1	29.1	33.3
Baseline w/ AT	48.7	52.6	32.5	39.0
Baseline w/ AAT	46.6	49.6	31.6	36.6
Baseline w/ E-VAT	46.5	49.2	30.9	35.7
BiAT w/o AT	50.8	55.4	33.0	39.6
BiAT w/o AAT	49.8	54.7	31.3	37.7
BiAT w/o E-VAT	50.6	53.4	34.6	40.5
BiAT w/o MME	46.0	53.5	29.0	35.1
BiAT	54.2	58.6	37.2	42.0

Table 3: Ablation study on DomainNet (%) with AlexNet.

Overall objectives. The overall training objective for training *HNet* can be written as

$$L_{HNet} = L_{\text{ce}}(X_s, X_l, Y_s, Y_l, \Theta_H) \tag{10}$$
$$+ \lambda_1 L_{\text{at}} + \lambda_2 L_{\text{aat}} + \lambda_3 L_{\text{e-vat}} + \lambda_4 L_{\text{mme}}, \tag{11}$$

where $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are hyper-parameters. L_{mme} is a simplified form of L^F_{mme} and L^C_{mme}.

In addition, the training objective of *TNet* is

$$L_{TNet} = L_{\text{ce}}(X_l, Y_l, \Theta_T). \tag{12}$$

The whole training process is shown in Algorithm 1.

5 Experiments

5.1 Experimental Setup

Datasets. We address the SSDA problem and validate the benefits of BiAT on three datasets. **DomainNet** is a recent benchmark dataset for large-scale domain adaptation that has 6 domains and 345 classes [Peng *et al.*, 2019]. Following the setup of [Saito *et al.*, 2019], we pick 4 domains (Real, Clipart, Painting, Sketch) and 126 classes to construct 7 scenarios, shown in the first row of Table 1. **Office** contains 3 domains with 31 classes and we eliminate the domains with less examples and construct 2 scenarios, Webcam to Amazon (W→A) and DSLR to Amazon (D→A). **Office-Home** contains 4 domains (Real, Clipart, Art, Product) with 65 classes. For each dataset, we use the randomly selected one or three labeled examples per class as the labeled target examples (1-shot and 3-shot) by [Saito *et al.*, 2019]. Other three labeled target examples are used as validation set and the remaining are used as unlabeled target data.

Implementation details. For fair comparisons with other methods, we follow previous work and take the shallow network AlexNet [Krizhevsky *et al.*, 2012] and deep network ResNet34 [He *et al.*, 2016] as the backbones. We employ a shared pre-trained network as feature extractor and randomly initialized classifiers for both *HNet* and *TNet*. We set λ_1 in Eq. (11) as 1, λ_2 as 2. To avoid misjudgments of $L_{\text{e-vat}}$ in early training, we use cosine warm-up on λ_3. L_{mme} is sensitive to datasets, even sub-domains of a same dataset, and we choose $\lambda_4 \in [0.1, 2.5]$ for different scenarios. We adopt SGD as an optimizer with initial learning rate 0.01. In order to make a fair comparison, other setup we choose the same as [Saito *et al.*, 2019].

5.2 The Analysis of Experimental Results

We compare our BiAT network against several baselines. "S+T" is the model trained with labeled data (both source and target) and without unlabeled data. DANN [Ganin and Lempitsky, 2014], ADR [Saito *et al.*, 2017], and CDAN [Long *et al.*, 2018] are advanced unsupervised domain adaptation methods. ENT is the model jointly trained by source and target labeled data, and unlabeled target data are employed for standard entropy minimization. MME [Saito *et al.*, 2019] is the SSDA method with only minimax entropy loss.

DomainNet. In Table 1, we show the detailed comparison on DomainNet dataset. Our BiAT method outperforms the most recent MME method on average by 1.3% (1-shot) and 1.2% (3-shot) with AlexNet, 0.7% (1-shot) and 0.8% (3-shot)

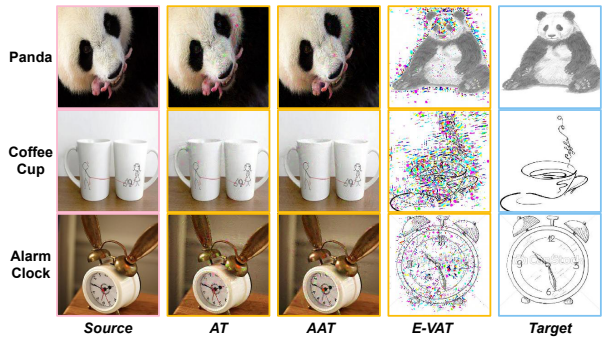

Figure 3: Examples of BiAT. The columns represent the source examples, the examples generated by three adversarial training methods, and the target examples.

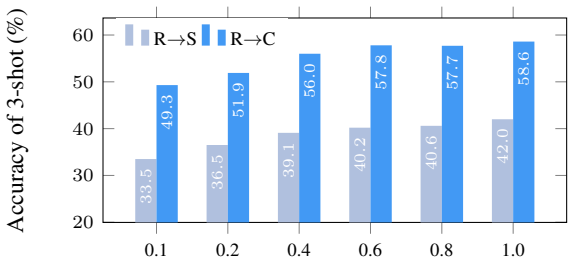

Figure 4: Accuracy vs the number of BiAT examples. The horizontal axis represents the ratio of used adversarial examples.

with ResNet. The result demonstrates that our BiAT method is effective in both shallow and deep networks. In some high-quality scenarios with Real images as source, taking AlexNet as an example, we achieve outstanding performance boost at 5.3% and 3.0% on R→C, 3.9% and 4.1% on R→S with 1-shot and 3-shot, separately.

