

# Unsupervised Domain Adaptation of Black-Box Source Models

Haojian Zhang\*<sup>1</sup>

eehjzhang@mail.scut.edu.cn

Yabin Zhang\*<sup>2</sup>

czybzhang@comp.polyu.edu.hk

Kui Jia<sup>1</sup>

kuijia@scut.edu.cn

Lei Zhang<sup>2</sup>

cslzhang@comp.polyu.edu.hk

<sup>1</sup> School of Electronic and Information Engineering

South China University of Technology  
Guangzhou, China

<sup>2</sup> Department of Computing

The Hong Kong Polytechnic University  
HongKong, China

---

## Abstract

Unsupervised domain adaptation (UDA) aims to learn models for a target domain of unlabeled data by transferring knowledge from a labeled source domain. In the traditional UDA setting, labeled source data are assumed to be available for adaptation. Due to increasing concerns for data privacy, source-free UDA is highly appreciated as a new UDA setting, where only a trained source model is assumed to be available, while labeled source data remain private. However, trained source models may also be unavailable in practice since source models may have commercial values and exposing source models brings risks to the source domain, e.g., problems of model misuse and white-box attacks. In this work, we study a subtly different setting, named Black-Box Unsupervised Domain Adaptation (B<sup>2</sup>UDA), where only the application programming interface of source model is accessible to the target domain; in other words, the source model itself is kept as a black-box one. To tackle B<sup>2</sup>UDA, we propose a simple yet effective method, termed Iterative Learning with Noisy Labels (IterLNL). With black-box models as tools of noisy labeling, IterLNL conducts noisy labeling and learning with noisy labels (LNL) iteratively. To adapt the LNL to B<sup>2</sup>UDA, we estimate the noise rate from model predictions of unlabeled target data and propose category-wise sampling to tackle the unbalanced label noise among categories. Experiments on benchmark datasets show the efficacy of IterLNL. Given neither source data nor source models, IterLNL performs comparably with traditional UDA methods that make full use of labeled source data.

## 1 Introduction

Although deep models have achieved success on various tasks, it is difficult to generalize the model learned from labeled training data to a target domain of slightly shifted data distribution. At the same time, it is expensive to collect a new target dataset with a large number of labeled training data. Therefore, unsupervised domain adaptation (UDA) [9, 24, 30] is introduced to learn the target model by transferring knowledge from the labeled source domain

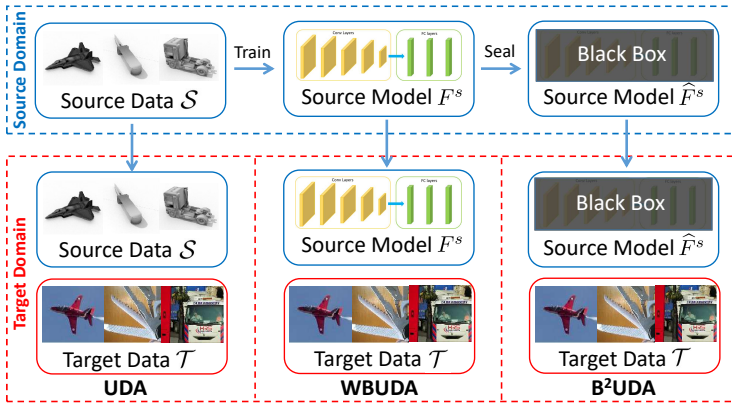


Figure 1: An illustration of different UDA settings. Source data and source model are respectively required in the traditional UDA and WBUDA settings. In contrast,  $B^2$ UDA requires a black-box access to the source model only, which is the least restrictive condition to apply domain adaptation to the unsupervised target data.

to the unlabeled target domain. Motivated by seminal theories [4, 49, 50], popular UDA methods [4, 24, 25, 36, 58, 48] target at learning domain invariant feature representations. In UDA, labeled source data are assumed to be available for target domain.

Although remarkable success has been achieved in UDA, increasing concerns for data privacy post new challenges to it. Specifically, data of source and target domains are typically captured and stored on different devices and contain private information. Thus it is risky to expose source data to the target domain and vice versa. In other words, labeled source data may be not available for the target domain, impeding the application of popular UDA methods [4, 24, 25, 36, 58, 48]. For this reason, a novel task, source-free UDA, is introduced [17, 23, 45] to facilitate the model adaptation and protect the source data privacy simultaneously.

Unlike the vanilla UDA, a well-trained white-box source model, instead of labeled source data, is provided to unlabeled target domain in the source-free UDA [23, 45]; thus, we term this task as *white-box unsupervised domain adaptation* (WBUDA) to distinguish it from our investigated one in later paragraphs. In WBUDA, the adaptation could be achieved by fine-tuning the source model on unlabeled target data with well-designed objectives [23, 45].

However, the white-box source model is not always given in practice. Most valuable models on cloud services (e.g., Google Cloud) are sealed as application programming interface (API), where only the input-output interface of a model is available and the model itself is kept as a black-box one. As stated in [29], releasing an API instead of a white-box model could commercialize the technology, reduce model misuse and make the model use conveniently for the public. Due to all reasons mentioned above, white-box source models are probably unavailable in practice, which hinders the application of WBUDA methods.

