



# Self-Training with Label-Feature-Consistency for Domain Adaptation

Yi Xin, Siqu Luo, Pengsheng Jin, Yuntao Du, and Chongjun Wang<sup>(✉)</sup>

National Key Laboratory for Novel Software Technology, Department of Computer  
Science and Technology, Nanjing University, Nanjing, China  
{mf21330098,mf21330060,mf21330042,dz1833005}@smail.nju.edu.cn,  
chjwang@nju.edu.cn

**Abstract.** Mainstream approaches for unsupervised domain adaptation (UDA) learn domain-invariant representations to address the domain shift. Recently, self-training has been used in UDA, which exploits pseudo-labels for unlabeled target domains. However, the pseudo-labels can be unreliable due to distribution shifts between domains, severely impairing the model performance. To address this problem, we propose a novel self-training framework—Self-Training with Label-Feature-Consistency (ST-LFC), which selects reliable target pseudo-labels via label-level and feature-level voting consistency principle. The former means target pseudo-labels generated by a source-trained classifier and the latter means the nearest source-class to the target in feature space. In addition, ST-LFC reduces the negative effects of unreliable predictions through entropy minimization. Empirical results indicate that ST-LFC significantly improves over the state-of-the-arts on a variety of benchmark datasets.

**Keywords:** Transfer learning · Domain Adaptation · Self-Training · Label-Feature-Consistency

## 1 Introduction

Supervised deep learning methods have achieved excellent performance for tasks such as computer vision [14], natural language processing [15]. However, such models usually generalize poorly due to the distribution shifts between the training samples and testing samples. For example, in object recognition tasks, the model is generally trained on images collected in fine days, but the model's performance degrades when applying the model on rainy days and foggy days.

Unsupervised domain adaptation (UDA) is able to overcome this challenge by transferring knowledge from a labeled source domain to the unlabeled target domain [13, 35, 36]. Existing UDA could be divided into moment matching, adversarial domain adaptation, and self-training based methods. The first two

---

Y. Xin and S. Luo—Equal contribution.

types reduce domain discrepancy by reducing the statistical distribution discrepancy across domains [25,30] or perform domain adversarial training [13,27]. Different from these methods, self-training based methods [8,9,20] are inspired by semi-supervised learning (SSL) [10,19] and do not need to explicitly calculate domain discrepancy or perform complex adversarial training.

Typical self-training method firstly trains a model with labeled source samples, and then it generates the pseudo-labels for the unlabeled target samples using the source model. Considering that some pseudo-labels are incorrect, only pseudo-labels with high confidence are selected and then they are added to the training samples together with source samples to train the model, and repeat this process until the model converges. Recently, a few methods follow self-training framework and optimize this process by designing new selection strategy for target samples [21], introducing new regularization [10], and optimizing training strategy [16,17].

Although achieving remarkable progress, there are still some issues to be addressed. Firstly, self-training is a method used in semi-supervised learning, which means domain shift will not be prominent or even absent in the original application scenario. In domain adaptation, because of the existence of domain shift such as covariate shift and label shift, the distribution of pseudo-labels may be remarkably different from the target ground-truths, while purely tailoring self-training with UDA. As a result, the accuracy of pseudo-labels cannot be guaranteed without any regulars because of the accumulated error and even trivial solution, which eventually leads to misalignment of the distribution and misclassification of several classes [20]. Secondly, previous works tailor self-training for UDA by selecting pseudo-labels with confidence threshold or reweighting through information entropy or other confidence measurement methods, but it is challenging to determine a criterion for confidence judgment when it is closely related to specific tasks, causing models based on previous works are not robust. Additionally, existing self-training paradigms tend to ignore the rich information in unreliable samples. Since the ignoration of global distribution including either reliable or unreliable samples of the target domain, leading to a poor-fit between self-training and UDA, there is still a great room for improvement.

To address the above limitations, in this paper, we propose a simple yet effective method called ST-LFC (Self-Training with Label-Feature-Consistency). Instead of using confidence score to determine whether a sample is reliable to avoid generating pseudo-labels with hard-to-tweak protocols, we use label-level and feature-level consistency as a criterion. Intuitively, a successful feature extractor should generate features with greater inter-class and smaller intra-class distances. For example, during self-training process, one target sample is assigned class  $k$  by pseudo-labels, then it should be closer to source samples belong to class  $k$  and farther from the source samples which belong to other classes except for  $k$  in the feature space. Motivated by [24], we use source class prototype feature to represent the class, which means the average feature embedding of the source samples with the same ground-truth labels. Moreover, ST-LFC also exploits unreliable target samples. The reason why model has a performance

upper bound is the existence of these unreliable samples. The samples are unreliable because the classifier predictions are inconsistent with the nearest prototype class in the feature space, and we can't judge which is right. Our method weights the class prediction probabilities of both, constrained by entropy minimization. Finally, our method is able to incorporate any domain adaptation methods that takes into account both domain alignment and sample label utilization during training.

The effectiveness of ST-LFC is reflected by improved adaptation accuracy on popular benchmarks like Digits and Office-31 datasets. We also achieve state-of-the-art results on a challenging adaptation dataset Birds-31 [41], which indicates the usefulness of our ST-LFC in handling wide variety of scenarios.

