Semi-Supervised Domain Adaptation with Source Label Adaptation

Yu-Chu Yu   Hsuan-Tien Lin
National Taiwan University
{r09922104, htlin}@csie.ntu.edu.tw

Abstract

Semi-Supervised Domain Adaptation (SSDA) involves learning to classify unseen target data with a few labeled and lots of unlabeled target data, along with many labeled source data from a related domain. Current SSDA approaches usually aim at aligning the target data to the labeled source data with feature space mapping and pseudo-label assignments. Nevertheless, such a source-oriented model can sometimes align the target data to source data of the wrong classes, degrading the classification performance. This paper presents a novel source-adaptive paradigm that adapts the source data to match the target data. Our key idea is to view the source data as a noisily-labeled version of the ideal target data. Then, we propose an SSDA model that cleans up the label noise dynamically with the help of a robust cleaner component designed from the target perspective. Since the paradigm is very different from the core ideas behind existing SSDA approaches, our proposed model can be easily coupled with them to improve their performance. Empirical results on two state-of-the-art SSDA approaches demonstrate that the proposed model effectively cleans up the noise within the source labels and exhibits superior performance over those approaches across benchmark datasets. Our code is available at https://github.com/chu0802/SLA.

1. Introduction

Domain Adaptation (DA) focuses on a general machine learning scenario where training and test data may originate from two related but distinct domains: the source domain and the target domain. Many works have extensively studied unsupervised DA (UDA), where labels in the target domain cannot be accessed, from both theoretical [2, 19, 36] and algorithmic [5, 8, 15, 16, 22, 37] perspectives. Recently, Semi-Supervised Domain Adaptation (SSDA), another DA setting that allows access to a few target labels, has received more research attention because it is a simple yet realistic setting for application needs.

The most naïve strategy for SSDA, commonly known as S+T [21, 33], aims to train a model using the source data and labeled target data with a standard cross entropy loss. This strategy often suffers from a well-known domain shift issue, which stems from the gap between different data distributions. To address this issue, many state-of-the-art algorithms attempt to explore better use of the unlabeled target data so that the target distribution can be aligned with the source distribution. Recently, several Semi-Supervised Learning (SSL) algorithms have been applied for SSDA [12, 21, 30] to regularize the unlabeled data, such as entropy minimization [6], pseudo-labeling [11, 24] and consistency regularization [1, 24]. These classic source-oriented strategies have prevailed for a long time. However, these algorithms typically require the target data to closely match some semantically similar source data in the feature space. Therefore, if the S+T space has been misaligned, it can be challenging to recover from the misalignment, as illustrated in Figure 1.

We take a deeper look into a specific example from the Office-Home dataset [27] to confirm the abovementioned issue. Figure 2 visualizes the feature space trained by S+T using t-SNE [3]. We observed that the misalignment between
Figure 2. Feature visualizations with t-SNE for an example of the misalignment on the Office-Home A → C dataset with ResNet34. The model is trained by S+T. Left: 0-th iteration. Right: 5000-th iteration. We observe that the misalignment has already happened at a very early stage. Guided by source labels and a few target labels, a portion of the target data from the 59th class misaligns with the source data from the 7th class.

Table 1. A partial confusion matrix of S+T on the 3-shot Office-Home A → C dataset with ResNet34.

<table>
<thead>
<tr>
<th>True\Pred</th>
<th>Class 7</th>
<th>Class 59</th>
<th>Class 41</th>
<th>Others</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 59</td>
<td>38.5%</td>
<td>19.8%</td>
<td>13.5%</td>
<td>28.2%</td>
</tr>
</tbody>
</table>

2. Related Work

Problem Setup. DA focuses on a K-class classification task with an m-dimensional input space $X \subseteq \mathbb{R}^m$ and a set of labels $\{1, 2, \ldots, K\}$. For simplicity, we define a label space $Y$ on the probability simplex $\Delta^K$. A label $y = k \in \{1, 2, \ldots, K\}$ is equivalent to a one-hot encoded vector $y \in Y$, where only the $k$-th element is 1 and the others are 0. We consider two domains over $X \times Y$, named source domain $D_s$ and target domain $D_t$. In SSDA, we sample an amount of labeled source data $S = \{(x^s_i, y^s_i)\}_{i=1}^{|S|}$ from $D_s$, labeled target data $L = \{(x^t_i, y^t_i)\}_{i=1}^{|L|}$ from $D_t$, and unlabeled target data $U = \{x^t_{u_i}\}_{i=1}^{|U|}$ from the marginal distribution of $D_t$ over $X$. Typically, $|L|$ is considerably smaller than $|S|$ and $|U|$, such as one or three examples per class. Our goal is to train an SSDA model $g$ with $S$, $L$, and $U$ to perform well on the target domain.