In this work, we study a subtly different setting of source-free UDA, where only the API of the source model is accessible for the target domain. In other words, the source model itself is kept as a black-box one; thus, we term this task as *black-box unsupervised domain adaptation* ( $B^2$ UDA). A few recent attempts [3, 46] have been made to tackle the  $B^2$ UDA problem, but achieving less satisfactory results. In this work, we propose a simple yet effective algorithmic framework, termed Iterative Learning with Noisy Labels (IterLNL). With

black-box models as tools of noisy labeling, IterLNL conducts noisy labeling and LNL iteratively. Specifically, we first get model predictions of target data based on the black-box model and obtain their noisy labels as the category with the maximum prediction probability. We note that the label noise via the black-box model may be highly unbalanced among categories (cf. Figure 2(c)), which is significantly different from the simulated and balanced ones in LNL [6, 40]; such unbalanced label noise hinders the application of state-of-the-art LNL methods [6, 24], inspiring the category-wise sampling strategy. To facilitate the implementation of LNL in B<sup>2</sup>UDA, we estimate the noise rate from model predictions of unlabeled target data. Experiments on benchmark datasets confirm the efficacy of our method.

## 2 Related Work

**Source Free UDA.** Traditional UDA [9, 24] assumes that labeled source data are available for the target domain. Due to increasing concerns for data privacy, source-free UDA [12, 17, 18, 22, 23, 45] is highly appreciated as a new UDA setting, where only a source model is available for the target domain while labeled source data remain private. Source free UDA methods typically fine-tune the source model for the target domain with unlabeled target data [17, 22, 23, 45]. Specifically, Liang *et al.* [23] fine-tune the source model with pseudo-labeling and information maximization between target data and their predictions; a weight constraint is adopted in [22] to encourage similarity between the source model and adapted target model. Additionally, source data and source-style target data are respectively generated in [18] and [17] using the statistics information stored in source model. The white-box source model is required in the methods above, but it may be unavailable due to the commercial and/or safety consideration [24]. To this end, we study a subtly different B<sup>2</sup>UDA setting, where only the API of source model is accessible for the target domain; in other words, the source model itself is kept as a black-box one. We note that several attempts have been made on the B<sup>2</sup>UDA problem recently. Based on pre-trained features, a denoising auto-encoder is used for prediction of target labels in [9] and an encoder-decoder framework is used in [46] where encoded target features are aligned to reference Gaussian distributions; however, both of the two methods obtain less satisfactory results on benchmark datasets. Morerio *et al.* [26] first train a conditional Generative Adversarial Network (cGAN) with unlabeled target data and their source predictions, and then learn the target model with samples generated by cGAN; its performance is conditioned on the high-quality samples generated by cGAN, thus limiting its general usage in UDA tasks. In general, the B<sup>2</sup>UDA problem is not well addressed yet. In the present work, we propose IterLNL and conduct thorough experiments on popular UDA benchmarks; results show that our proposed method works successfully for B<sup>2</sup>UDA.

**Learning with Noisy Labels (LNL).** LNL aims to learn models with noisy labeled data robustly. Seminal LNL methods include estimating noise transition matrix to transfer observed noisy labels to latent clean ones [37, 39], refining the objective function [7, 51], and avoiding overfitting noisy labels with memorization effects of neuron networks [6, 24], as summarized in [8]. Although we tackle B<sup>2</sup>UDA with a LNL-like method, it should be noted that tasks of B<sup>2</sup>UDA and LNL are fundamentally different. Specifically, a black-box source model, rather than noisy labels, is given in B<sup>2</sup>UDA to help the learning with unlabeled data; apart from introducing noisy labels as in our IterLNL, the black-box source model could be used in other ways, e.g., model distillation [41], which are left for further studies.

### 3 Problems and the proposed Method

Given unlabeled target data  $\mathcal{T} = \{\mathbf{x}_i^t\}_{i=1}^{n^t}$  sampled from a distribution  $\mathcal{Q}$ , our problem of interest is to learn a model  $F: \mathcal{X}^t \rightarrow [0, 1]^K$  such that the empirical target risk  $\frac{1}{n^t} \sum_{i=1}^{n^t} [\mathcal{L}(F(\mathbf{x}_i^t), y_i^t)]$  (or ideally, the expected risk  $\mathbb{E}_{(\mathbf{x}^t, y^t) \in \mathcal{Q}} [\mathcal{L}(F(\mathbf{x}^t), y^t)]$ ) could be minimized, where  $K$  is the category number,  $\mathcal{L}$  is the loss function of the task, and  $y_i^t \in \{1, \dots, K\}$ ,  $i = 1, \dots, n^t$ , is the target label to be estimated. Depending on how much knowledge one may have from a source domain, the problem can fall in different established realms of unsupervised learning [41], unsupervised domain adaptation (UDA) [9, 24], and source-free UDA [23, 45]. While the first one assumes no the source knowledge and is of machine learning foundations, in this work, we focus on different problem settings of UDA. We can formulate the UDA procedure as:  $F^t \leftarrow \mathcal{O}(\mathcal{T}, \mathcal{S})$ , where  $F^t$  is the expected target model,  $\mathcal{O}$  is a certain method, and  $\mathcal{S}$  means the available resource of source domain. In UDA,  $\mathcal{S}$  contains labeled source data  $\{\mathbf{x}_i^s, y_i^s\}_{i=1}^{n^s}$  sampled from a distribution  $\mathcal{P}$ ; in WBUDA,  $\mathcal{S}$  is a source model  $F^s$  with visible parameters and structures; in B<sup>2</sup>UDA,  $\mathcal{S}$  is an API of  $F^s$ , which only provides input-output interface of  $F^s$  and is denoted as  $\widehat{F}^s$ ; in other words, we could get the output of  $F^s(\mathbf{x})$  via  $\widehat{F}^s(\mathbf{x})$  while the source model  $F^s$  itself is kept as a black-box one. As there are many model APIs on cloud services, B<sup>2</sup>UDA is a promising way to improve their adaptabilities, presenting broad practical values.

Labeled source data  $\mathcal{S}$  and white-box source model  $F^s$  are respectively required in UDA and WBUDA methods, impeding their applications in the B<sup>2</sup>UDA task. To this end, we propose an IterLNL framework by conducting noisy labeling and LNL iteratively, which are introduced as follows.