In summary, the key highlights of the paper are:

- We propose a novel self-training framework ST-LFC for unsupervised domain adaptation. We propose a new sample strategy to effectively recognize reliable samples and we adequately utilize unreliable samples to improve the upper bound on model performance.
- ST-LFC is designed to be general for existing domain adaptation approaches. It is able to incorporate any domain adaptation methods. In order to verify, we combine SF-LFC with two popular methods, DANN and CDAN, and observe consistent improvement over both the baselines.
- We validate the effectiveness of the proposed approach numerically by applying it on multiple tasks from various challenging benchmark datasets used for domain adaptation like Digits, Office-31 and Birds-31 and observe improved accuracies in all the cases, sometimes outperforming the state-of-the-art by a large margin.

## 2 Related Work

### 2.1 Unsupervised Domain Adaptation

Unsupervised Domain Adaptation is proposed to address the domain shift between source domains and target domains, so that networks trained on source domain can be used directly on completely unlabeled target domains [34–36]. Motivated by theoretical bound proposed in [34], Discrepancy-based approaches [30–33] measure and minimize the dissimilarity between the feature embedding of the source and target domains. DAN [33] leverages the transferability of deep neural networks [22] and introduces MMD-based regularizer to minimize the cross-domain distribution discrepancy in multiple layer of neural networks. Adversary-based approaches [25–28] utilize adversarial learning [29] to align domain distribution on feature level and pixel level [25] by obtaining domain invariant features. Motivated by image translation techniques, pixel-level alignment methods [5–7] utilize image-to-image translation network to translate images from source domain to target domain. Feature-level alignment methods [1, 2, 4] tend to adapt distributions of source and target images explicitly in feature space. In addition to global domain adaptation, several studies take advantage of contrastive learning to align domain distribution on class-level [23, 24].

## 2.2 Self-Training

Self-Training [9, 10, 19] increases the amount of training data by iteratively selecting pseudo-labeled target data based on model which is trained with existing labeled data. Then retrain the model with the enlarged training data. Specifically, the quality of pseudo-labels of the target data and training strategy both have great impact on the performance of the model.

Existing methods can be generally categorized into three groups. The first group tend to design selection strategies for target samples [1, 3, 11, 21]. CAN [21] utilize domain discriminator to assign weights to target data and jointly trains the model with source and target data. The second group is introducing new regularization to assist domain adaptation. [3] uses asymmetric tritraining method to improve the accuracy of pseudo labels, which means that it utilizes three asymmetric classifiers, two networks are used to label unlabeled target samples, and one network is trained by the pseudo-labeled samples to obtain target-discriminative representations [12] first generates coarse pseudo labels by a conventional UDA method, and iteratively exploits the intra-class similarity of the target samples for improving the generated coarse pseudo labels. [10] employ confidence regularization technique to help discriminative feature representations of the source and target domains by introducing soft pseudo labels. The third group is optimizing training strategy. [17] proposed a class-balanced self-training framework to address the problem of imbalanced pseudo-labels. Motivated by co-training, CODA [16] tries to slowly adapt training set from the source domain to the target domain. [20] proposed Cycle Self-training which mainly focus on pseudo-labeling and learns to generalize the pseudo-labels across domains.

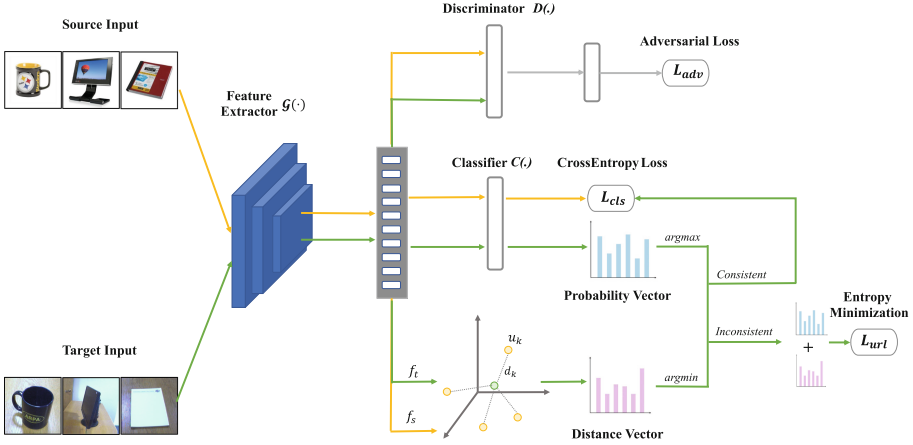
## 3 Method

In this section, we first give a brief overview of adversarial adaptation methods, and then introduce how to find nearest source-class in feature space. Finally, we explain our self-training with label-level and feature-level consistency method in detail.

