Front. Comput. Sci., 2025, 19(3): 193310
https://doi.org/10.1007/s11704-024-3810-0

RESEARCH ARTICLE

Robust domain adaptation with noisy and shifted label distribution

Shao-Yuan LI (✉), Shi-Ji ZHAO, Zheng-Tao CAO, Sheng-Jun HUANG, Songcan CHEN

MIIT Key Laboratory of Pattern Analysis and Machine Intelligence, College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China

Abstract Unsupervised Domain Adaptation (UDA) intends to achieve excellent results by transferring knowledge from labeled source domains to unlabeled target domains in which the data or label distribution changes. Previous UDA methods have acquired great success when labels in the source domain are pure. However, even the acquisition of scare clean labels in the source domain needs plenty of costs as well. In the presence of label noise in the source domain, the traditional UDA methods will be seriously degraded as they do not deal with the label noise. In this paper, we propose an approach named Robust Self-training with Label Refinement (RSLR) to address the above issue. RSLR adopts the self-training framework by maintaining a Labeling Network (LNet) on the source domain, which is used to provide confident pseudo-labels to target samples, and a Target-specific Network (TNet) trained by using the pseudo-labeled samples. To combat the effect of label noise, LNet progressively distinguishes and refines the mislabeled source samples. In combination with class re-balancing to combat the label distribution shift issue, RSLR achieves effective performance on extensive benchmark datasets.

Keywords unsupervised domain adaptation, label noise, label distribution shift, self-training, class rebalancing

1 Introduction

Deep neural networks (DNNs) have achieved great success in various tasks. However, training a deep model often requires huge precise annotations and is thus costly. Domain Adaptation (DA) transfers knowledge from one or more datasets called source domain with a large number of labeled examples to another different but correlated dataset called target domain [1,2].

As a particular case of DA, Unsupervised Domain Adaptation (UDA) merely leverages supervision in the source domain (i.e., clean source labeled data) to train a model that generalizes well to the unlabeled target domain (i.e., unlabeled target data) which saves huge labeling costs. Since unlabeled data in the target domain can be easily obtained, UDA presents great potential in realistic scenes. UDA has received great attention and achieved wide applications in fields such as action recognition [2], medical imaging segmentation [3], and automatic speech recognition [4].

Theoretically, UDA aims to minimize the $\mathcal{H}\Delta\mathcal{H}$-divergence distance measure between the two domains [5,6]. For this purpose, various methods have been developed which can be into three categories: minimizing a measurement of distributional discrepancy like MMD metric through discrepancy-based methods [7,8] or learning domain-invariant representations through adversarial-based methods [9–11]; generating samples similar to the target domain data through GAN [12–14] or enabling source feature augmentation towards the target data in an implicit manner [15]; and self-training based methods which predict pseudo-labels for the target domain [16,17].

Although previous UDA methods have made significant successes, they present a harsh requirement that the labels in the source domain must be pollution-free which easily breaks down in practice. Since high-quality annotations often require high costs, some low-cost platforms like crowdsourcing systems have emerged [18]. However, the labels obtained from the low-cost platforms are often noisy, which brings great challenges to the model learning process. Label noise will ruin the performance of classification and even generate a negative effect on the transfer process [19].

Hence, robust UDA with noisy labels has attracted much attention. Few works have explored this by filtering out the noisy samples in the source domain as well as minimizing the distribution discrepancy [19–22]. [19] presented an end-to-end approach that can directly transfer knowledge from the noisy source domain to the unsupervised target domain with a co-teaching branching network structure. [20,21] overcame the noisy label issue by using a two-step solution, which first filters the noise factors through curriculum learning, and then transfers the knowledge from clean source data to the target domain by adversarial training rather than pseudo labeling. Different from [19–21] that exploited unidirectional relationships from the source domain to the target domain, [22] proposed to exploit bilateral relationships between the

Received October 11, 2023; accepted January 31, 2024

E-mail: lisy@nuaa.edu.cn

two domains. It took the two domains as different inputs to train two models alternately. It used a symmetrical Kullback-Leibler loss to selectively match the predictions of the two models in the same domain. This interactive learning schema enables implicit label noise canceling and exploits correlations between the source and target domains. Despite the great success achieved, previous robust UDA methods have two drawbacks. First, they check out noisy samples and merely learn on the subset of high-quality clean samples, and the recycling of low-quality noisy samples is not considered. Thus they would fail in the case of a high noise ratio because the clean samples are so limited. Second, most of them focus on covariate shift across domains, but ignore the label distribution shift (LDS) issue, which means class distribution changes across domains and can lead to biased learning.

Figure 1 shows an illustrative example. Here triangle and circle denote two different classes. The green dashed line represents the ideal classification boundary which perfectly separates the triangle and circle samples for both domains. Orange and blue respectively represent the clean source samples and the target samples. In noisy UDA, the source domain is corrupted with red mislabeled samples, e.g., the three boundary source samples 1, 2, 3. Besides, the two domains also face label distribution shift, e.g., in the source domain, the circle class is the majority class, whereas, in the target domain, the triangle class is the majority. Previous robust UDA methods alleviate the label noise by filtering out the mislabeled source samples, which lose important information about the boundary samples. Besides, they ignore LDS. As shown in the left of Fig. 1, their label predictions for unlabeled target samples tend to be dominated by and biased towards the majority circle class in the source domain (the black classification boundary). However, in the target domain, the triangle class is the majority and under-fitted.

To mitigate the above issues, we propose a novel method called Robust Self-training with Label Refinement (RSLR). Rather than discard the mislabeled source samples, we propose to correct them thus keeping the essential information of the critical boundary samples. Moreover, to ensure that the label distributions of both domains are aligned and the learned classification boundary is not biased towards any class, as in the right of Fig. 1, balanced pseudo-labeling predictions for both domains are kept during the learning process. Specifically, RSLR adopts the self-training idea [16] by training a Labeling Network (LNet) on the source domain, and a Target-specific Network (TNet) on pseudo-labeled target samples predicted by LNet, thus obtaining target-discriminative representations. To combat the source label noise, LNet borrows ideas from noisy label learning and semi-supervised learning, i.e., LNet first distinguishes the clean and noisy source samples by modeling their loss distribution using a Gaussian Mixture Model, then generates and retains the reliable pseudo-labels for noisy samples. RSLR further incorporates class distribution re-balancing to make the model train equally in each class. We compare our RSLR with previous robust UDA methods and highlight the key differences in Table 1.

To validate the effectiveness of RSLR, we conduct extensive experiments under various noise levels. Empirical results demonstrate that RSLR can robustly transfer knowledge from the source domain to the target domain, and achieves better performance with higher noise levels.

Contributions In summary, we have the following contributions:

- We propose RSLR, a robust self-training algorithm for dealing with noisy UDA scenarios. We propose to reuse

Table 1 Comparison of the proposed RSLR approach and previous robust UDA methods

Methods	Adversarial training	Pseudo vlabeling	Using noisy samples	Considering LDS	Branching structure
Butterfly [19]		✓			✓
TCL [20]	✓				
RDA [21]	✓			✓	
GearNet [22]	✓	✓			✓
RSLR		✓	✓	✓	✓

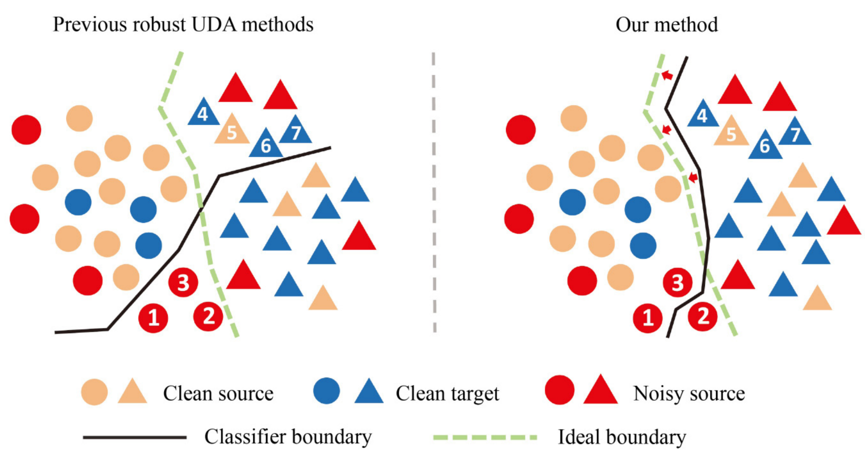


Fig. 1 Illustration of the challenge in noisy UDA. **Left:** Previous robust UDA methods filter out the mislabeled source samples, thus losing important class boundary information, e.g., samples 1, 2, 3. Besides, they ignore the label distribution shift issue and tend to be biased towards the majority circle class in the source domain (the black classification boundary). But in the target domain, the triangle majority class is under-fitted. **Right:** We propose keeping the essential information of the mislabeled samples and learning classification boundary not biased towards any class

the noisy source domain data to improve the generalization performance instead of discarding them. Meanwhile, we consider label distribution shift and covariate shift simultaneously.