#### 3.1 Noisy Labeling

Given a black-box model  $\widehat{F}$  (e.g., the black-box source model  $\widehat{F}^s$ ) in B<sup>2</sup>UDA, we could get label predictions of target data  $\{\mathbf{x}_i^t\}_{i=1}^{n^t}$  with  $\widehat{F}$  as:

$$\widehat{\mathcal{Y}} = \{\widehat{F}(\mathbf{x}_i^t)\}_{i=1}^{n^t}. \quad (1)$$

The corresponding pseudo label of target sample  $\mathbf{x}_i^t$  is defined as  $\widehat{y}_i^t = \arg \max_k \widehat{F}_k(\mathbf{x}_i^t)$ . The pseudo labels  $\{\widehat{y}_i^t\}_{i=1}^{n^t}$  could be highly noisy due to the divergence across source and

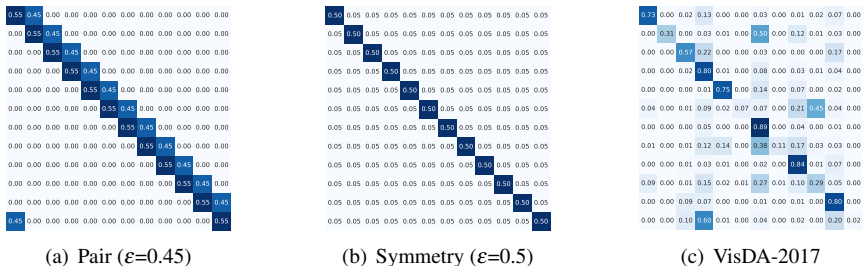


Figure 2: (a)-(c): Transition matrices [31, 34] of different noise types, where the simulated (a) pair flipping [9] and (b) symmetry flipping [40] are widely adopted in LNL works [6, 42], and (c) presents the realistic noise matrix in the VisDA-2017 dataset based on the black-box source model  $\widehat{F}^s$ . The value in row  $r$ , column  $c$  represents the probability with which samples of category  $r$  are assigned with label  $c$ . In all figures, deeper colour indicates a larger value and all values are rounded to the level of 0.01 (Zoom in to see the exact values).

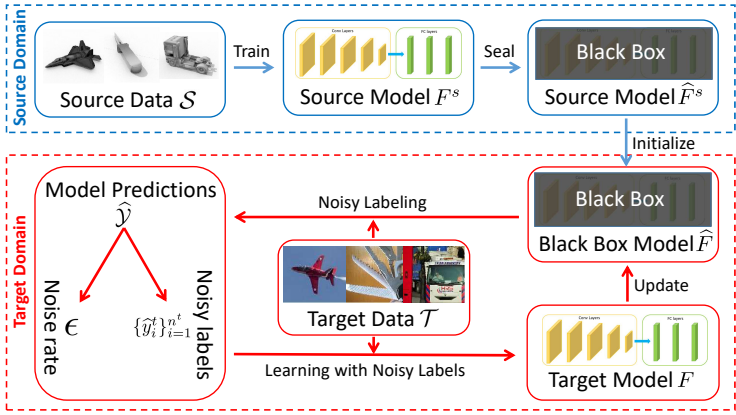


Figure 3: Framework of our proposed Iterative Learning with Noisy Labels (IterLNL), where we conduct noisy labeling and LNL iteratively.

target domains. Furthermore, we emphasize that the label noise in  $\{\hat{y}_i^t\}_{i=1}^m$  could be significantly unbalanced among categories; for example, the noise rate could be extremely high for some categories and extremely low for the others, as illustrated in Figure 2(c).

Such unbalanced label noise via domain shift is substantially different from the simulated ones [6, 40] in many LNL works, as compared in Figure 2. In addition, the noise rate, which is usually required in LNL algorithms, is unknown in B<sup>2</sup>UDA, while it is usually assumed to be given in LNL [6, 40]. In the next paragraph, we propose strategies to estimate the noise rate and tackle unbalanced label noise, which support the successful target model learning in B<sup>2</sup>UDA.

## 3.2 Learning with Noisy Labels

Given noisy target label predictions  $\hat{\mathcal{Y}}$  in Section 3.1, we resort to LNL to learn the target model. State-of-the-art LNL methods [6, 10, 24] usually combat noisy labels by selecting ‘clean’ samples from each mini-batch for training, which is achieved by utilizing the memorization effects of neuron networks [10]. Before going into detail, we denote  $R(n)$  as the percentage of instances selected for training in the mini-batch of  $n$ -th iteration. LNL methods [6, 10, 24] typically keep more instances in the mini-batch (i.e.,  $R(n)$  is large) at the beginning, and then gradually drop noisy samples (i.e.,  $R(n)$  becomes smaller) as training proceeds; by using more training instances at the beginning, a relatively reliable model could be achieved since deep models learn clean and easy patterns at the beginning [10]; with the reliable model, noisy instances could be filtered out by gradually dropping instances with larger losses.

We also adopt the aforementioned LNL strategy in our method since it presents high robustness even with extremely noisy labels [6]. In LNL [6, 44], the selecting percentage  $R(n)$  is depending on the noise rate, which is either assumed to be known in advance [6] or estimated with few labeled clean data [33, 47]. However, realistic noisy labels are introduced by domain shift in B<sup>2</sup>UDA, where the unavailable of labeled target data and unknown noise rate impede the design of  $R(n)$ .

To this end, we propose a simple yet efficient strategy to estimate noise rate. We first

follow [6] to define  $R(n)$  as:

$$R(n) = 1 - \min\left(\frac{n}{0.5N}\varepsilon, \varepsilon\right), \quad (2)$$

where  $N$  is the total number of training iterations and  $\varepsilon$  is the noise rate.