### 3.1 Overview of Adversarial Domain Adaptation

In unsupervised domain adaptation, we are given a labeled source domain  $\mathcal{D}^s = \{x_i^s, y_i\}_{i=1}^{|\mathcal{D}^s|}$ , and an unlabeled target domain  $\mathcal{D}^t = \{x_i^t\}_{i=1}^{|\mathcal{D}^t|}$ . The source domain and target domain are characterized by probability distributions  $P_s$  and  $P_t$ , respectively. The goal of UDA is to train a model using  $\mathcal{D}^s$  and  $\mathcal{D}^t$  to make predictions on  $\mathcal{D}_t$ . Figure 1 shows the overall architecture of our proposed method. Feature extractor  $\mathcal{G}$  is shared by the source and target domains, extracting the low-dimensional feature representations corresponding to the inputs, given by  $f = \mathcal{G}(x)$ . The classifier  $\mathcal{C}$  then outputs the softmax prediction distribution over the classes, and is trained using the cross-entropy (CE) loss on the labeled data given by

$$\mathcal{L}_{cls} = \mathbb{E}_{(x,y) \sim \mathcal{D}^s \cup \mathcal{D}_{select}^t} [-\log[\mathcal{C}(\mathcal{G}(x))]]_y \quad (1)$$



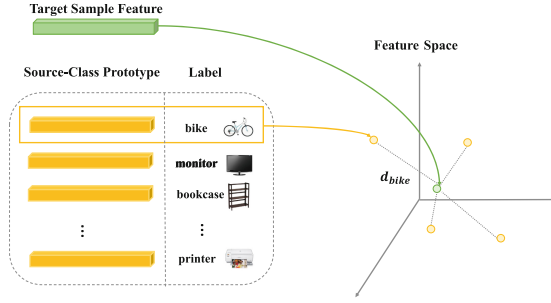
**Fig. 1.** Illustration of proposed ST-LFC approach. Our architecture consists of a feature extractor  $\mathcal{G}$  which is shared by source and target domains. The classifier  $\mathcal{C}$  is trained to classify the source images and generate target pseudo-labels using cross entropy loss  $\mathcal{L}_{cls}$ . The domain discriminator  $\mathcal{D}$  aims to achieve domain alignment using adversarial loss  $\mathcal{L}_{adv}$ . Additionally, we use source features to obtain the class feature prototype  $\mu$ , and compute the distance between target features and class feature prototype to get its nearest source-class. Then, we judge whether the generated pseudo-labels are consistent with the nearest source-class results. The consistent target samples are trained together with the source domain (Self-Training Process), and the inconsistent use  $\mathcal{L}_{url}$  to participate in training.

where  $y$  is the ground truth labels for the source data in  $D_s$  or pseudo-labels for carefully selected target data in  $D_t$ . This is a self-training process, the model generates pseudo-labels of unlabeled data, and jointly trains the model with source labels and selected target pseudo-labels. The target domain samples selection method will be introduced in the next section. However, since  $P_s \neq P_t$  and the classifier are biased to the source domain, the classifier does not generalize well to target samples. Thus, the adversarial learning strategy [28] is used to alleviate this issue. The domain discriminator  $D$  is trained by  $\mathcal{L}_{DC}$  to distinguish source samples from target samples, while the feature extractor  $\mathcal{G}$  is trained to generate domain-invariant features that can confuse the discriminator:

$$\min_D \max_{\mathcal{G}} \mathcal{L}_{DC} \tag{2}$$

$$\mathcal{L}_{DC} = -\mathbb{E}_{x \sim \mathcal{D}^s} [\log \mathcal{D}(\mathcal{G}(x))] - \mathbb{E}_{x \sim \mathcal{D}^t} [\log(1 - \mathcal{D}(\mathcal{G}(x)))] \tag{3}$$

The min-max training between feature extractor and discriminator can learn help domain-invariant features, but this is not enough for good adaptation. On the basis of learning domain-invariant features, our method utilizes the pseudo-labels of the target samples by self-training, which leads to a qualitative improvement in the performance on the target domain.



**Fig. 2.** Illustration of source-class prototypes. It means the average feature embedding of the source samples with the same ground-truth labels. We can find the nearest source-class for target sample in feature space.

### 3.2 Source-Class Prototype in Feature Space

Our source-class prototype definition is intentionally tailored to serve UDA task. Recall that during each forward pass, we have obtained source features  $f_i^s = \mathcal{G}(x_i^s)$ , which is generated by the feature extractor  $\mathcal{G}$ . We enqueue each  $(f_i^s, y_i^s)$  pair sequentially into each of the specific source-class prototypes, which means the average feature embedding of the source samples with the same ground-truth labels. The prototype of source-class  $k$  is denoted as:

$$\mu_k = \frac{1}{n_k} \sum_{y_i^s=k} f_i^s \quad (4)$$

where  $n_k$  represents the number of corresponding samples. Assuming that the source contains a total of  $\mathcal{K}$  categories,  $\mathcal{K}$  source-class prototypes can be obtained. Figure 2 shows the source-class prototype part in the overall framework in detail. Figure 2 takes target sample  $x_j^t$  as an example,  $\mathcal{K}$ -dimensional distance vectors  $d_j^t$  can be obtained by measuring the distance between target sample  $x_j^t$  feature and  $\mathcal{K}$  source-class prototypes, we adopt the calculation method of squared euclidean distance, which is given by:

$$d_{j,k}^t = \|\mathcal{G}(x_j^t) - \mu_k\|^2, k = \{1, 2, 3, \dots, \mathcal{K}\} \quad (5)$$

where  $d_{j,k}^t$  is the  $k^{th}$  element of  $d_j^t \in \mathbb{R}^{\mathcal{K}}$ . From this, we can obtain the feature distance vector between target samples and source-class prototypes. The nearest source-class has the smallest distance.

---

**Algorithm 1.** Self-training with Label-Feature-Consistency

---

**Require:** Batches for source  $B_s \in D_s$ ; Batches for target  $B_t \in D_t$ ; Feature extractor  $\mathcal{G}$ ; Classifier  $\mathcal{C}$ ; Domain discriminator  $\mathcal{D}$ .