- We propose a novel strategy by judging the consistency of model predictions on weakly augmentation and strongly augmentation of samples and solve the LDS issue through re-balanced pseudo-labeling in both domains.
- RSLR achieves effective performance on Digits, Office-31, and Office-Home on various noise ratio tasks especially high noise levels.

2 Related works

Unsupervised domain adaptation (UDA) UDA aims to leverage supervision information in some source domain to train a model that generalizes well to a different unlabeled target domain [5]. Different from traditional supervised learning, there are marginal distribution changes between the labeled source domain and unlabeled target domain which is called covariate shift. Current UDA methods mainly focus on narrowing the domain gap from different perspectives. One classical idea is by directly minimizing some discrepancy measurement between domains, e.g., maximum mean discrepancy [23], Wasserstein distance [24]. Recently, domain-adversarial learning has become a prominent paradigm [9–11,25–27]. These methods try to learn feature representations for the source and target domains in an adversarial manner, such that data distributions for different domains are hard to distinguish. Some work explored transforming the source data to resemble the target data using generative models [13,15], e.g., inspired by the fact that deep feature transformations towards a certain direction can be represented as meaningful semantic in the raw feature space, TSA (Transferable Semantic Augmentation) [15] generated source data towards the target semantic direction. [28] derived novel insights to show that a mixup between original and corresponding translated generic samples enhances the discriminability-transferability tradeoff while duly respecting the privacy-oriented source-free setting.

Another major direction is pseudo-labeling methods, which produce pseudo-labels for the target data and learn from them to obtain target-discriminative representation and classifier. Especially, ATDA (Asymmetric Tri-training for unsupervised Domain Adaptation) [16] constructed three networks asymmetrically for confident pseudo-labeling and training. CST (Cycle Self-Training) [29] cycled between a forward step and a reverse step to enforce pseudo-labels generalizable across domains. SENTRY (Selective Entropy Optimization) [17] selectively minimized pseudo-label prediction entropy on consistent target samples and maximized entropy on inconsistent samples.

Label noise learning Noisy annotations are inevitable and degrade model performance. In the noisy label learning field, previous work has shown an interesting phenomenon that deep neural networks fit easy samples first, and then gradually memorize hard samples [30]. Inspired by this, some methods tend to treat small-loss ones as clean samples [31–33]. Among these methods, co-teaching [31] and co-teaching+ [34] performed sample selection by using two networks, each trained on a subset of small-loss samples selected by the other network in every mini-batch. DivideMix [33] utilized two networks to select small-loss samples as clean ones through modeling a two-component mixture model. [35] proposed a novel contrastive regularization function to learn representations over noisy data where label noise does not dominate the representation learning. This paper deals with the more complex and challenging problem of robust UDA with noisy source labels.

Noisy unsupervised domain adaptation Robust UDA in a noisy environment is a practical and under-explored challenging problem. TCL (Transferable Curriculum Learning) [20] proposed one online curriculum learning method to filter out noisy samples in the source domain; RDA (Robust Domain Adaptation) [21] improved [20] by using offline curriculum learning and minimizing the distribution discrepancy. The above two methods adopted a two-step strategy that checks noisy samples first and aligns margin distribution across domains later. Different from the above two methods, [19] aimed to check out the noisy labels through a co-teaching network structure in an end-to-end manner. GearNet [22] took the two domains as different inputs to train two models alternately and used a symmetrical Kullback-Leibler loss to selectively match the predictions of the two models in the same domain. This interactive learning schema exploits bilateral relationships between the two domains and enables implicit label noise canceling.

Although these works could handle more realistic scenarios, they merely filter out the noisy samples and only train on the identified clean samples, whereas the full recycling of noisy sample information is not considered. Meanwhile, they ignore the label distribution shift issue. We will show that the above methods fail at a high noise rate because the clean samples are so limited.

3 The proposed approach

In this section, we first give the preliminaries used in this paper in Table 2, then introduce the details about RSLR. Presented below is the summary table featuring the most frequently used notations in our paper.

3.1 Preliminaries

We use $\mathcal{X} \subset \mathbb{R}^d$, $\mathcal{Y} = \{1,2,\ldots,K\}$ to denote the instance and label space respectively. In UDA with corrupted source labels, a set of source domain instances $D_s = \{(x_{si}, \tilde{y}_{si})\}_{i=1}^{N_s}$ drawn from distribution $\tilde{P}_s(X,Y)$ and unlabeled test target domain

Table 2 Summary of the key notations used in the paper

$\mathcal{X}$	Instance space	$\mathcal{Y}$	Label space
P_s	Source distribution	P_t	Target distribution
D_s	Source data	D_t	Target data
G	Feature extractor	C_1	Common classifier 1
C_2	Common classifier 2	C_t	Target classifier
$\mathcal{A}(x)$	Strong augmentation	$\alpha(x)$	Weak augmentation

instances $D_t = \{(x_{ti})\}_{i=1}^{N_t}$ from distribution $P_t(X)$ are available. For each source instance x_s, its noisy label $\tilde{y}_s$ is observed whereas its clean label y_s is unknown. $\tilde{P}_s(X,Y)$ means the corrupted version of the clean source domain distribution $P_s(X,Y)$. Note that in UDA, the source and target domain distributions are different, i.e.,$P_t(X,Y) \neq P_s(X,Y)$.

Previous work [19–21] have theoretically and empirically shown that, without handling the source label noise, existing UDA methods cannot well transfer useful knowledge from the noisy source data to unlabelled target data. They have proposed methods by combining techniques from the label noise learning field and domain distribution matching for the noisy UDA problem.

In this paper, similar to previous works, we propose to combine techniques in label noise learning to distinguish between the clean and noisy samples in the source domain. Furthermore, we take advantage of semi-supervised learning to better use the low-quality noisy samples, instead of simply eliminating them. Besides, we also propose a class distribution rebalancing strategy to deal with the commonly occurring LDS issue in DA, which occurs often and can severely accumulate, but was ignored by previous robust UDA works.

3.2 Robust self-training with label refinement

In principle, we build up our RSLR approach based on the self-training procedure to iteratively correct the noisy source samples and predict pseudo-labels for target samples. Figure 2 shows the overall structure of RSLR. Three sub-nets C_1, C_2, and C_t are designed. C_1, C_2 are Labeling Networks (LNets) trained to correct noisy labels and predict target sample pseudo-labels in an ensemble way. C_t is the Target-specific Network (TNet) trained on pseudo-labeled target samples and acts as the final desired target classifier. G denotes the feature extractor network that outputs shared features for C_1, C_2, and C_t. Note that two LNets C_1, C_2 are exploited to filter errors for each other thus avoiding the confirmation bias of individual network [36].

By sharing the feature extractor network G, and training C_1, C_2 on the combination of source and pseudo-labeled target samples to obtain target discriminative representations, RSLR implicitly solves the covariate shift issue between source and target domain. To cope with the label noise in the source domain, RSLR trains the two LNets C_1, C_2 with three major purposes: 1) distinguishing between the clean and noisy source samples; 2) refining the labels for noisy source samples; and 3) providing pseudo-labels for the target data. To cope with the LDS issue between domains, RSLR further incorporates class distribution rebalancing for the source and target pseudo-labels. In the following, we explain the details.

Distinguishing clean & noisy samples Following [33], we initially train the feature extractor network G and LNets C_1, C_2 on D_s using the regular cross entropy loss with early stop:

$$L_s = -\sum_{i=1}^{N_s} \ell_i = -\sum_{i=1}^{N_s}\sum_{k=1}^{K} \tilde{y}_{si} \log p^k_{model}(x_{si}). \tag{1}$$

Then we fit a two-component Gaussian Mixture Model (GMM) [37] to the per-sample prediction losses of each LNet, and use the posterior probability $p(g|\ell_i)$ over each LNet's loss on instance i as its clean probability w_i, where g is the Gaussian component with a smaller loss. Then by setting a threshold τ on w_i, each LNet C_1, C_2 divides the source data D_s into a clean set and a noisy set $D_s = \{D_s^{c1}, D_s^{n1}\}, D_s = \{D_s^{c2}, D_s^{n2}\}$.