**Noise rate Estimation.** We first present the empirical noise rate  $\varepsilon$  as:

$$\varepsilon = 1 - \frac{1}{n^t} \sum_{i=1}^{n^t} \mathcal{I}[\mathbf{x}'_i, \hat{y}'_i], \quad (3)$$

where  $\mathcal{I}[\mathbf{x}'_i, \hat{y}'_i] \in \{0, 1\}$  is a binary indicator;  $\mathcal{I}[\mathbf{x}'_i, \hat{y}'_i] = 1$  if  $\hat{y}'_i$  is the correct label of  $\mathbf{x}'_i$  and 0 otherwise. It is obvious that the empirical noise rate  $\varepsilon$  (3) is close correlated to the classification accuracy of the black-box model  $\hat{F}$ . In the meantime, there is a correlation between the classification accuracy and maximum prediction probability, as observed in [14, 24, 25]. Although the prediction probability may be overconfident and misleading viewed in isolation, the probability statistics is often sufficient to reflect on the overall classification accuracy [8, 14], and also the noise rate  $\varepsilon$ .

To estimate the noise rate  $\varepsilon$ , we calculate the proportion of target data  $\mathcal{T}$  with high prediction probability as:

$$\rho = \frac{1}{n^t} \sum_{i=1}^{n^t} \mathbb{I}[\max(\hat{F}(\mathbf{x}'_i)) > \gamma], \quad (4)$$

where  $\gamma \in [0, 1]$  is the threshold and  $\mathbb{I}[\text{var}] = \begin{cases} 1, & \text{var} = \text{True} \\ 0, & \text{Otherwise} \end{cases}$ . And then we approximate noise rate  $\varepsilon$  as:

$$\varepsilon = 1 - \rho. \quad (5)$$

Although the estimated noise rate  $\varepsilon$  (5) is not precise, we find that such an estimation of noise rate works well in different tasks and achieves good results close to that using the grounding truth noise rate, as presented in Section 4.1.

**Category-wise Sampling.** Given the estimated noise rate  $\varepsilon$  (5), we could conduct LNL by selecting  $R(n)$  (2) percent samples with smaller loss for training in the mini-batch of  $n$ -th iteration. However, as we state in Section 3.1, the label noise is unbalanced among categories in B<sup>2</sup>UDA (cf. Figure 2(c)); thus samples in categories with higher noise rate are prone to present larger loss and be rejected for training, leading to worse results for these categories, as presented in Table 3.

To this end, we propose to sample the  $R(n)$  (2) percent samples with smaller loss for each category individually, where  $R(n)$  (2) is shared across categories. Technically, we introduce a probability queue buffer  $\mathbf{u}_k \in [0, 1]^h$  for category  $k \in [1, \dots, K]$ , where  $\mathbf{u}_k$  is initialized as a vector filled with positive infinity values and  $h$  is the buffer length. For any instance  $\mathbf{x}'$  in the  $n$ -th iteration, we obtain its corresponding noisy label  $\hat{y}' = \arg \max_k \hat{F}_k(\mathbf{x}')$  and loss  $\mathcal{L}(F(\mathbf{x}'), \hat{y}') = -\log(F_{\hat{y}'}(\mathbf{x}'))$ , where  $F$  is the model in learning. We propose the following indicator  $\mathbf{I}(\mathcal{L}(F(\mathbf{x}'), \hat{y}'), \mathbf{u}_{\hat{y}'}, n)$  to decide whether  $\mathbf{x}'$  should be used in training:

$$\mathbf{I}(\mathcal{L}(F(\mathbf{x}'), \hat{y}'), \mathbf{u}_{\hat{y}'}, n) = \begin{cases} 1, & \mathcal{L}(F(\mathbf{x}'), \hat{y}') \leq L_{R(n)}(\mathbf{u}_{\hat{y}'}) \\ 0, & \text{Otherwise,} \end{cases} \quad (6)$$

where  $L_{R(n)}(\mathbf{u}_{\hat{y}'})$  is the  $\lfloor hR(n) \rfloor$ -th largest value in  $\mathbf{u}_{\hat{y}'}$ . We utilize the loss  $\mathcal{L}(F(\mathbf{x}'), \hat{y}')$  to update current model  $F$  if  $\mathbf{I}(\mathcal{L}(F(\mathbf{x}'), \hat{y}'), \mathbf{u}_{\hat{y}'}, n) = 1$  and drop it otherwise.

Settings	Methods	U→M	S→M	M→U
$B^2$ UDA	Source Model	82.0	69.4	79.4
	sMDA [9]	83.4	69.9	81.2
	IterLNL	<b>97.6±0.1</b>	<b>97.7±0.1</b>	<b>97.7±0.0</b>
WBUDA	SDDA [13]	–	75.5	89.9
	PLR [14]	91.6±1.9	96.9±0.3	90.3±1.3
	3C-GAN [12]	99.3±0.1	99.4±0.1	97.3±0.2
	SHOT [13]	98.4±0.6	98.9±0.0	98.0±0.2
UDA	DANN [9]	86.3±0.3	85.5±0.4	84.9±0.6
	MCD [16]	–	96.2±0.4	96.5±0.3
	CDAN [12]	98.0	89.2	95.6
	RWOT [15]	97.5±0.2	98.8±0.1	98.5±0.2

Tasks	Methods	Acc.
$B^2$ UDA	Source Model	51.5
	sMDA [9]	53.1
	SoFA [16]	60.4
	IterLNL	<b>83.1</b>
WBUDA	PrDA [17]	76.7
	3C-GAN [12]	81.6
	SHOT [13]	82.9
UDA	DANN [9]	57.4
	MCD [16]	71.9
	RWOT [15]	84.0

Table 1: Results on Digits dataset.