**Ensure:** Trained feature extractor and trained classifier by ST-LFC.

```

1: for epoch=0 to MaxEpoch do
2:   for t=0 to MaxIter do
3:     Feature extractor  $\mathcal{G}$  generates features  $f_s$  for  $B_s$  and  $f_t$  for  $B_t$ .
4:     Classifier  $\mathcal{C}$  outputs probability vectors  $p_s$  for  $B_s$  and  $p_t$  for  $B_t$ .
5:     Source features  $f_s$  generate source-class prototypes by Equ 4.
6:     Calculate distance vectors from the source-class prototypes by Equ 5.
7:     Find reliable samples by Equ 6.
8:     if reliable sample then
9:       Train source feature extractor  $\mathcal{G}$  and source classifier  $\mathcal{C}$  by classification loss
          $\mathcal{L}_{rl}$ .
10:    else
11:      Train source feature extractor  $\mathcal{G}$  and source classifier  $\mathcal{C}$  by classification loss
         $\mathcal{L}_{url}$  by Equ 8.
12:    end if
13:    Train source feature extractor  $\mathcal{G}$  and source classifier  $\mathcal{C}$  by source classification
    loss  $\mathcal{L}_{cls}$  by Equ 1.
14:    Train source feature extractor  $\mathcal{G}$  by domain adversarial loss  $\mathcal{L}_{adv}$  by Equ 2.
15:  end for
16: end for
17: return  $\mathcal{G}, \mathcal{C}$ 

```

---

### 3.3 Self-training with Label-Feature-Consistency

We first develop a voting consistency strategy to select certain pseudo-labeled target samples for self-training to adapt the model to the target domain. The certainty is decided by the consistency of two different predictions: the classifier prediction and the nearest source-class prediction. The classifier  $\mathcal{C}$  outputs probability vectors  $p_j^t \in \mathbb{R}^{\mathcal{K}}$  for  $x_j^t$  and distance vector  $d_j^t \in \mathbb{R}^{\mathcal{K}}$  can be obtained which is introduced in last section. Then the consistency score for target sample is defined as:

$$\text{con}(x_j^t) = \begin{cases} 1 & \text{argmax}(p_j^t) == \text{argmin}(d_j^t) \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

where *argmax* obtains the label with the largest predicted probability by the classifier, *argmin* obtains the nearest source-class. For target samples with consistency score of one, we consider them to be reliable samples. Reliable samples with pseudo-labels train the model together with the source by cross-entropy loss  $\mathcal{L}_{rl}$ . This part plays an integral role in our ST-LFC. Directly training the model with reliable target samples with pseudo-labels will make the model perform better on the target.

Existing domain adaptation methods have an upper limit, in other words, the upper limit is caused by these unreliable samples. ST-LFC also makes use of the remaining unreliable samples. In our ST-LFC, the samples are unreliable because

the classifier predictions are inconsistent with the nearest prototype class in the feature space, and we can't judge which is right. The solution of our ST-LFC is to consider comprehensively, combining the probability vector  $p_j^t \in \mathbb{R}^{\mathcal{K}}$  and distance vector  $d_j^t \in \mathbb{R}^{\mathcal{K}}$ . However, the distance vector needs to be processed and converted into a class probability value, which is given by:

$$q_{j,k}^t = \frac{e^{-d_{j,k}^t}}{\sum_{i=1}^{\mathcal{K}} e^{-d_{j,i}^t}}, k = \{1, 2, 3, \dots, \mathcal{K}\} \quad (7)$$

where  $q_{j,k}^t$  is the  $k_{th}$  element of  $q_j^t$ , which means the probability of class  $k$  considering feature distance. Then we use entropy minimization to update model, which is defined as:

$$\mathcal{L}_{url} = - \sum_{k=1}^{\mathcal{K}} (p_{j,k}^t + q_{j,k}^t) \log(p_{j,k}^t + q_{j,k}^t) \quad (8)$$

### 3.4 Optimization

To sum up, the optimization of ST-LFC is mainly divided into four parts: source classification loss  $\mathcal{L}_{cls}$ , target reliable samples classification loss  $\mathcal{L}_{rl}$ ; entropy minimization two level probability of the target unreliable samples  $\mathcal{L}_{url}$  and domain adaptation loss  $\mathcal{L}_{adv}$ . Overall optimization of the model is given by:

$$\mathcal{L}_{total} = \mathcal{L}_{cls} + \mathcal{L}_{rl} + \mathcal{L}_{url} + \alpha \mathcal{L}_{adv} \quad (9)$$

where  $\alpha$  is the trade-off parameter. Algorithm 1 depicts the complete training procedure of ST-LFC.

## 4 Experiments

In this section, we conduct extensive experiments on multiple domain adaptation benchmarks to verify the effectiveness of ST-LFC. We present the datasets used to evaluate our results, baselines methods we compared against, followed by results and discussion. In the experiment, we choose well-known UDA methods DANN [49] and CDAN [13] as our infrastructure respectively.

### 4.1 Datasets

To test the effectiveness of our method, we experiment on three different kinds of benchmark datasets used for domain adaptation, which are Digits, Office-31 and Birds-31.