Office and Office-Home. In Table 2, we show the experimental results on Office and Office-Home dataset. In the two scenarios of Office, BiAT demonstrates comparable performance, which is only behind the MME on D→A (1-shot). That may be due to the DSLR set is small with only hundreds of images and less data are inefficient to fill the domain gap. For the Office-Home dataset, BiAT achieves a best overall accuracy at 49.6% (1-shot) and 56.4% (3-shot), outperforming 0.4% and 1.2% than the prior state-of-the-art.

In Fig. 3, we visualize the adversarial examples from BiAT on the DomainNet dataset. The middle three columns show the generated examples with additional perturbations from AT, AAT, and E-VAT, respectively.

5.3 Ablation Study

In order to exploit different model variants and analyze the effectiveness, we conduct ablation study in two scenarios on DomainNet, R→C and R→S, shown in Table 3.

We regard the "S+T" model as the baseline of BiAT without adversarial training methods. As can be seen in the first part of Table 3, AT apparently boosts the accuracy by around 4% from baseline on all the scenarios. AAT can promote the baseline by around 3% accuracy. Similarly, E-VAT achieves around 3% accuracy improvements. These two results not only show the validity of the BiAT methods, but also indicate the two generation directions are almost equally important.

As shown in the second part of Table 3, it causes a decline in performance without any adversarial training methods. This proves that all the adversarial training components are necessary. Note that the MME is also used as a term for stable training, and it is sensitive to hyper-parameters. Simply removing the MME may lead to poor performance, such as the R→S (1-shot) case. When all methods are combined, the scores of the BiAT demonstrate the effectiveness of BiAT.

Varying number of bidirectional adversarial examples. We take the number of bidirectional adversarial examples as a variable and analyse the effort of example size. As shown in Fig. 4, the horizontal axis represents the ratio of adversarial examples. The ratio increases from left to right, and "1.0" indicates all the adversarial examples are used, i.e. $2N$ AT examples, N AAT examples, and $2N$ E-VAT examples in a mini-batch, where N is the batch size. Obviously, the increase of example size significantly improves the accuracy of the model. This result is quite reasonable, because more adversarial examples can better smooth the data points, and fill the gaps between domains, thus learning a capable classifier.

5.4 Discussion of BiAT

In general, our BiAT method is validated to be valid on all the three datasets and has around 1% accuracy improvement on average. Because BiAT is essentially a data enhancement method and does not innovate the image recognition backbone, it is reasonable that BiAT cannot greatly improve the classification performance. Even so, BiAT still performs well in many cases (around 5% accuracy improvements). In addition, as a general adversarial training paradigm, BiAT can universally boost models, which can be regarded as a plug-in method for SSDA or other domain adaptation tasks. As long as there is an input example and a network that can give a clear gradient direction, AAT can guide the perturbation along the desired direction and obtain the corresponding adversarial examples. For unsupervised data, E-VAT is also universal. We argue that the BiAT method can act as a general technique for SSDA or other domain adaptation problem.

6 Conclusion

In this paper, we devise a general bidirectional adversarial training method and employ gradient to guide adversarial examples across domain gap and fill the gap. We propose two effective components, AAT and E-VAT. AAT generates adaptive adversarial examples from the source to target domain, while E-VAT does the opposite. We evaluate our method on three benchmark datasets and conduct extensive ablation study to the effectiveness of the BiAT. Experiments show the network achieves the state-of-the-art.

Acknowledgments

This work is supported by the NSFC (under Grant 61876130, 61932009). Thanks to Huawei Noah's Ark Lab NetMIND Research Team for funding this research.

References

[Berthelot *et al.*, 2019] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 5050–5060, 2019.

[Carlini and Wagner, 2017] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy*, pages 39–57. IEEE, 2017.

[Ganin and Lempitsky, 2014] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495*, 2014.

[Goodfellow *et al.*, 2014] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[Grandvalet and Bengio, 2005] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *Advances in Neural Information Processing Systems*, pages 529–536, 2005.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[Krizhevsky *et al.*, 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[Laine and Aila, 2016] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016.

[Liu *et al.*, 2019] Hong Liu, Mingsheng Long, Jianmin Wang, and Michael Jordan. Transferable adversarial training: A general approach to adapting deep classifiers. In *International Conference on Machine Learning*, pages 4013–4022, 2019.

[Long *et al.*, 2015] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I Jordan. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791*, 2015.

[Long *et al.*, 2017] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *International Conference on Machine Learning*, pages 2208–2217, 2017.

[Long *et al.*, 2018] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pages 1640–1650, 2018.

[Miyato *et al.*, 2016] Takeru Miyato, Andrew M Dai, and Ian Goodfellow. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*, 2016.

[Miyato *et al.*, 2018] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):1979–1993, 2018.

[Park *et al.*, 2018] Sungrae Park, JunKeon Park, Su-Jin Shin, and Il-Chul Moon. Adversarial dropout for supervised and semi-supervised learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[Peng *et al.*, 2019] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1406–1415, 2019.

[Rasmus *et al.*, 2015] Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *Advances in Neural Information Processing Systems*, pages 3546–3554, 2015.

[Saito *et al.*, 2017] Kuniaki Saito, Yoshitaka Ushiku, Tatsuya Harada, and Kate Saenko. Adversarial dropout regularization. *arXiv preprint arXiv:1711.01575*, 2017.

[Saito *et al.*, 2018] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3723–3732, 2018.

[Saito *et al.*, 2019] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8050–8058, 2019.

[Szegedy *et al.*, 2013] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[Tarvainen and Valpola, 2017] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*, pages 1195–1204, 2017.