Table 2: Results on VisDA-2017 (ResNet-101).

---

**Algorithm 1** Iterative Learning with Noisy Labels.

**Input:** Black-box source model  $\widehat{F}^s$ , target data  $\mathcal{T}$ 
**Output:** Target model  $F$ 

- 1: Initialize  $\widehat{F}$  with  $\widehat{F}^s$
  - 2: **for**  $m = 1$  to  $M$  **do** ▷ For each iterative step
  - 3:   Acquire noisy labels  $\widehat{\mathcal{Y}}$  with  $\mathcal{T}$  and  $\widehat{F}$  using (1)
  - 4:   Estimate noise rate  $\varepsilon$  with (4) and (5)
  - 5:   Initialize target model  $F$  and buffers  $\{\mathbf{u}_k\}_{k=1}^K$
  - 6:   **for**  $n = 1$  to  $N$  **do** ▷ For each iteration
  - 7:     Acquire  $R(n)$  with (2)
  - 8:     Update model  $F$  using data selected with (6)
  - 9:     Update buffers  $\{\mathbf{u}_k\}_{k=1}^K$
  - 10:   **end for**
  - 11:   Update  $\widehat{F}$  as the API (i.e., black-box model) of  $F$
  - 12: **end for**
- 

We also update the queue buffers  $\{\mathbf{u}_k\}_{k=1}^K$  with all samples in the  $n$ -th iteration. Specifically, given an instance  $\mathbf{x}^t$  and its corresponding noisy label  $\widehat{y}^t$  and loss  $\mathcal{L}(F(\mathbf{x}^t), \widehat{y}^t)$  defined above, we push  $\mathcal{L}(F(\mathbf{x}^t), \widehat{y}^t)$  into the queue buffer  $\mathbf{u}_{\widehat{y}^t}$  and pop the oldest value from  $\mathbf{u}_{\widehat{y}^t}$  simultaneously. In this way, we adopt samples with the  $R(n)$  percent smallest losses in each category for training in  $n$ -th iteration.

### 3.3 Iterative Learning Strategy

With the noisy labeling in Section 3.1 and LNL in Section 3.2, we could get a more reliable target model over the original black-box one (i.e., the one introduces noisy labels), as illustrated in Figure 4(a). In other words, the achieved target model could produce improved noisy labels over the original noisy labels. It is a natural idea to conduct noisy labeling with the achieved target model (or its black-box counterpart), and conduct LNL on the new noisy labels again. Finally, we define our algorithmic framework by conducting noisy labeling and LNL iteratively, leading to the Iterative Learning with Noisy Labels (IterLNL). We summarize IterLNL in Algorithm 1 and illustrate its framework in Figure 3.

**Remarks.** Although we tackle  $B^2$ UDA with a LNL-like method, it should be noted that tasks of  $B^2$ UDA and LNL are fundamentally different, since a black-box source model, rather than noisy labels, is given in  $B^2$ UDA to help the learning with unlabeled data. Other ways of source model utilization, e.g., model distillation [18], are left for further studies. Additionally, we adapt the LNL method for  $B^2$ UDA by estimating noise rate with unlabeled target data (cf. Sec. 3.2), tackling category unbalanced label noise via category-wise sam-

pling (cf. Sec. 3.2), and utilizing improved models via iterative learning (cf. Sec. 3.3), leading to an algorithmic framework specifically designed for B<sup>2</sup>UDA and significant performance improvement over vanilla LNL methods (cf. Sec. 4). Note that there are significant differences between our IterLNL and popular self-training in UDA; firstly, our IterLNL, as well as all LNL methods, is proposed for tasks with noisy labeled training data while self-training is applied to tasks with labeled and unlabeled training data, impeding its application in B<sup>2</sup>UDA; secondly, samples with high prediction confidence are directly used for model training in self-training while LNL methods and our IterLNL use samples to learn models only if their current model predictions are consistent with the given noisy labels. Finally, we propose a new framework for UDA by modeling influence of domain shift as noisy labels and tackling UDA with LNL-like methods. It should be noted that the domain shift only affects the noise rate of initial noisy labels and makes no influence to the following model learning via LNL-like methods.

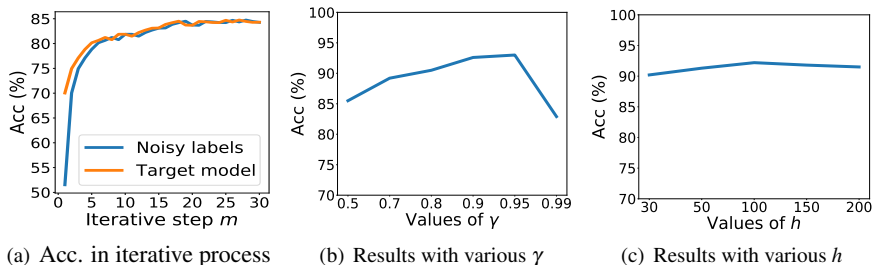


Figure 4: (a) Illustration of the accuracy of noisy labels  $\{\tilde{y}_i^n\}_{i=1}^n$  and target model  $F$  via LNL. (b-c) Hyper-parameter ablations with various values of  $\gamma$  (b) and  $h$  (c).