**Digits.** We investigate three digits datasets: USPS (U), MNIST (M), and SVHN (S). We show results on three transfer tasks:  $M \rightarrow U$ ,  $U \rightarrow M$ ,  $S \rightarrow M$ . USPS contains 7,438 images. MNIST is composed of 55,000 images and SVHN is composed of 73,257 images.



**Table 1.** Accuracy (%) on Digits for unsupervised domain adaptation.

METHOD	M→U	U → M	S → M	AVG.
Source Only	76.7	63.4	67.1	69.1
DANN	90.8	94.0	83.1	89.3
ADDA	89.4	90.1	76.0	85.2
DSN	91.3	-	82.7	-
ATT	-	-	85.0	-
CDAN	93.9	96.9	88.5	93.1
ST-LFC(with DANN)	92.9	97.3	91.5	93.9
ST-LFC(with CDAN)	<b>94.9</b>	<b>97.9</b>	<b>92.0</b>	<b>94.9</b>

*Office-31* is the most widely used dataset for visual domain adaptation, with 4,652 images and 31 categories collected from three distinct domains: Amazon (A), DSLR (D) and Webcam (W). We show results for all the 6 task pairs  $A \rightarrow W, D \rightarrow W, W \rightarrow D, A \rightarrow D, D \rightarrow A$  and  $W \rightarrow A$ . Following prior works, we report results on the complete unlabeled examples of the target domain.

*Birds-31* is recently proposed by [41] for fine grained adaptation consisting of different types of birds. There are three domains in Birds-31: CUB-200-2011 (C), NABirds (N) and iNaturalist2017 (I). The numbers of images selected are 1,848, 2,988 and 2,857 respectively. We show the adaptation results on six transfer tasks formed from three domains:  $C \rightarrow I, I \rightarrow C, I \rightarrow N, N \rightarrow I, C \rightarrow N$  and  $N \rightarrow C$ .

## 4.2 Setup

*Baselines.* We compare our method ST-LFC with state-of-art domain adaptation methods: DAN [42], DAA [45], CDAN [13], CAT [37], ALDA [38] and SRDA [39] as well as works which perform class aware alignment such as MCD [46], SimNet [47], MADA [40]. For Birds-31, we additionally verify our result with prior fine grained adaptation work, PAN [41]. Finally, we have ST-LFC with DANN, which is using ST-LFC approach on top of DANN and ST-LFC with CDAN which uses ST-LFC in combination with CDAN. We compare the task-wise accuracy and report the average accuracy across all the transfer tasks.

*Implementation.* We implement our method on Pytorch, we use DTN [13] architecture for digits and ResNet-50 [48] pretrained on ImageNet as the feature extractor for Office-31 and Birds-31. The classifier is made up of fully connected layers. For achieving training stability, we observe that it is essential to pretrain the model on the labeled source dataset for a few iterations before self-training process. We use mini-batch SGD with a learning rate of 0.03 for Office-31 and Birds-31. For the classifier we multiply the learning rate by 10. We use a similar annealing strategy as used in [49].

To illustrate the benefits of the proposed ST-LFC, we employ it on top of two competing adaptation benchmarks in DANN [49] and CDAN [13], while noting

**Table 2.** Accuracy (%) of different unsupervised domain adaptation methods on Office-31 using ResNet-50 as the backbone for 6 transfer tasks among three domains: Amazon (A), Webcam (W) and Dslr (D). Our method shows consistent improvements.

METHOD	A→W	D→W	W→D	A→D	D→A	W→A	AVG.
ResNet50	68.4	96.7	99.3	68.9	62.5	60.7	76.1
DAN	80.5	97.1	99.6	78.6	63.6	62.8	80.4
DANN	82.0	96.9	99.1	79.7	68.2	67.4	82.2
ADDA	86.2	96.2	98.4	77.8	69.5	68.9	82.9
MCD	88.6	98.5	100.0	92.2	69.5	69.7	86.5
SimNet	88.6	98.2	99.7	85.3	73.4	71.8	86.2
CDAN	93.1	98.2	100.0	89.8	70.1	68.0	86.6
CDAN+E	94.1	98.6	100.0	92.9	71.0	69.3	87.7
DAA	86.8	99.3	100.0	88.8	<b>74.3</b>	73.9	87.2
SAFN	88.8	98.4	99.8	87.7	69.8	69.7	85.7
MADA	90.0	97.4	99.6	87.8	70.3	66.4	85.2
CAT	94.4	98.0	100.0	90.8	72.2	70.2	87.6
ALDA	<b>95.6</b>	97.7	100.0	<b>94.0</b>	72.2	72.5	88.7
SRDA	95.2	98.6	100.0	91.7	74.5	73.7	89.0
ST-LFC(with DANN)	91.3	98.6	100.0	88.6	72.1	69.9	86.8
ST-LFC(with CDAN)	95.2	<b>99.3</b>	<b>100.0</b>	93.4	73.6	<b>75.7</b>	<b>89.5</b>

that our ST-LFC is general and applicable in combination with any adversarial adaptation approach. For experiments with DANN, we replace the adversarial loss with a gradient reversal layer.

### 4.3 Results

**Digits Dataset.** In Table 1, we show the results for adaptation using ST-LFC. We observe that we outperform prior methods when we use CDAN or DANN in combination with ST-LFC. ST-LFC with DANN improves average accuracy from 89.3% to 93.9% and ST-LFC with CDAN improves average accuracy from 93.1% to 94.9%, indicating the usefulness of ST-LFC for improving existing methods for domain adaptation. To better illustrate performance of ST-LFC, the results for office-31 and birds-31 are describe below.