The idea of using 2 components GMM to model and separate the loss distributions of clean and noisy labels is motivated by the memorization effect of DNNs, i.e., they fit easy (clean) patterns first, then gradually overfit hard (noisy) patterns. Thus with early stopped training, the small (large) loss samples can be identified as clean (noisy) ones. For scenarios where the noisy samples may exhibit complex patterns such as instance-dependent noise, as long as the clean samples patterns are more evident and fitted faster, the loss gap between clean and noisy samples is still meaningful to instruct the 2 components GMM. Works on classic noisy label

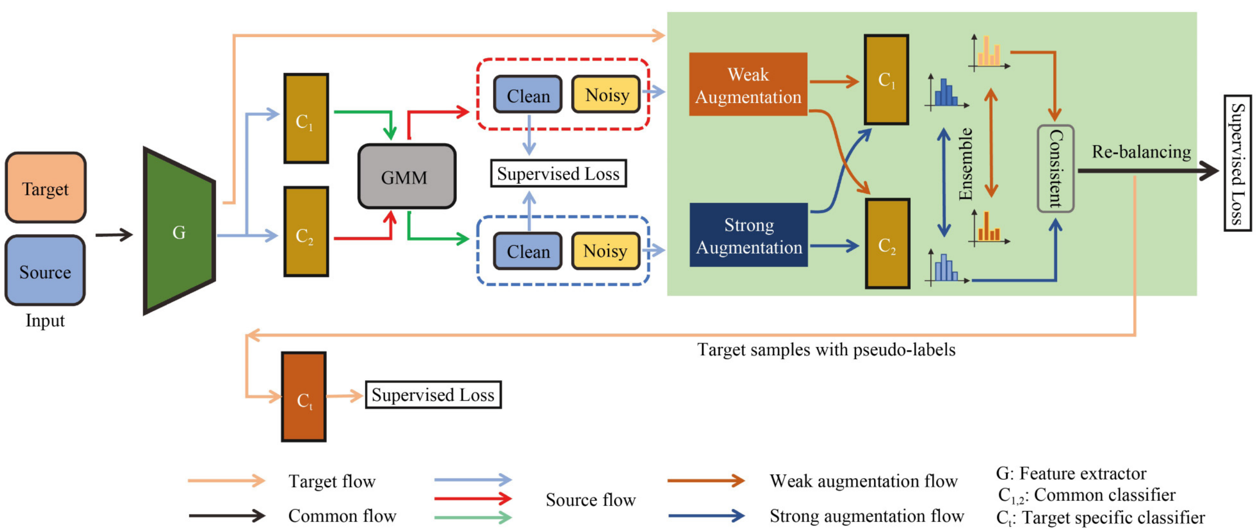


Fig. 2 Flowchart of the proposed RSLR approach. G is a shared deep feature extractor, C_1 and C_2 are dual Labeling Networks trained to correct the noisy labels and predict target sample pseudo-labels in an ensemble way. C_t is the Target-specific Network (TNet) trained on pseudo-labeled target samples and acts as the final desired target classifier

learning without domain shift have promisingly shown the validity of 2 components GMM on real-world noisy label data [33].

Refining label of noisy & target samples The LNets C_1 and C_2 predict pseudo-labels for the noisy source samples D_s^{n1} and D_s^{n2} and keep the highly reliable pseudo-labels, i.e., two subsets of pseudo-labeled samples $D_s^{sp1} \subset D_s^{n1}, D_s^{sp2} \subset D_s^{n2}$ are identified. This is reasonable as the learning proceeds, the LNets C_1 and C_2 are more and more powerful to give label predictions for some noisy samples.

Figure 3 shows the schema of the pseudo-labeling strategy, which adopts data augmentation and ensemble. For the noisy sample $x \in D_s^{n1} \cup D_s^{n2}$, the two LNets C_1, C_2 predict labels for its *weakly*-augmented version $\alpha(x)$ (e.g., flip-and-shift data augmentation) and *strongly*-augmented version $\mathcal{A}(x)$ (e.g., RandAugment [38]). We view the two Labeling Networks as weak classifiers and only retain the pseudo-labels that are consistent among the ensemble of the predictions. The unlabeled target samples are operated in the same manner and produce pseudo-labeled sample D_t^{sp}.

Learning objective function After achieving the clean and confident pseudo-labeled noisy samples for source data and confident pseudo-labeled samples for target data, we further construct three subsets of balanced samples $D_s^{bsp1}, D_s^{bsp2}, D_t^{bsp}$ from the pseudo-labeled samples $D_s^{sp1}, D_s^{sp2}, D_t^{sp}$ to deal with the LDS issue between the source and target domain, according to a sample selection percent threshold γ. We align the label distributions by keeping and learning from class-balanced pseudo-labeled samples for the source and target domain. This idea is implemented by sampling an equal size of $(\gamma/K) * N_c$ pseudo-labeled samples for each class from the whole set of N_c confident pseudo-labeled samples:

$$\gamma_t = \begin{cases} \gamma/K, & \text{if } t = 1, \\ min\{0.8, (\gamma_{t-1} + 0.05)/K\}, & \text{otherwise.} \end{cases} \tag{2}$$

Here K is the number of classes, and γ is a scaling factor to control the number of selected samples. We limit the percent of the number of pseudo-labels in each epoch by setting an equal upper percent γ/K in every class. γ is set small at the early learning stage because the pseudo-labels are less reliable. As learning proceeds, the pseudo-labels become more reliable, and we can trust more of them to be involved in model updates by increasing γ. We gradually increase the value of threshold by using $\gamma_t = \gamma_{t-1} + 0.05$ in epoch t, and set the maximum value of γ as 0.8.

After the above rebalancing operation, we train the LNets C_1, C_2 alternately using regular cross entropy loss L_c over the clean source samples D_s^c, together with a new consistent loss L_{sp} over the balanced pseudo-labeled source samples and target samples $\{D_s^{bsp}, D_t^{bsp}\}$. For every mini-batch instances $(x_b, \hat{y}_b) \in \{D_s^{bsp}, D_t^{bsp}\}$, the consistent loss is defined as following:

$$L_{sp} = \frac{1}{B} \sum_{b=1}^{B} H(\hat{y}_b, C(y \mid G(\mathcal{A}(x_b)))). \tag{3}$$

Here L_{sp} enforces the model's prediction on the *strongly*-augmented versions of samples $\mathcal{A}(x_b) = \{Aug_i(x) \mid 1 \leqslant i \leqslant \mathrm{b}\}$ to be consistent with their pseudo-labels $\hat{y}_b$. H denotes the cross entropy loss. $Aug()$ denotes *strongly*-augmented version of samples. G, C denotes the feature extractor and the LNet respectively, and y is the prediction of C on the *strongly*-augmented version of x_b.

After the LNets C_1, C_2 are trained, the TNet C_t is trained on the balanced pseudo-labeled target samples $\{D_t^{bsp}\}$ using regular cross entropy loss. When the whole training is fished, the TNet classifier C_t is used to predict labels for samples in the target domain. Algorithm 1 shows the pipeline of the whole learning procedure.

We emphasize that while we use DivideMix [33] style heuristics including data augmentations, multiple networks, and GMM noise detection, we deal with noisy samples differently from DivideMix. We conduct weak and strong augmentations and predict pseudo-labels for the detected noisy samples, and select the most reliable ones with balanced class distribution to update the learning model. Whereas DivideMix treats the detected noisy samples as unlabeled and uses them all to update the model, which more likely introduces unlabeled data prediction error and unwanted data distribution change. Furthermore, to our best knowledge, this is the first work that simultaneously deals with data covariate shift, label distribution shift, and high label noise.