Semi-Supervised Domain Adaptation (SSDA). SSDA can be viewed as a relaxed yet realistic version of UDA. An SSDA algorithm usually involves three loss functions:

$$\mathcal{L}_{SSDA} = \mathcal{L}_s + \mathcal{L}_t + \mathcal{L}_u$$

(1)

where $\mathcal{L}_s$ stands for the loss derived by the source data. $\mathcal{L}_t, \mathcal{L}_u$ denotes the losses from the labeled and unlabeled target data. As discussed in Section 1, based on S+T, a typical SSDA algorithm usually focuses on designing $\mathcal{L}_u$ to better align the target data with the source data. Recently, many existing works have borrowed SSL techniques to conquer SSDA because of the problem similarity [35]. [21]
proposes a variant of entropy minimization [6] to explicitly align the target data with source clusters. [31] decomposes SSDA into an SSL and a UDA task. The two different sub-tasks produce pseudo labels respectively, and learn from each other via co-training. [12] groups target features into clusters by measuring pairwise feature similarity. [30] utilizes consistency regularization at three different levels to perform domain alignment. Besides, both [12, 30] apply pseudo labeling with data augmentations [24] to enhance their performance. To the best of our knowledge, all methods listed above mainly explore the usage of unlabeled target data while treating the source data with the most straightforward strategy. In our study, we noticed that source labels could appear noisy from the viewpoint of the target data. We thus developed a source-adaptive framework to gradually adapt the source data to the target space. Since we are addressing a new facet of the issue, our framework can be easily applied to several SSDA algorithms mentioned above, further improving the overall performance.

**Noisy Label Learning (NLL).** The effectiveness of a machine learning algorithm highly depends on the quality of collected labels. With regard to the present deep neural network design [7], the aforementioned issue could worsen as deep models have the capability to fit the data set in a seemingly random manner, regardless of the quality of the labels [34]. To clean the noisy labels, [20] proposes a smoothing mechanism to mix noisy labels with self-prediction. [26] models clean labels as trainable parameters and designs a joint optimization algorithm to alternatively update parameters. [17, 25, 32] estimate a transition matrix to correct the corrupted labels. However, learning a global transition matrix usually need a strong assumption of how noisy labels come from, which is difficult to verify in the real-world scenarios [29]. [38] trains a label correction network in a meta-learning manner to help correct noisy labels. Motivated by [20, 38], we propose a simple framework that can efficiently build a label adaptation model to correct the noisy source labels.

### 3. Proposed Framework

Next, we propose a novel SSDA framework, **Source Label Adaptation.** An overview of our proposed framework is shown in Figure 3. In Section 3.1, we connect the (SS)DA problem to NLL and point out that a classic NLL method [20] cannot be directly applied to solve SSDA. In Section 3.2, we review a classic few-shot learning algorithm, **Prototypical Network [23]** and propose **Protonet with Pseudo Centers** to better estimate the prototypes. In Section 3.3, we summarize our framework and describe the implementation details.

#### 3.1. Domain Adaptation as Noisy Label Learning

In Domain Adaptation, we seek an ideal model \( g^* \) that can minimize unlabeled target risk. Ideally, the most suitable label for a source instance \( x_s^i \) in the target space should be \( g^*(x_s^i) \). That is, the ideal source loss \( L_s^* \) is:

\[
L_s^*(g|S) = \frac{1}{|S|} \sum_{i=1}^{|S|} H(g(x_s^i), g^*(x_s^i)),
\]

where \( H \) measures the cross entropy between two distributions.

Combining with the labeled target loss \( L_{\ell} \), we refer to the model trained by \( L_s^* \) and \( L_{\ell} \) as ideally-adapted S+T.
output from an ideal model. As an ideal clean label is the case, doing the correction is nearly equivalent to not doing prediction is almost the same as the original label. In this be close to [20].

We define an adapted labels \( \tilde{y}_i \) according to Eq. 3.

\[
\tilde{y}_i = (1 - \alpha) \cdot y_i^* + \alpha \cdot g(x_i^*)
\]  

Then, the modified source loss \( \hat{L}_s \) is:

\[
\hat{L}_s(g | S) = \frac{1}{|S|} \sum_{i=1}^{|S|} H(g(x_i^*), \hat{y}_i)
\]  

However, in DA, such a method might not be helpful since the model usually overfits the source data, which makes \( g(x_i^*) \approx y_i^* \). That is, the modified source label \( \hat{y}_i \) can be almost the same as the original source label \( y_i^* \) according to Eq. 3.