**Office-31 Dataset.** We present results on the 6 transfer tasks on Office-31, including their average, in Table 2. We observe that we achieve an accuracy of 89.5% on the average (ST-LFC with CDAN), outperforming all the competing baselines. However, the  $A \rightarrow W$  and  $A \rightarrow D$  are slightly less effective than ALDA, and  $D \rightarrow A$  is less effective than DAA. This is because we chose to build on DANN and CDAN, they are widely known UDA methods. Finally, our ST-LFC is generally applicable, it improves accuracy over both the approaches

**Table 3.** Results for domain adaptation on fine-grained adaptation setting, shown for 3 challenging datasets: CUB-200-2011 (C), iNaturalist2017 (I) and NABirds (N). Our ST-LFC performs consistently better than all other methods by explicitly modeling the finegrained nature of the adaptation process. All the baseline numbers taken from [41].

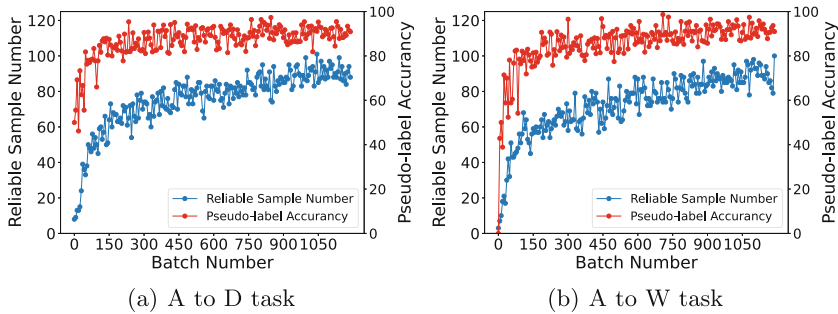
Method	C→I	I→C	I→N	N→I	C→N	N→C	Avg.
ResNet50	64.25	87.19	82.46	71.08	79.92	89.96	79.14
DAN	63.90	85.86	82.91	70.67	80.64	89.40	78.90
DANN	64.59	85.64	80.53	71.00	79.37	89.53	78.44
JAN	63.69	86.29	83.34	71.09	81.06	89.55	79.17
ADDA	63.03	87.26	84.36	72.39	79.69	89.28	79.33
MADA	62.03	89.99	87.05	70.99	81.36	92.09	80.50
MCD	66.43	88.02	85.57	73.06	82.37	90.99	81.07
CDAN	68.67	89.74	86.17	73.80	83.18	91.56	82.18
SAFN	65.23	90.18	84.71	73.00	81.65	91.47	81.08
PAN	69.79	90.46	88.10	75.03	84.19	92.51	83.34
ST-LFC(with DANN)	68.67	92.32	87.55	74.94	83.00	92.53	83.17
ST-LFC(with CDAN)	<b>71.47</b>	<b>92.80</b>	<b>90.36</b>	<b>76.82</b>	<b>85.37</b>	<b>94.53</b>	<b>85.23</b>

DANN [49] and CDAN [13], consistently over all the tasks (by 4.6% and 2.9% on average, respectively). It is worth noting that ST-LFC improves the accuracy of the DANN method on  $A \rightarrow W$  task by 9.3% and  $A \rightarrow D$  task by 8.9%.

**Birds-31 Dataset.** The difficulty in this setting lies in the fact that birds from same class but different domains look quite distinct, sometimes more different than images from an other class. We verify the results on all 6 transfer tasks on Birds-31 dataset in Table 3, and show that ST-LFC outperforms prior works across all the tasks. From the result, prior works that rely on global alignment objectives [36, 42] do not perform any better than a source-only model (ResNet-50 baseline), possibly because they suffer from negative alignment. However, our ST-LFC directly considers pseudo-labels, which is a more fine-grained solution for Birds-31. As a result, we improve the accuracy over DANN on all the tasks, and average accuracy from 78.44% to 83.17%. In fact, with an average accuracy of 85.23% we achieve the new state-of-the-art result using ST-SLC in combination with CDAN. More remarkably, ST-LFC even outperform PAN [41], that is specifically designed for fine-grained adaptation. This result underlinesthat ST-LFC is able to perform well on fine-grained vision categorization despite the domain shift.

#### 4.4 Insight Analysis

**Ablation Study.** A highlight of ST-LFC compared with other methods based on Self-Training is that it utilizes both carefully selected reliable target and con-



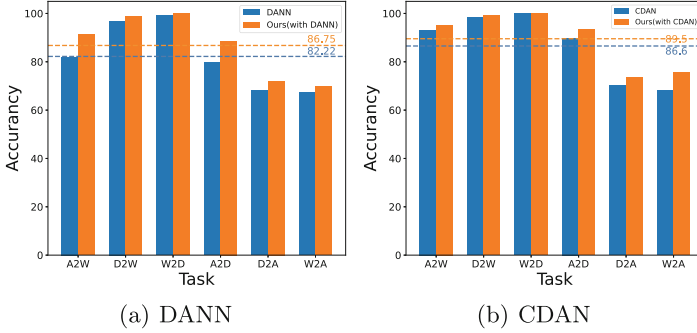
**Fig. 3.** (a) shows the number of selected samples and the accuracy of pseudo-labels during training in A to D task; (b) shows the number of selected samples and the accuracy of pseudo-labels during training in A to W task.