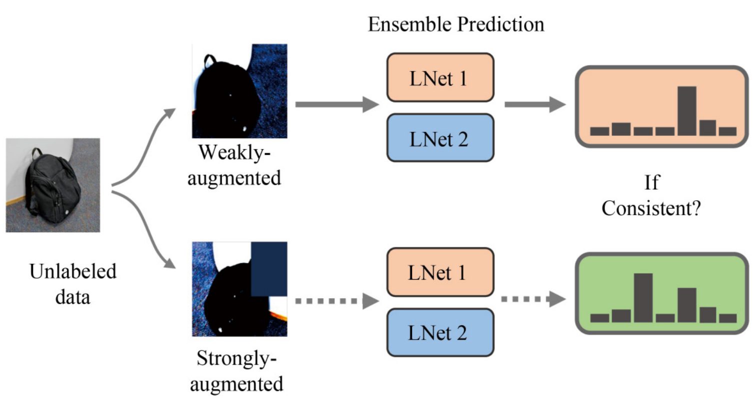


Fig. 3 The pseudo-labeling strategy for noisy source samples and unlabeled target samples. The two LNets C_1, C_2 predict labels for *weakly*-augmented version and *strongly*-augmented version of unlabeled data, then only retain the pseudo-labels that are consistent among the ensemble of the predictions

Algorithm 1 Robust domain adaptation with noisy and shifted label distribution

Input: D_s, D_t
Output: θ_G and θ_{C_t}
Initialize $G, C_1, C_2, C_t, \gamma_0$
Warm-up G, C_1, C_2 by training on D_s
while $t < T_{max}$ **do**
 // Distinguishing clean and noisy samples
 $\mathcal{W} = \mathrm{GMM}(D_s, \theta_G, \theta_C)$
 Acquire D_s^c and D_s^n according to $w_i \in \mathcal{W}$
 // Refining label of noisy and target samples
 $D_s^{sp}, D_t^{sp} = \mathrm{Labeling}(D_s^n, D_t)$
 // Re-balancing pseudo-labeled samples
 $D_s^{bsp}, D_t^{bsp} = \text{Re-balancing}(D_s^{sp}, D_t^{sp}, \gamma_t)$
 // Training C_j **for** $j = 1, 2$ **do**
  for $iter < N_{max}$ **do**
   Fetch mini-batch $D_s^{c,B}$, $D_s^{bsp,B}$, $D_t^{bsp,B}$
   Compute cross entropy loss on $D_s^{c,B}$ using Eq . (1)
   Compute consistent loss on $(D_s^{bsp,B}, D_t^{bsp,B})$ using Eq. (3).
   Update G, C_j
  end
 end
 // Training C_t
 for $iter < N_{max}$ **do**
  Fetch mini-batch $D_t^{bsp,B}$
  Compute cross entropy loss on $D_t^{bsp,B}$
  Update G, C_t
 end
 Update $\gamma_{t+1} = \min\{\gamma_0 + \gamma_t{*}0.05, 0.8\}$
end

4 Experiments

4.1 Settings

Datasets We evaluate our method on four benchmark vision datasets: Digits datasets [16] include 1) MNIST (M), 2) SYDN (S). They include images of numbers from 0 to 9. MNIST is a handwritten digits dataset that contains 60,000 train data and 10,000 test data. SYN-DIGITS consists of 479,400 train data and 9,553 test data generated from Windows fonts through changing position, background, and the amount of blur, etc. 3) **Office-31** is a standard dataset for visual domain adaptation, which involves 31 objective categories in 3 unique domain: Amazon (A), with images gathered from amazon.com, Webcam (W) and DSLR (D), with images taken by web camera and digital SLR camera respectively, for a total of 4,652 images. 4) **Office-Home** is a more challenging benchmark dataset, containing 15,500 images from 65 categories in 4 domains: Art (Ar), Clipart (Cl), Real-world (Rw), and Product (Pr). Figures 4 and 5 show the label distribution shift issue of Office-31 and Office-Home, each with different imbalance ratios over different domains.

Constructing noisy tasks Since the original datasets are clean, we corrupt the label to another class in the source dataset through a noise transition matrix T with symmetry flipping [39] and pair flipping [31] noise patterns. 1) **Digits**: following Butterfly [19], we construct 2 UDA tasks MNIST → SYDN (M → S) and SYDN → MNIST (S → M). For each task, we have four noise scenarios: P0.20, P0.45, S0.20, and S0.45, with S and P meaning Symmetry and Pair flipping noise respectively, 0.20 the proportion of noise labels. 2) **Office-31**: we construct 6 UDA tasks between the 3 domains. For each task, we simulate one low-level symmetry noise 0.4 following [21], with four additional high-level noise ratios 0.5, 0.6, 0.7, 0.8. 3) **Office-Home**: we construct 9 UDA tasks. For each task, we simulate one mixed noise feature and label noise ratio 0.4 (with average Gaussian feature noise and symmetry label noise ratio 0.2) following [21], with three additional pure symmetry label noise ratio 0.2, 0.4, 0.6.

Comparison methods 1) **Digits**: referring [19], we compare with the following baselines: Deep Adaptation Network (**DAN**) [40], Domain Adversarial Neural Network (**DANN**) [9], Asymmetric Tri-training for unsupervised Domain Adaptation (**ATDA**) [16], Transferable Curriculum for weakly-supervised domain adaptation (**TCL**) [20], **Butterfly** [19]. 2) **Office-31** and **Office-Home**: referring [21], we set the following baselines: **MentorNet** [41], **DAN** [40], Residual Transfer Network (**RTN**) [42], **DANN** [9], Adversarial Discriminative Domain Adaptation (**ADDA**) [43], Margin Disparity Discrepancy based algorithm (**MDD**) [44], **TCL** [20], **GearNet** [22], Robust Domain Adaptation under noisy environments (**RDA**) [21]. Note that **RDA** is specifically designed to handle both feature and label noise. On subtasks for which only label noise is introduced, we remove the functional part for feature noise of RDA in the experiments.

Among the baselines, **TCL**, **Butterfly**, **GearNet**, **RDA** are designed for robust UDA with noisy labels. Up to our knowledge, **TCL**, **Butterfly**, **GearNet**, and **RDA** were the best implementations to augment UDA networks for noisy domain adaptation. **MentorNet** is a normal noisy label learning method treating the two domains the same, and the others are representative standard UDA methods. For baselines, we adopt the recommended network and parameter setting in the literature paper/code. For RSLR, on Digits tasks, we employ the CNN structure [9] as the feature extractor, on Office-31 and Office-Home, we use the ResNet-50 pre-trained on ImageNet as a feature extractor. The classifiers for LNets and TNet on all datasets use 3-layer neural networks. We train

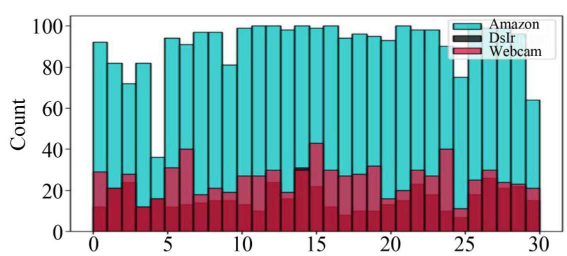


Fig. 4 The label distribution shift issue on Office-31

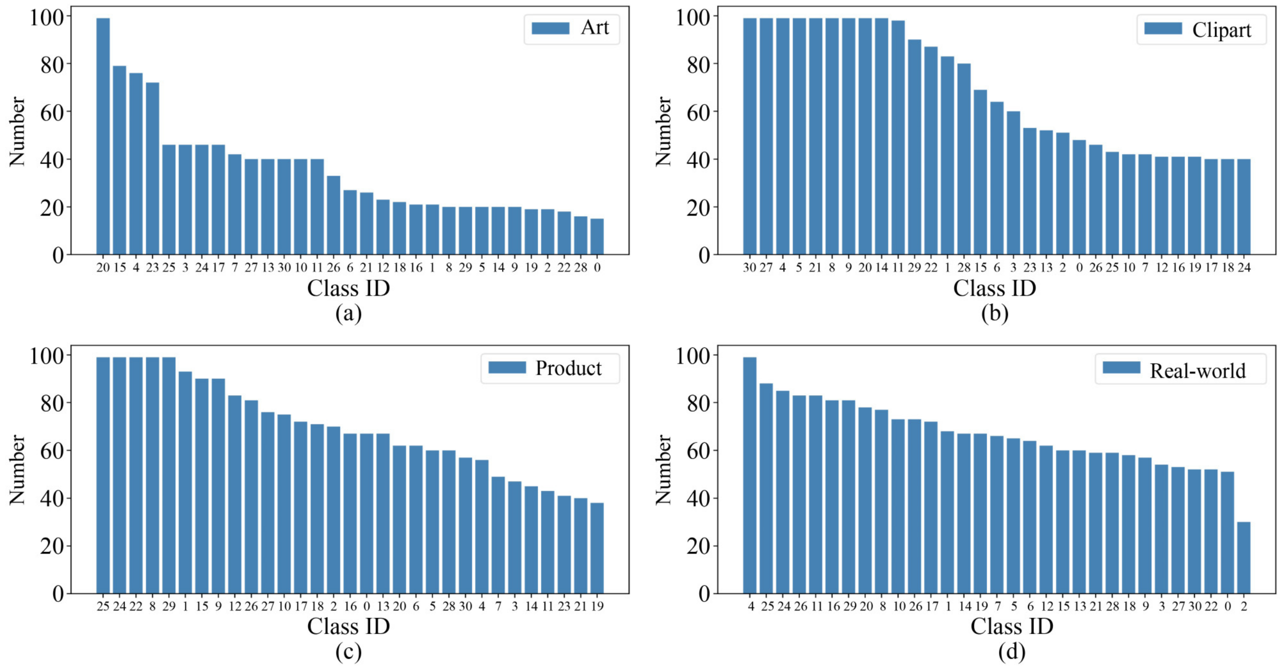


Fig. 5 The label distribution shift issue on Office-Home. (a) Art; (b) Clipart; (c) Product; (d) Real-world