## 4 Experiment

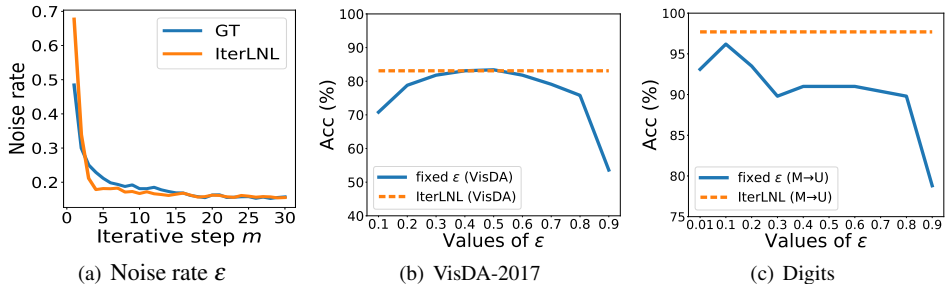
**Office-31** [15] is the most popular benchmark dataset for UDA. There are 4,110 samples shared by three domains: amazon (A), webcam (W), and dslr (D). **VisDA-2017** [12] aims to transfer knowledge from synthetic images to real-world ones, which is a challenging task with significant domain divergence. There are 152K synthetic images and 55K real-world images shared by 12 classes. Datasets of MNIST [19], Street View House Numbers (SVHN) [28], and USPS [13] constitute the **Digits** task, which includes 10 classes. There are 50K training samples, 10K validation samples and 10K test samples in the MNIST dataset (M), where all images are black-and-white handwritten digits; the SVHN dataset (S) contains 73,257 training and 26,032 test images with colored backgrounds; the USPS dataset (U) contains 7,291 training and 2,007 test images with black backgrounds.

**Implementation Details.** For experiments on datasets of Office-31 and VisDA-2017, we employ the pre-trained ResNet model [9] as the backbone and replace the last fully connected (FC) layer with a task-specific FC classifier following [1, 23, 24]. We introduce the source model  $F^S$  by fine-tuning the constructed model on source data following [15] and then seal  $F^S$  as the black-box  $\hat{F}^S$ , i.e., only the input-output interface of  $F^S$  is available. For experiments on Digits dataset, we follow [16] to introduce the source model  $F^S$  with convolutional layers and FC layers. Following [9], we utilize the SGD optimizer and adopt the learning rate strategy as  $\eta_n = \frac{\eta_0}{(1+10\zeta)^{0.75}}$ , where  $\eta_0$  is initial learning rate and  $\zeta$  is the process of training iterations linearly changing from 0 to 1. Hyper-parameters of our method are tuned on the A→W task of Office31 dataset and apply to other tasks. We set the batch size as 64,  $\eta_0 = 0.003$ ,  $\gamma = 0.9$  in (4), and buffer length  $h = 100$  for all experiments.



Methods	plane	bicycl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	truck	cate. avg	ins. avg
Source Model	73.2	30.7	57.0	79.5	75.4	7.4	88.8	11.4	84.0	29.1	80.0	2.0	51.5	57.7
IterLNL (w/o Iter)	89.8	57.8	77.1	88.0	89.6	21.7	<b>94.2</b>	37.8	91.6	65.4	86.3	11.3	67.6	71.5
IterLNL (w/o CateS)	<b>95.6</b>	<b>87.2</b>	<b>86.0</b>	<b>90.0</b>	<b>96.9</b>	0.0	93.7	51.2	<b>93.3</b>	<b>88.1</b>	<b>87.4</b>	0.0	72.4	75.6
IterLNL	88.7	83.4	78.3	67.7	91.4	<b>87.6</b>	91.8	<b>79.5</b>	86.2	86.7	78.7	<b>77.2</b>	<b>83.1</b>	<b>81.2</b>

Table 3: Ablation study on VisDA-2017 dataset (ResNet-101).

Figure 5: Illustration of (a) estimated noise rate in different iterative steps and (b-c) IterLNL’s results with fixed  $\epsilon$  in different iterative steps on VisDA-2017 and Digits dataset.

## 4.1 Ablation Study and Analyses

**Ablation study.** We introduce several variants of IterLNL to investigate the individual components in IterLNL. Specifically, following [8], we replace the category-wise sampling (6) by simply using the  $R(n)$  percent samples with smaller losses in the  $n$ -th iteration for model learning, leading to ‘IterLNL (w/o CateS)’. We also present the results of IterLNL by conducting noisy labeling and learning with noisy labels only once (i.e., setting  $M = 1$  in Algorithm 1, resulting in ‘IterLNL (w/o Iter)’. As illustrated in Table 3, IterLNL improves over the IterLNL (w/o CateS) and IterLNL (w/o Iter) significantly, justifying the efficacy of category-wise sampling (6) and iterative learning. Specifically, the category-wise sampling (6) largely alleviates the biased prediction prior, which is due to the unbalanced label noise across categories (cf. Figure 2(c)), and leads to more balanced results and better mean accuracy. We also intuitively visualize the accuracy improvement of model  $F$  in the iterative learning process (i.e., with different  $m \in [1, M]$ ). As illustrated in Figure 4(a), the accuracy of  $F$  via LNL significantly outperforms that of initial noisy labels in the beginning; the improvement is gradually reduced as the iterative step  $m$  increases, leading to the final convergence. As presented in Figure 5(a), the noise rate estimated via our method (5) approximates that calculated with labeled target data (i.e., GT in Figure 5(a)) and, on VisDA-2017 dataset, IterLNL achieves 83.4% with GT noise rate, which is close to our result of 83.1%. We also make in-depth analysis for the noise rate estimation by utilizing fixed noise rate in all iterative steps, which is illustrated in Figures 5(b) and 5(c), and detailed in the appendices.

**Analyses on  $\gamma$  and  $h$ .** We investigate the hyper-parameters  $\gamma$  (4) and buffer length  $h$  in Section 3.2 on A $\rightarrow$ W task of Office-31 dataset. As illustrated in Figure 4(b) and 4(c),  $\gamma \in [0.9, 0.95]$  leads to the best result and IterLNL performs robustly under a wide range of  $h$  values. We empirically set  $\gamma = 0.9$  and  $h = 100$ , which work well for all tasks but not necessarily result in the best results.