**Table 4.** Results for ablation study. To verify the effectiveness of the ST-LFC method, we experiment with DANN, ST-LFC(without exploiting unreliable samples) and our ST-LFC(with both) on Office-31 and Birds-31.

<i>Reliable</i>	<i>Unreliable</i>	Office-31	Birds-31
✗	✗	82.2	78.4
✓	✗	85.5	82.5
✓	✓	<b>86.8</b>	<b>83.2</b>

straints on unreliable target. We design experiments on the Office31 and Birds-31 datasets to verify the importance of these two parts, and the results are shown in Table 4. The experiment is divided into three parts: DANN method, ST-LFC(with DANN) without using unreliable target, our ST-LFC(with DANN). ST-LFC without using unreliable target is improved on the basis of DANN methods by 3.3% and 4.1% respectively, which shows that ST-LFC is effective for the utilization of reliable target. In addition, the performance of ST-LFC is improved after adding constraints on the unreliable target by 1.3% and 0.7% respectively, which demonstrates the effectiveness of unreliable target exploitation.

**Target Pseudo-Labels Analysis.** For all methods based on self-training, the accuracy of pseudo-labels and the number of selected samples are the key factors affecting the performance of the method, and our method is no exception. Figure 3 shows the accuracy rate of pseudo-labels and the variation in the number of samples selected per batch during ST-LFC training. We can see that as the number of ST-LFC training rounds continues to increase, the number of target samples selected in each batch also increases. Each batch of this experiment contains 124 samples, from which a maximum of about 110 reliable samples can be selected in both  $A \rightarrow D$  and  $A \rightarrow W$  task, and the pseudo-label accuracy of reliable samples can reach about 90% with increasing training. In this way, our ST-LFC is very successful in the selection of reliable target samples, and not only performs well in the selection accuracy, but also selects a sufficient number of samples.



**Fig. 4.** (a) shows the results of ST-LFC with DANN and DANN comparison; (b) shows the results of ST-LFC with CDAN and CDAN comparison.

**Generality Analysis.** In order to analysis ST-LFC can be applied to enhance any existing domain adaptation approach, we combine two well-known domain adaptation methods DANN and CDAN as examples. We plot the improvement of Office-31 by our ST-LFC, as shown in Fig. 4. The previous results section mainly describes the experimental data, while this part mainly analyzes the results. There are six tasks in Office-31, and it can be seen from the Fig. 4 that our ST-LFC has a certain magnitude of performance improvement for each task. The performance improvement for DANN is better, because the upper limit of DANN is lower than CDAN, the improvement space of CDAN is smaller. In addition, among the six tasks, A to W and A to D have the best improvement effect, which also shows that ST-LFC is friendly to tasks with large improvement space. In fact, ST-LFC can combine any state-of-the-art methods and can further improve performance.

## 5 Conclusion

In this work, we propose Self-Training with Label-Feature-Consistency(ST-LFC), which is designed to be general and can be applied to enhance any existing adaptation approach. Firstly, we design a new selection strategy for reliable target samples, which uses label-level and feature-level voting consistency principle. This selection strategy lays the foundation for ST-LFC to perform well in UDA problems. Secondly, ST-LFC does not adopt the abandonment strategy for unreliable samples, but using entropy minimization to constrain the class probability of two levels. Finally, the combination of ST-LFC and adaptation methods can both enable domain alignment and exhibit the powerful advantages of self-training. We show numerical results on various challenging benchmark datasets and perform favorably against many existing adaptation methods.

**Limitations and Future Work.** Although our ST-LFC performs well on UDA problems, it is not sufficiently applicable. Recently, Test Time Adaptation [50] has been proposed, which uses source to train the model and uses unlabeled

target to adjust model only when testing. The scenario solution of this problem is similar to self-training, and we hope that ST-LFC can go a step further in this problem.

**Acknowledgements.** This paper is supported by the National Key Research and Development Program of China (Grant No. 2018YFB1403400), the National Natural Science Foundation of China (Grant No. 62192783, 61876080), the Key Research and Development Program of Jiangsu (Grant No. BE2019105), the Collaborative Innovation Center of Novel Software Technology and Industrialization at Nanjing University.

## References

1. Wang, Z., et al.: Differential treatment for stuff and things: a simple unsupervised domain adaptation method for semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2020)
2. Du, L., et al.: SSF-DAN: separated semantic feature based domain adaptation network for semantic segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2019)
3. Saito, K., Ushiku, Y., Harada, T.: Asymmetric tri-training for unsupervised domain adaptation. In: International Conference on Machine Learning. PMLR (2017)
4. Zhang, Yi., et al. Fully convolutional adaptation networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2018)
5. Murez, Z., et al.: Image to image translation for domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2018)
6. Li, Y., Yuan, L., Vasconcelos, N.: Bidirectional learning for domain adaptation of semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2019)
7. Chen, Y.C., et al.: Crdoco: pixel-level domain transfer with cross-domain consistency. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2019)
8. Li, B., et al.: Rethinking distributional matching based domain adaptation. arXiv preprint [arXiv:2006.13352](https://arxiv.org/abs/2006.13352) (2020)
9. Kumar, A., Ma, T., Liang, P.: Understanding self-training for gradual domain adaptation. In: International Conference on Machine Learning. PMLR (2020)
10. Zou, Y., et al.: Confidence regularized self-training. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2019)
11. Chen, C., et al.: Progressive feature alignment for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2019)
12. Wang, J., Zhang, X.L.: Improving pseudo labels with intra-class similarity for unsupervised domain adaptation. arXiv preprint [arXiv:2207.12139](https://arxiv.org/abs/2207.12139) (2022)
13. Long, M., et al.: Conditional adversarial domain adaptation. In: Advances in Neural Information Processing Systems, vol. 31 (2018)
14. Voulodimos, A., et al.: Deep learning for computer vision: a brief review. *Comput. Intell. Neurosci.* 2018 (2018)
15. Torfi, A., et al.: Natural language processing advancements by deep learning: A survey. arXiv preprint [arXiv:2003.01200](https://arxiv.org/abs/2003.01200) (2020)