the model using mini-batch SGD with a momentum of 0.9, a weight decay of 0.0004, and a batch size of 32 (128 in Digits tasks). We set the learning rate of G and C respectively as 0.01 and 0.1. The initial candidate factor of pseudo-labels samples γ_0 is set as 0.4 on Office-31/Office-Home and 0.1 on Digits. Before starting training, we warm up the model on noisy data with E_{warm} epochs according to the noise rate ratio. We set the clean probability threshold $\tau = 0.6$. We follow FixMatch [43] for data augmentation and set the number of augmentations as 2.

4.2 Results

Tables 3–6 respectively report the target domain accuracy on the three noisy UDA benchmarks. As we can see, the proposed RSLR consistently outperforms the baselines on most subtasks under various noise scenarios, except for a few cases, e.g., $\mathrm{A} \rightarrow \mathrm{W}, \mathrm{A} \rightarrow \mathrm{D}$ with label noise ratio 0.4 on Office-31 and $\mathrm{Pr} \rightarrow \mathrm{Rw}, \mathrm{Rw} \rightarrow \mathrm{Pr}$ with mixed noise ratio 0.4 on Office-Home, in which RDA performs slightly superior.

From Table 3 for **Digits**, for the relatively easy transfer task $S \rightarrow M$ (complicated domain to simple domain), in low-level symmetry noise case S0.20, the standard UDA method DAN, ATDA still perform well. Whereas in pair flipping noise cases,

Table 3 Target domain accuracy (%) on MNIST↔SYDN (8 tasks). Bold value represents the best method

Tasks	Noise type	DAN	DANN	ATDA	TCL	Butterfly	RSLR
S→M	S0.20	90.7	75.2	89.9	80.2	95.9	**98.4**
	S0.45	89.3	65.9	87.5	68.5	95.0	**98.5**
	P0.20	90.2	79.1	56.0	80.8	95.3	**98.4**
	P0.45	67.0	55.3	53.7	56.0	90.2	**97.7**
M→S	S0.20	30.6	53.5	49.8	56.7	57.1	**89.4**
	S0.45	28.2	43.8	17.2	49.9	56.2	**88.7**
	P0.20	40.8	58.8	33.7	58.9	60.4	**93.9**
	P0.45	28.4	43.7	19.5	45.3	56.6	**80.3**
Average		58.2	58.0	50.9	62.1	75.8	**93.1**

i.e., P0.20 and P0.45, all standard UDA methods fail to achieve successful knowledge transfer, except for DAN in P0.20. But the robust UDA approach Butterfly is more robust, which obtains high accuracy even in severe noisy cases P0.45, S0.45. However, it seems that all previous works degenerate severely when transferring the simple domain to the complicated domain $M \rightarrow S$ with source label noise. In contrast, RSLR performs stably well, which reveals the importance of label correction on noisy samples.

In Table 4, we report results for **Office-31** with one low

Table 4 Target domain accuracy (%) on Office-31. Bold value represents the best method

Method	Label corruption: 0.4						
	A→W	W→A	A→D	D→A	W→D	D→W	Avg
MentorNet	74.4	54.2	75.0	43.2	85.9	70.6	67.2
DAN	63.2	39.0	58.0	36.7	71.6	61.6	55.0
RTN	64.6	56.2	76.1	49.0	82.7	71.7	66.7
DANN	61.2	46.2	57.4	42.4	74.5	62.0	57.3
ADDA	61.5	49.2	61.2	45.5	74.7	65.1	59.5
MDD	74.7	55.1	76.7	54.3	89.2	81.6	71.9
TCL	82.0	65.7	83.3	60.5	90.8	77.2	76.6
GearNet	76.7	58.2	82.5	59.1	91.3	85.3	76.7
RDA	89.7	67.2	**92.0**	65.5	96.0	92.7	83.6
RSLR	**90.9**	**71.6**	86.9	**69.7**	**98.6**	**95.8**	**85.6**
	Label corruption: 0.5						
RDA	61.4	51.4	71.9	50.3	90.0	82.9	68.0
RSLR	**85.7**	**67.5**	**82.3**	**54.1**	**98.1**	**87.4**	**79.2**
	Label corruption: 0.6						
RDA	58.6	45.2	63.1	47.5	80.1	67.4	60.3
RSLR	**74.6**	**61.5**	**81.3**	**55.8**	**84.4**	**72.4**	**71.6**
	Label corruption: 0.7						
RDA	51.8	36.1	60.4	27.7	72.1	42.3	48.4
RSLR	**67.5**	**54.5**	**66.3**	**44.0**	**76.3**	**62.6**	**61.8**
	Label corruption: 0.8						
RDA	45.4	19.4	47.2	0.1	31.7	0.1	24.0
RSLR	**55.5**	**46.3**	**61.5**	**32.3**	**61.0**	**43.4**	**50.0**

Table 5 Target domain accuracy (%) on Office-Home. Bold value represents the best method