**Comparison with LNL Method.** We conduct the LNL methods of Co-teaching [8] and DivideMix [20] with our estimated noise rate  $\epsilon$  (5) and noisy labels  $\{\hat{y}_i^n\}_{i=1}^n$  introduced by

Tasks	Method	A→D	A→W	D→A	D→W	W→A	W→D	Avg
B <sup>2</sup> UDA	Source Model [15]	79.7	78.1	64.9	96.0	65.4	99.2	80.1
	BPDA [16]	89.4	88.4	61.4	93.3	60.7	94.8	81.3
	sMDA [9]	80.5	79.3	65.7	96.3	67.3	99.2	81.4
	IterLNL	<b>92.6±0.3</b>	<b>92.2±0.0</b>	<b>74.3±1.3</b>	<b>98.0±0.1</b>	<b>74.3±0.0</b>	<b>99.4±0.0</b>	<b>88.5</b>
WBUDA	SDDA [18]	85.3	82.5	66.4	99.0	67.7	99.8	83.5
	PrDA [16]	91.1±0.3	98.2±0.3	99.5±0.2	92.2±0.2	71.0±0.2	71.2±0.2	87.2
	SHOT [13]	94.0	90.1	74.7	98.4	74.3	99.9	88.6
	3C-GAN [17]	92.7±0.4	93.7±0.2	75.3±0.5	98.5±0.1	77.8±0.1	99.8±0.2	89.6
UDA	DANN [9]	79.7±0.4	82.0±0.4	68.2±0.4	96.9±0.2	67.4±0.5	99.1±0.1	82.2
	CDAN [19]	92.9±0.2	94.1±0.1	71.0±0.3	98.6±0.1	69.3±0.3	100.0±0.0	87.7
	RWOT [16]	94.5±0.2	95.1±0.2	77.5±0.1	99.5±0.2	77.9±0.3	100.0±0.0	90.8

Table 4: Results on Office31 dataset, where all methods are based on a ResNet-50 model.

the black-box source model  $\hat{F}^s$  on the S→M task. The results of 91.0% with Co-teaching and 93.3% with DivideMix are lower than 97.7% with our IterLNL, justifying the advantage of IterLNL over vanilla LNL method on B<sup>2</sup>UDA.

**Comparison with Knowledge Distillation [10] and Label Smoothing [27].** To adapt knowledge distillation to B<sup>2</sup>UDA, we promote target student model to produce similar prediction probabilities (e.g., soft labels) to the black-box source model, leading to a result of 56.9% on VisDA-2017. Besides, we learn the target model with smoothed pseudo label introduced by the black-box source model [27] with a label smoothing factor of 0.1, resulting in a result of 57.1%. Compared to IterLNL (83.1%), they are less effective for B<sup>2</sup>UDA, possibly since knowledge distillation promotes the target student to be similar to the not-good source teacher, and the label smoothing only partially alleviates the problem of label noise.

## 4.2 Results

The results of IterLNL on datasets of Digits, Office31, and VisDA-2017 are illustrated in Table 1, Table 4, and Table 2, respectively. Most comparable results are directly reported from their original papers, except the Source Model [15] and sMDA [9], which are implemented by ourselves. Taking advantages of feature learning, our IterLNL improves over existing methods of B<sup>2</sup>UDA [9, 16] significantly; for example, IterLNL improves over sMDA [9] and SoFA [16] by 30.0% and 22.7% on the VisDA-2017 dataset respectively. Besides, IterLNL also outperforms WBUDA methods on VisDA-2017 dataset. All in all, although we only use the black-box source model for the transfer use in target domain, IterLNL achieves comparable results to methods of WBUDA and traditional UDA.

## 5 Discussions and Broader Impact

Considering that the source model itself may not be available due to commercial and/or safety considerations [24], we investigate the B<sup>2</sup>UDA task, and propose a baseline algorithm by modeling domain shift as the initial label noise and tackling DA with a LNL-like method. We verify its efficacy on popular DA benchmarks with thorough experiments. The B<sup>2</sup>UDA task is of broad practical value since it could further improve the utilities of APIs as cloud services, pushing the DA research closer to practical applications.

**Acknowledgement.** This work was partially supported by the Guangdong R&D key project of China (No.: 2019B010155001), the National Natural Science Foundation of China (No.: 61771201), and the Program for Guangdong Introducing Innovative and Entrepreneurial Teams (No.: 2017ZT07X183).

## References

- [1] Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *International Conference on Machine Learning*, pages 233–242. PMLR, 2017.
- [2] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010.
- [3] Boris Chidlovskii, Stephane Clinchant, and Gabriela Csurka. Domain adaptation in the absence of source domain data. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 451–460, 2016.
- [4] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.
- [5] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. *arXiv preprint arXiv:1706.04599*, 2017.
- [6] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in neural information processing systems*, pages 8527–8537, 2018.
- [7] Bo Han, Gang Niu, Xingrui Yu, Quanming Yao, Miao Xu, Ivor Tsang, and Masashi Sugiyama. Sigua: Forgetting may make learning with noisy labels more robust. In *International Conference on Machine Learning*, pages 4006–4016. PMLR, 2020.
- [8] Bo Han, Quanming Yao, Tongliang Liu, Gang Niu, Ivor W Tsang, James T Kwok, and Masashi Sugiyama. A survey of label-noise representation learning: Past, present and future. *arXiv preprint arXiv:2011.04406*, 2020.
- [9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [10] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *ICLR*, 2017.
- [11] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.
- [12] Yunzhong Hou and Liang Zheng. Source free domain adaptation with image translation. *arXiv preprint arXiv:2008.07514*, 2020.
- [13] Jonathan J. Hull. A database for handwritten text recognition research. *IEEE Transactions on pattern analysis and machine intelligence*, 16(5):550–554, 1994.