16. Chen, M., Weinberger, K.Q., Blitzer, J.: Co-training for domain adaptation. In: *Advances in Neural Information Processing Systems*, vol. 24 (2011)
17. Zou, Y., et al.: Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In: *Proceedings of the European conference on computer vision (ECCV)* (2018)
18. Riloff, E., Wiebe, J., Wilson, T.: Learning subjective nouns using extraction pattern bootstrapping. In: *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003* (2003)
19. Prabhu, V., et al.: Sentry: selective entropy optimization via committee consistency for unsupervised domain adaptation. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021)
20. Liu, H., Wang, J., Long, M.: Cycle self-training for domain adaptation. *Adv. Neural. Inf. Process. Syst.* **34**, 22968–22981 (2021)
21. Zhang, W., et al.: Collaborative and adversarial network for unsupervised domain adaptation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018)
22. Yosinski, J., et al.: How transferable are features in deep neural networks?. In: *Advances in Neural Information Processing Systems*, vol. 27 (2014)
23. Sharma, A., Kalluri, T., Chandraker, M.: Instance level affinity-based transfer for unsupervised domain adaptation. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2021)
24. Chen, Y., et al.: Transferrable contrastive learning for visual domain adaptation. In: *Proceedings of the 29th ACM International Conference on Multimedia* (2021)
25. Bousmalis, K., et al.: Unsupervised pixel-level domain adaptation with generative adversarial networks. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017)
26. Bousmalis, K., et al.: Domain separation networks. In: *Advances in Neural Information Processing Systems*, vol. 29 (2016)
27. Cui, S., et al.: Gradually vanishing bridge for adversarial domain adaptation. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020)
28. Ganin, Y., et al.: Domain-adversarial training of neural networks. *J. Mach. Learn. Res.* **17**(1), 2030–2096 (2016)
29. Goodfellow, I., et al.: Generative adversarial networks. *Commun. ACM* **63**(11), 139–144 (2020)
30. Kang, G., et al.: Contrastive adaptation network for unsupervised domain adaptation. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019)
31. Gretton, A., et al.: A kernel method for the two-sample-problem. In: *Advances in Neural Information Processing Systems*, vol. 19 (2006)
32. Zhang, X., et al.: Deep transfer network: unsupervised domain adaptation. *arXiv preprint [arXiv:1503.00591](https://arxiv.org/abs/1503.00591)* (2015)
33. Long, M., et al.: Learning transferable features with deep adaptation networks. In: *International Conference on Machine Learning*. PMLR (2015)
34. Ben-David, S., et al.: Analysis of representations for domain adaptation. In: *Advances in Neural Information Processing Systems*, vol. 19 (2006)
35. Ben-David, S., et al.: A theory of learning from different domains. *Mach. Learn.* **79**(1), 151–175 (2010)
36. Long, M., et al.: Deep transfer learning with joint adaptation networks. In: *International Conference on Machine Learning*. PMLR (2017)

37. Deng, Z., Luo, Y., Zhu, J.: Cluster alignment with a teacher for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2019)
38. Chen, M., et al.: Adversarial-learned loss for domain adaptation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34. no. 04 (2020)
39. Wang, S., Zhang, L.: Self-adaptive re-weighted adversarial domain adaptation. arXiv preprint [arXiv:2006.00223](https://arxiv.org/abs/2006.00223) (2020)
40. Pei, Z., et al.: Multi-adversarial domain adaptation. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
41. Wang, S., et al.: Progressive adversarial networks for fine-grained domain adaptation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2020)
42. Jia, Y., et al.: Caffe: convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM International Conference on Multimedia (2014)
43. Long, M., et al.: Unsupervised domain adaptation with residual transfer networks. In: Advances in Neural Information Processing Systems, vol. 29 (2016)
44. Sankaranarayanan, S., et al.: Generate to adapt: aligning domains using generative adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2018)
45. Kang, G., et al.: Deep adversarial attention alignment for unsupervised domain adaptation: the benefit of target expectation maximization. In: Proceedings of the European Conference on Computer Vision (ECCV) (2018)
46. Saito, K., et al.: Maximum classifier discrepancy for unsupervised domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2018)
47. Pinheiro, P.O.: Unsupervised domain adaptation with similarity learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2018)
48. He, K., et al.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
49. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: International Conference on Machine Learning. PMLR (2015)
50. Sun, Y., et al.: Test-time training with self-supervision for generalization under distribution shifts. In: International Conference on Machine Learning. PMLR (2020)