Methods	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Pr	Cl→Rw	Cl→Ar	Pr→Rw	Pr→Cl	Pr→Ar	Rw→Cl	Rw→Pr	Rw→Ar	Avg
	Label corruption: 0.2												
MentorNet	33.29	47.89	61.92	44.91	50.08	39.93	65.16	31.75	43.88	36.88	68.48	56.45	48.39
DAN	40.78	58.93	68.05	57.45	60.20	49.16	68.69	37.37	51.42	44.49	71.84	60.82	55.77
RTN	32.35	51.84	59.35	48.52	52.47	40.05	62.27	30.84	42.97	36.31	66.05	53.93	48.08
DANN	36.29	52.92	31.34	50.01	53.39	40.96	65.99	32.07	44.17	39.73	71.01	55.71	47.80
ADDA	28.36	42.26	52.03	41.36	43.29	30.61	57.61	28.50	31.40	31.98	63.17	49.98	41.71
MDD	35.10	56.95	64.91	54.27	55.86	43.02	65.46	31.55	46.64	34.85	70.74	56.24	51.30
TCL	29.07	43.84	56.87	42.37	45.26	37.25	64.08	33.22	43.79	37.37	71.05	60.24	47.03
GearNet	37.78	59.62	69.99	61.68	61.58	49.08	73.21	37.82	51.71	43.36	75.59	62.04	56.96
RDA	54.04	70.17	75.79	65.06	67.94	**61.10**	75.63	51.89	**61.15**	57.80	80.27	**68.85**	65.81
RSLR	**58.46**	**72.60**	**76.68**	**74.96**	**74.79**	58.96	**78.45**	**56.20**	60.13	**61.81**	**82.12**	66.01	**68.43**
	Label corruption: 0.4												
MentorNet	23.89	37.10	50.40	33.30	37.40	36.71	55.30	28.40	40.17	33.60	63.90	53.03	41.10
DAN	30.20	48.70	61.00	52.40	56.70	48.58	64.90	34.10	46.64	40.10	68.20	55.62	50.60
RTN	19.40	38.70	46.70	45.00	47.10	36.55	55.40	24.20	33.13	32.20	57.70	46.15	40.19
DANN	27.70	44.20	57.20	42.30	44.10	38.81	60.40	27.80	37.82	34.70	67.10	52.58	44.56
ADDA	16.30	26.10	38.80	31.60	33.50	30.82	35.10	16.90	29.21	20.40	41.80	41.99	30.21
MDD	34.40	54.30	60.80	53.90	53.80	43.35	62.80	28.60	40.71	37.40	68.40	50.60	49.09
TCL	24.20	37.40	44.50	30.40	33.30	32.55	52.40	25.90	34.61	33.60	60.30	50.23	38.28
GearNet	32.10	51.68	62.52	50.63	54.55	44.17	67.26	33.09	47.58	40.40	72.31	56.79	51.09
RDA	31.71	50.92	63.32	52.50	54.70	57.89	66.20	34.01	**56.90**	42.00	68.60	**67.78**	53.88
RSLR	**50.91**	**60.72**	**66.52**	**68.20**	**70.40**	**57.95**	**77.90**	**51.40**	56.29	**60.50**	**80.56**	63.59	**63.75**
	Label corruption: 0.6												
MentorNet	19.84	30.46	38.99	32.28	36.01	28.51	49.53	24.99	33.79	29.26	53.01	45.90	35.21
DAN	23.05	35.41	42.67	43.16	50.65	37.25	59.79	28.98	38.32	33.93	61.41	45.86	41.71
RTN	13.72	24.85	32.84	32.85	35.16	22.13	46.96	23.28	27.48	26.23	47.02	37.12	30.80
DANN	20.53	31.34	38.97	35.50	40.74	28.22	52.03	27.42	32.67	30.13	55.58	42.89	36.34
ADDA	15.10	22.89	26.44	29.38	31.03	19.61	44.89	20.78	26.16	24.97	43.75	36.84	28.49
MDD	19.73	31.72	37.37	43.30	42.94	25.96	53.87	24.95	33.99	28.98	58.64	41.24	36.89
TCL	15.66	29.22	38.56	27.82	31.95	26.97	47.19	23.07	32.10	29.16	50.62	43.22	32.96
GearNet	25.99	44.63	54.21	44.20	47.45	39.58	60.09	28.61	40.36	34.83	63.93	53.88	44.81
RDA	29.39	45.03	52.26	43.03	50.01	36.92	61.28	32.85	41.08	38.74	67.04	55.13	46.06
RSLR	**36.42**	**50.71**	**59.04**	**63.04**	**64.27**	**49.21**	**72.05**	**49.91**	**52.46**	**55.42**	**75.52**	**57.16**	**57.10**

Table 6 Target domain accuracy (%) on Office-Home with 40% mixed noise. The bold value represents the best method

Methods	Mixed corruption: 0.4									
	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Pr	Cl→Rw	Pr→Rw	Pr→Cl	Rw→Cl	Rw→Pr	Avg
MentorNet	34.5	57.1	66.7	56.1	57.6	70.2	34.0	37.2	70.4	53.8
DAN	31.2	52.3	61.2	53.1	54.6	61.5	30.3	36.7	67.4	49.8
RTN	29.3	57.8	66.3	58.6	58.3	67.5	30.1	32.2	69.9	52.2
DANN	32.9	50.6	60.1	49.2	50.6	60.4	32.6	38.4	67.4	49.1
ADDA	32.6	52.0	60.6	53.5	54.3	63.1	31.6	37.7	67.5	50.3
MDD	44.6	62.4	68.8	58.9	60.8	65.2	39.5	47.1	72.9	57.8
TCL	38.8	62.1	69.4	58.5	59.8	72.3	39.5	43.5	74.0	57.5
GearNet	38.6	58.5	68.3	59.2	60.8	71.6	37.5	44.3	75.1	57.1
RDA	50.8	68.7	72.3	67.4	67.9	**74.6**	50.5	57.7	**80.2**	65.6
RSLR	**52.6**	**69.5**	**73.1**	**68.7**	**70.2**	73.1	**51.2**	**58.3**	77.0	**66.0**

noise ratio 0.4 and four high level noise cases $\{0.5, 0.6, 0.7, 0.8\}$. For the high-level cases, we only report comparison results for the best baseline RDA. Our method outperforms significantly in almost all cases, whereas RDA gets ruined when facing 0.8 noise, e.g., on $D \to A, D \to W$. This phenomenon can be explainable because RDA learns only on the subset of clean labeled samples, which however in high-level noise cases are too limited.

In Table 5, we show results for **Office-Home** with low (0.2), medium (0.4), and high (0.6) label corruptions. For the low-level cases, our method gets better performance on most tasks except Cl $\to$ Ar, Pr $\to$ Ar, and Rw $\to$ Ar. Meanwhile, our method achieves the best average performance across all 12 tasks. Similar results are observed fo 0.4 label corruption. It seems that when the noise corruption ratio rises to 0.6, our strength becomes more prominent. We can achieve 2%−20% better performance on all tasks. The above results fully demonstrate the effectiveness of our methods not only in low-level corruption but also at high- levels.

We observe an interesting phenomenon that even in the mixed noise case in Table 6 with features corrupted by Gaussian blur and salt-and-pepper noise, our method is still robust, outperforming RDA most time. RDA performs inferior to our approach not because it's weak on noisy label detection, but because it uses samples differently. Proper using of noisy samples includes not only correcting their labels but also in a distribution-aware way. This is not straightforward.

4.3 Further in-depth analysis

Feature space analysis In Fig. 6, we show the t-SNE embedding [44] of the target domain on Office-31 A → W in low noise ratio: 0.4 (a, b, c, d) and high noise ratio: 0.6 (e, f, g, h) learned by DANN [9], TCL [20], RDA [21] and RSLR. The features learned by our method are more discriminative in every class compared with DANN, TCL, and RDA, which verifies that our method can learn better features under different noise levels. Our method learns tighter representation clusters across all categories not only at low noise levels but also at high noise levels. Other methods just learn good representations of a few categories and fail to discriminate other categories in high noise levels.

Pseudo-labeling strategy In Fig. 7, we show comparisons for the accuracy of selected pseudo-labels between our method and the confidence strategy [43] under 40% label corruption. The strategy in our method could keep a stable and high accuracy of pseudo-labels as training proceeds, e.g., A → W, A → D. However, the accuracy of other strategies easily drops. This indicates that it is unreliable to select pseudo-labels just through confidence when faced with domain shift.

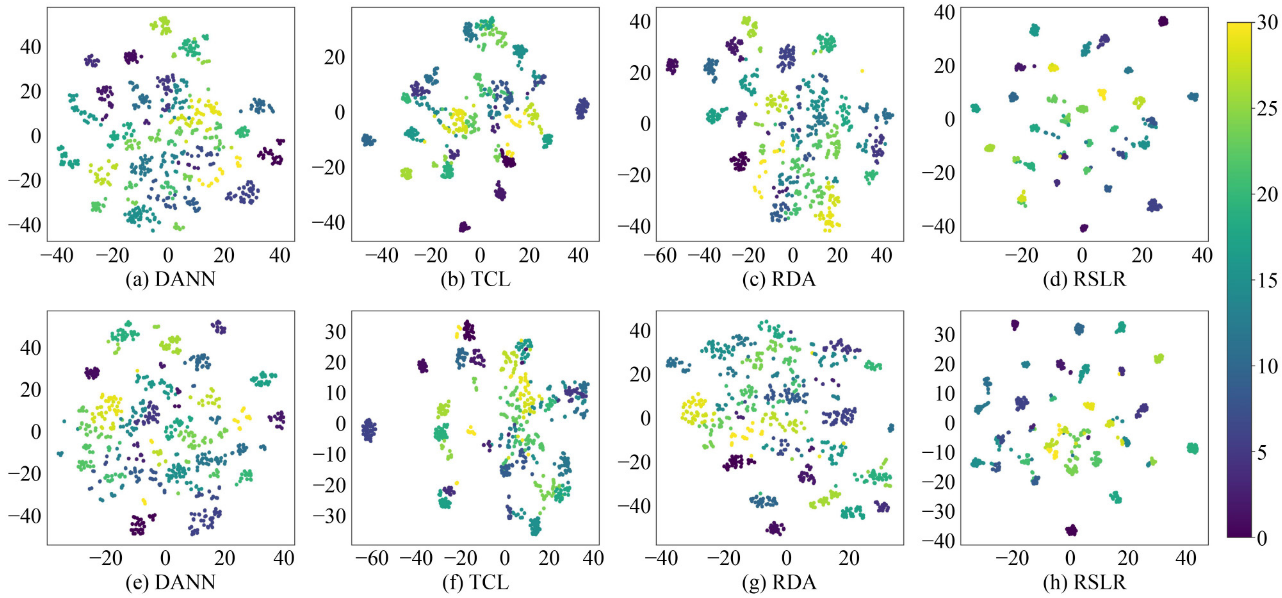


Fig. 6 Target domain t-SNE visualization of class features on Office-31 A→W tasks in low noise corruption: 0.4 (Top) and high noise corruption: 0.6 (Bottom). Office-31 includes 31 classes with different colors