- [14] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pages 2304–2313. PMLR, 2018.
- [15] Mingsheng Long Junguang Jiang, Bo Fu. Transfer-learning-library. <https://github.com/thuml/Transfer-Learning-Library>, 2020.
- [16] Youngeun Kim, Donghyeon Cho, Priyadarshini Panda, and Sungeun Hong. Progressive domain adaptation from a source pre-trained model. *arXiv preprint arXiv:2007.01524*, 2020.
- [17] Youngeun Kim, Sungeun Hong, Donghyeon Cho, Hyoungseob Park, and Priyadarshini Panda. Domain adaptation without source data. *arXiv preprint arXiv:2007.01524*, 2020.
- [18] Vinod K Kurmi, Venkatesh K Subramanian, and Vinay P Namboodiri. Domain impression: A source data free domain adaptation method. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 615–625, 2021.
- [19] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [20] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 2, 2013.
- [21] Junnan Li, Richard Socher, and Steven C.H. Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *International Conference on Learning Representations*, 2020.
- [22] Rui Li, Qianfen Jiao, Wenming Cao, Hau-San Wong, and Si Wu. Model adaptation: Unsupervised domain adaptation without source data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9641–9650, 2020.
- [23] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. *arXiv preprint arXiv:2002.08546*, 2020.
- [24] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I. Jordan. Learning transferable features with deep adaptation networks. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37, ICML'15*, pages 97–105. JMLR.org, 2015. URL <http://dl.acm.org/citation.cfm?id=3045118.3045130>.
- [25] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pages 1640–1650, 2018.
- [26] Pietro Morerio, Riccardo Volpi, Ruggero Ragonese, and Vittorio Murino. Generative pseudo-label refinement for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3130–3139, 2020.

- [27] Rafael Müller, Simon Kornblith, and Geoffrey Hinton. When does label smoothing help? *arXiv preprint arXiv:1906.02629*, 2019.
- [28] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.
- [29] OpenAI. Why did openai choose to release an api instead of open-sourcing the models? [EB/OL]. <https://openai.com/blog/openai-api/> Accessed March 4, 2021.
- [30] Sinno Jialin Pan, Qiang Yang, et al. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.
- [31] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1944–1952, 2017.
- [32] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017.
- [33] Harish Ramaswamy, Clayton Scott, and Ambuj Tewari. Mixture proportion estimation via kernel embeddings of distributions. In *International conference on machine learning*, pages 2052–2060. PMLR, 2016.
- [34] Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.
- [35] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European conference on computer vision*, pages 213–226. Springer, 2010.
- [36] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3723–3732, 2018.
- [37] Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. *ICLRW*, 2015.
- [38] Hui Tang and Kui Jia. Discriminative adversarial domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5940–5947, 2020.
- [39] Brendan Van Rooyen and Robert C Williamson. A theory of learning with corrupted labels. *J. Mach. Learn. Res.*, 18(1):8501–8550, 2017.
- [40] Brendan Van Rooyen, Aditya Krishna Menon, and Robert C Williamson. Learning with symmetric label noise: The importance of being unhinged. *NIPS*, 2015.
- [41] Markus Weber, Max Welling, and Pietro Perona. Unsupervised learning of models for recognition. In *European conference on computer vision*, pages 18–32. Springer, 2000.

- [42] Kunhong Wu, Yucheng Shi, Yahong Han, Yunfeng Shao, and Bingshuai Li. Black-box probe for unsupervised domain adaptation without model transferring. *arXiv preprint arXiv:2107.10174*, 2021.
- [43] Renjun Xu, Pelen Liu, Liyan Wang, Chao Chen, and Jindong Wang. Reliable weighted optimal transport for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4394–4403, 2020.
- [44] Hansi Yang, Quanming Yao, Bo Han, and Gang Niu. Searching to exploit memorization effect in learning from corrupted labels. *arXiv preprint arXiv:1911.02377*, 2019.
- [45] Shiqi Yang, Yaxing Wang, Joost van de Weijer, and Luis Herranz. Unsupervised domain adaptation without source data by casting a bait. *arXiv preprint arXiv:2010.12427*, 2020.
- [46] Hao-Wei Yeh, Baoyao Yang, Pong C Yuen, and Tatsuya Harada. Sofa: Source-data-free feature alignment for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 474–483, 2021.
- [47] Xiyu Yu, Tongliang Liu, Mingming Gong, Kayhan Batmanghelich, and Dacheng Tao. An efficient and provable approach for mixture proportion estimation using linear independence assumption. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4480–4489, 2018.
- [48] Yabin Zhang, Hui Tang, Kui Jia, and Mingkui Tan. Domain-symmetric networks for adversarial domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5031–5040, 2019.
- [49] Yabin Zhang, Bin Deng, Hui Tang, Lei Zhang, and Kui Jia. Unsupervised multi-class domain adaptation: Theory, algorithms, and practice. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.
- [50] Yuchen Zhang, Tianle Liu, Mingsheng Long, and Michael Jordan. Bridging theory and algorithm for domain adaptation. In *International Conference on Machine Learning*, pages 7404–7413, 2019.
- [51] Zhilu Zhang and Mert R Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *arXiv preprint arXiv:1805.07836*, 2018.
- [52] Tianyi Zhou, Shengjie Wang, and Jeff Bilmes. Time-consistent self-supervision for semi-supervised learning. In *International Conference on Machine Learning (ICML)*, 2020.