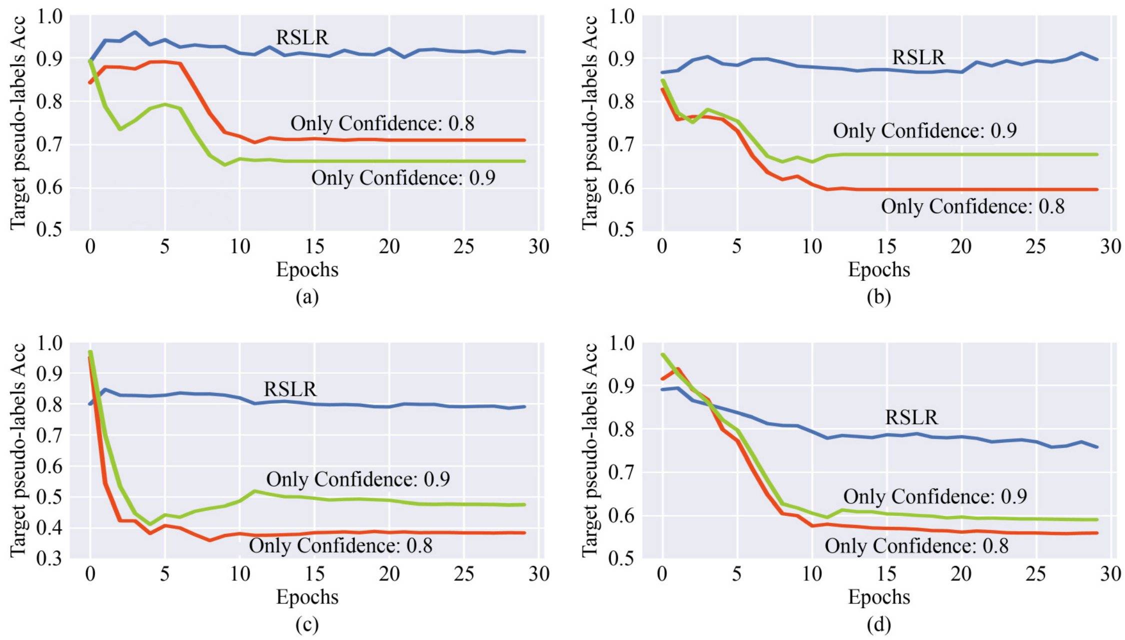


Fig. 7 Comparison of the accuracy of target pseudo-labeled samples during training on Office-31 under 40 % label corruption. (a) A→W; (b) A→D; (c) D→A; (d) W→A

Table 7 Ablation study of our method on 6 sub-tasks of Office-31 under 40% label corruption

Methods			Label corruption: 0.4						
Double LNets	Rebalance	Recycling	A→W	W→A	A→D	D→A	D→W	W→D	Avg
		✓	84.4	69.7	85.4	63.2	92.4	96.5	81.9
✓			83.1	68.6	**87.3**	55.2	88.4	96.3	79.8
✓		✓	89.9	70.6	84.9	67.7	97.6	94.8	84.3
✓	✓	✓	**90.9**	**71.6**	86.9	**69.7**	**95.8**	**98.6**	**85.6**

Ablation study Table 7 shows the ablation study results of the proposed RSLR approach on **Office-31** with label noise ratio 0.4. We explore the effect of removing different components to explain what makes RSLR successful. To verify the effectiveness of two LNets for better labeling, we study using one single LNet to refine the noisy data and pseudo-label the unlabeled data, i.e., **w/o double LNets**; to study the effect of skewed label distribution when no class distribution rebalancing is conducted, we remove the rebalancing strategy of pseudo-labeling, i.e., **w/o rebalance**; most importantly, we fully present the significance of recycling the noisy samples information for knowledge transfer, by only training LNets on distinguished clean source samples and pseudo-labeled target samples, i.e., **w/o recycling**.

From Table 7, it can be seen that each component plays a critical role in the success of RSLR, especially the distribution rebalancing strategy to deal with the label distribution shift issue between domains, and the noisy sample recycling which however is completely ignored by previous work.

5 Conclusion

Existing methods on robust UDA with noisy labels mainly combat the label noise and domain gap by learning on the distinguished clean samples and matching the domain covariate distributions but have ignored recycling the low-quality noisy samples and the label distribution shift issue between domains. In this paper, we propose a novel improved approach named RSLR, which makes full recycling of the noisy samples by refining and assigning reliable pseudo-labels for them. To cope with the label distribution shift, RSLR also incorporates a class distribution rebalancing strategy for the source and target data. Extensive experiments have been conducted to validate the effectiveness of RSLR.

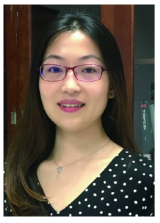
Acknowledgements This work was supported by the National Key R&D Program of China (2022ZD0114801), the National Natural Science Foundation of China (Grant No. 61906089), and the Jiangsu Province Basic Research Program (BK20190408).

Competing interests The authors declare that they have no competing interests or financial conflicts to disclose.

References

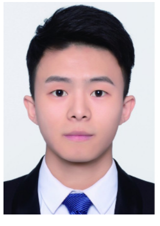
1. Wang M, Deng W. Deep visual domain adaptation: a survey. Neurocomputing, 2018, 312: 135–153
2. Csurka G. A comprehensive survey on domain adaptation for visual applications. In: Csurka G, ed. Domain Adaptation in Computer Vision Applications. Cham: Springer, 2017, 1–35
3. Perone C S, Ballester P, Barros R C, Cohen-Adad J. Unsupervised domain adaptation for medical imaging segmentation with self-ensembling. NeuroImage, 2019, 194: 1–11
4. Ojha R, Sekhar C C. Unsupervised domain adaptation in speech recognition using phonetic features. 2021, arXiv preprint arXiv: 2108.02850
5. Ben-David S, Blitzer J, Crammer K, Pereira F. Analysis of representations for domain adaptation. In: Schölkopf B, Platt J, Hofmann T, eds. Advances in Neural Information Processing Systems 19: Proceedings of 2006 Conference. Cambridge: MIT Press, 2007, 137−144
6. Mansour Y, Mohri M, Rostamizadeh A. Domain adaptation: learning bounds and algorithms. 2009, arXiv preprint arXiv: 0902.3430
7. Ghifary M, Kleijn W B, Zhang M. Domain adaptive neural networks for object recognition. In: Proceedings of the 13th Pacific Rim International Conference on Artificial Intelligence. 2014, 898−904
8. Yan H, Ding Y, Li P, Wang Q, Xu Y, Zuo W. Mind the class weight bias: weighted maximum mean discrepancy for unsupervised domain adaptation. In: Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition. 2017, 945−954
9. Ganin Y, Lempitsky V. Unsupervised domain adaptation by backpropagation. In: Proceedings of the 32nd International Conference on Machine Learning. 2015, 1180−1189
10. Bousmalis K, Trigeorgis G, Silberman N, Krishnan D, Erhan D. Domain separation networks. In: Proceedings of the 30th International Conference on Neural Information Processing Systems. 2016, 343−351
11. Long M, Cao Y, Wang J, Jordan M I. Learning transferable features with deep adaptation networks. In: Proceedings of the 32nd International Conference on Machine Learning. 2015, 97−105
12. Taigman Y, Polyak A, Wolf L. Unsupervised cross-domain image generation. In: Proceedings of the International Conference on Learning Representations. 2017
13. Hoffman J, Tzeng E, Park T, Zhu J Y, Isola P, Saenko K, Efros A, Darrell T. CyCADA: Cycle-consistent adversarial domain adaptation. In: Proceedings of the 35th International Conference on Machine Learning. 2018, 1989−1998
14. Bousmalis K, Silberman N, Dohan D, Erhan D, Krishnan D. Unsupervised pixel-level domain adaptation with generative adversarial networks. In: Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition. 2017, 95−104
15. Li S, Xie M, Gong K, Liu C H, Wang Y, Li W. Transferable semantic augmentation for domain adaptation. In: Proceedings of 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021, 11511−11520
16. Saito K, Ushiku Y, Harada T. Asymmetric tri-training for unsupervised domain adaptation. In: Proceedings of the 34th International Conference on Machine Learning. 2017, 2988−2997
17. Prabhu V, Khare S, Kartik D, Hoffman J. SENTRY: selective entropy optimization via committee consistency for unsupervised domain adaptation. In: Proceedings of 2021 IEEE/CVF International Conference on Computer Vision. 2021, 8538−8547
18. Sheng V S, Zhang J. Machine learning with crowdsourcing: a brief summary of the past research and future directions. In: Proceedings of the 33rd AAAI Conference on Artificial Intelligence. 2019, 9837−9843
19. Liu F, Lu J, Han B, Niu G, Zhang G, Sugiyama M. Butterfly: a panacea for all difficulties in wildly unsupervised domain adaptation. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems Workshop. 2019

20. Shu Y, Cao Z, Long M, Wang J. Transferable curriculum for weakly-supervised domain adaptation. In: Proceedings of the 33rd AAAI Conference on Artificial Intelligence. 2019, 4951−4958
21. Han Z, Gui X J, Cui C, Yin Y. Towards accurate and robust domain adaptation under noisy environments. In: Proceedings of the 29th International Joint Conference on Artificial Intelligence. 2020, 2269−2276
22. Xie R, Wei H, Feng L, An B. GearNet: stepwise dual learning for weakly supervised domain adaptation. In: Proceedings of the 36th AAAI Conference on Artificial Intelligence. 2022, 8717−8725
23. Gretton A, Borgwardt K M, Rasch M J, Schölkopf B, Smola A. A kernel two-sample test. The Journal of Machine Learning Research, 2012, 13: 723–773
24. Lee J, Raginsky M. Minimax statistical learning with wasserstein distances. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems. 2018, 2692−2701
25. Long M, Zhu H, Wang J, Jordan M I. Deep transfer learning with joint adaptation networks. In: Proceedings of the 34th International Conference on Machine Learning. 2017, 2208−2217
26. Ganin Y, Ustinova E, Ajakan H, Germain P, Larochelle H, Laviolette F, Marchand M, Lempitsky V. Domain-adversarial training of neural networks. The Journal of Machine Learning Research, 2016, 17(1): 2096–2030
27. Tzeng E, Hoffman J, Saenko K, Darrell T. Adversarial discriminative domain adaptation. In: Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition. 2017, 2962−2971
28. Kundu J N, Kulkarni A R, Bhambri S, Mehta D, Kulkarni S A, Jampani V, Radhakrishnan V B. Balancing discriminability and transferability for source-free domain adaptation. In: Proceedings of the 39th International Conference on Machine Learning. 2022, 11710−11728
29. Liu H, Wang J, Long M. Cycle self-training for domain adaptation. In: Proceedings of the 35th International Conference on Neural Information Processing Systems. 2021, 22968−22981
30. Arpit D, Jastrzębski S, Ballas N, Krueger D, Bengio E, Kanwal M S, Maharaj T, Fischer A, Courville A, Bengio Y, Lacoste-Julien S. A closer look at memorization in deep networks. In: Proceedings of the 34th International Conference on Machine Learning. 2017, 233−242
31. Han B, Yao Q, Yu X, Niu G, Xu M, Hu W, Tsang I W, Sugiyama M. Co-teaching: robust training of deep neural networks with extremely noisy labels. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems. 2018, 8536−8546
32. Arazo E, Ortego D, Albert P, O'Connor N, McGuinness K. Unsupervised label noise modeling and loss correction. In: Proceedings of the International Conference on Machine Learning. 2019, 312−321
33. Li J, Socher R, Hoi S C H. DivideMix: learning with noisy labels as semi-supervised learning. In: Proceedings of the International Conference on Learning Representations. 2020
34. Yu X, Han B, Yao J, Niu G, Tsang I, Sugiyama M. How does disagreement help generalization against label corruption? In: Proceedings of the 36th International Conference on Machine Learning. 2019, 7164−7173
35. Yi L, Liu S, She Q, McLeod A I, Wang B. On learning contrastive representations for learning with noisy labels. In: Proceedings of 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022, 16661−16670
36. Tarvainen A, Valpola H. Mean teachers are better role models: weight-averaged consistency targets improve semi-supervised deep learning results. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. 2017, 1195−1204
37. Permuter H, Francos J, Jermyn I. A study of Gaussian mixture models of color and texture features for image classification and segmentation. Pattern Recognition, 2006, 39(4): 695–706
38. Cubuk E D, Zoph B, Shlens J, Le Q V. Randaugment: practical automated data augmentation with a reduced search space. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2020, 3008−3017
39. Patrini G, Rozza A, Krishna Menon A, Nock R, Qu L. Making deep neural networks robust to label noise: a loss correction approach. In: Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition. 2017, 2233−2241
40. Jiang L, Zhou Z, Leung T, Li L J, Fei-Fei L. MentorNet: learning data-driven curriculum for very deep neural networks on corrupted labels. In: Proceedings of the 35th International Conference on Machine Learning. 2018, 2304−2313
41. Long M, Zhu H, Wang J, Jordan M I. Unsupervised domain adaptation with residual transfer networks. In: Proceedings of the 30th International Conference on Neural Information Processing Systems. 2016, 136−144
42. Zhang Y, Liu T, Long M, Jordan M. Bridging theory and algorithm for domain adaptation. In: Proceedings of the 36th International Conference on Machine Learning. 2019, 7404−7413
43. Sohn K, Berthelot D, Li C L, Zhang Z, Carlini N, Cubuk E D, Kurakin A, Zhang H, Raffel C. FixMatch: simplifying semi-supervised learning with consistency and confidence. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. 2020, 51
44. Donahue J, Jia Y, Vinyals O, Hoffman J, Zhang N, Tzeng E, Darrell T. DeCAF: a deep convolutional activation feature for generic visual recognition. In: Proceedings of the 31st International Conference on Machine Learning. 2014, I-647−I-655

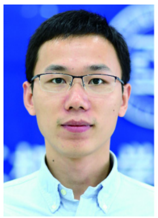


Shao-Yuan Li is an associate professor in the College of Computer Science and Technology at Nanjing University of Aeronautics and Astronautics, China. She received BSc and PhD degrees in computer science from Nanjing University, China in 2010 and 2018. Her research interests include machine learning and data mining. She has won the Champion of PAKDD'12 Data Mining Challenge, the Best Paper Award of PRICAI'18, the 2nd place of Learning and Mining with Noisy Labels Challenge at IJCAI'22, and the 4th place of Continual Learning Challenge at CVPR'23.

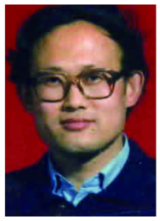
Shi-Ji Zhao received the BSc degree in computer science from Nanjing Agricultural University, China in 2022. Currently, he is working towards an MS degree in the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China. His research interests include Domain Adaptation and Test Time Adaptation.



Zheng-Tao Cao received the BSc degree in computer science in 2020 from Shandong University of Technology, and the MS degree from Nanjing University of Aeronautics and Astronautics, China in 2023. His research interests include machine learning and domain adaptation.



Sheng-Jun Huang received the BSc and PhD degrees in computer science from Nanjing University, China in 2008 and 2014, respectively. He is now a professor in the College of Computer Science and Technology at Nanjing University of Aeronautics and Astronautics, China. His main research interests include machine learning and data mining. He has been selected to the Young Elite Scientists Sponsorship Program by CAST in 2016, and won the China Computer Federation Outstanding Doctoral Dissertation Award in 2015, the KDD Best Poster Award in 2012, and the Microsoft Fellowship Award in 2011. He is a Junior Associate Editor of Frontiers of Computer Science.



Songcan Chen received the BS degree in mathematics from Hangzhou University (now merged into Zhejiang University), China in 1983, and the MS degree in computer applications from Shanghai Jiao Tong University, China in 1985, and then worked with Nanjing University of Aeronautics and Astronautics (NUAA) , China in January 1986. He received the PhD degree in communication and information systems from NUAA in 1997. Since 1998, as a full-time professor, he has been with the College of Computer Science and Technology, NUAA. His research interests include pattern recognition, machine learning, and neural computing. He is also an IAPR fellow.