Figure 4 shows that when doing label correction with self-prediction, the KL divergence from \( y^* \) to \( g(x^*) \) could be close to 0 after 2000 iterations, indicating that the self-prediction is almost the same as the original label. In this case, doing the correction is nearly equivalent to not doing so.

To benefit from the modified labels, we need to eliminate supervision from source data. As an ideal clean label is the output from an ideal model \( g^* \), we should instead find a label adaptation model \( g_c \) that can approximate the ideal model and adapt the source labels to the view of target data. We define an adapted labels \( \hat{y}_i^* \) as a convex combination between the original labels \( y_i^* \) and the output from \( g_c \), which is the same as [20].

\[
\hat{y}_i^* = (1 - \alpha) \cdot y_i^* + \alpha \cdot g_c(x_i^*)
\]

The results of the ideally-adapted S+T reveal the full potential to adapt source labels. As shown in Table 2, there is a significant difference in performance between a standard S+T and an ideally-adapted S+T, demonstrating that performance can be dramatically affected by only modifying the source labels.

In practice, however, we can only approximate the ideal model. To address the issue, we take the original source labels as a noisy version of the ideal labels and approach DA as a NLL problem. We first apply a simple method proposed by [20] to help correct the source labels, which we refer to it as label correction with self-prediction [28]. Specifically, for each source instance \( x_i^* \), we construct the modified source label \( \hat{y}_i^* \) by combining the original label \( y_i^* \) and the prediction from the current model \( g \) with a ratio \( \alpha \).

\[
\hat{y}_i = (1 - \alpha) \cdot y_i^* + \alpha \cdot g(x_i^*)
\]

Then, the modified source loss \( \hat{L}_s \) is:

**Table 2.** Accuracy (%) of S+T and ideally-adapted S+T on the 3-shot OfficeHome dataset with ResNet34.

<table>
<thead>
<tr>
<th>Method</th>
<th>A → C</th>
<th>P → C</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-shot</td>
<td>3-shot</td>
</tr>
<tr>
<td>S+T</td>
<td>52.9</td>
<td>58.1</td>
</tr>
<tr>
<td>ideally-adapted S+T</td>
<td>82.9</td>
<td>87.4</td>
</tr>
</tbody>
</table>

Figure 4. Average KL divergence from \( y^* \) to \( g(x^*) \) at each iteration (3-shot Office-Home A → C with ResNet34, smoothing by EMA with a ratio 0.8).

### 3.2. Protonet with Pseudo Centers

In the semi-supervised setting, we can access a few target labels. Nonetheless, learning from a limited number of target labels might suffer from a severe overfitting issue. Thus, we learn a prototypical network (protonet) [23] to overcome the few-shot problem.

Given a dataset \( \{x_i, y_i\}_{i=1}^N \) and a feature extractor \( f \); let \( N_k \) denote the number of data labeled with \( k \). The prototype of class \( k \) is defined as the center of features with the same class:

\[
c_k = \frac{1}{N_k} \sum_{i=1}^N \mathbb{1}\{y_i = k\} \cdot f(x_i).
\]

Let \( C_f = \{c_1, \ldots, c_K\} \) collects all centers with extractor \( f \). We define \( P_{C_f} : X \mapsto Y \) as a protonet with centers \( C_f \):

\[
P_{C_f}(x_i)_k = \frac{\exp(-d(f(x_i), c_k) \cdot T)}{\sum_{j=1}^K \exp(-d(f(x_i), c_j) \cdot T)}
\]

Here \( d : F \times F \mapsto [0, \infty) \) is a distance measure over feature space \( F \), usually measuring Euclidean distance. \( T \) is a hyper-parameter that controls the smoothness of output. As \( T \to 0 \), the output of a protonet would be close to a uniform distribution.

Since we have access to the labeled target dataset \( L \), by Eq. 6 and Eq. 7, we can derive labeled target centers \( C^l_f \) and construct a protonet with labeled target centers \( P_{C^l_f} \).

[23] demonstrated that when \( d \) measures Euclidean distance, a protonet is equivalent to a linear classifier with particular parameterization over \( F \). Thus, we can take the protonet as a label adaptation model over a particular feature space. The protonet with labeled target centers is purely built from the viewpoint of target data, which should reduce our concerns about the issue mentioned in Section 3.1.
To better estimate the ideal centers, we propose to find the \( \tilde{\text{label}} \) adaptation loss \( \tilde{\ell} \) and compute the modified source label \( \tilde{\ell} \) for each source instance. Place the typical source loss with a standard cross entropy loss. For each source instance \( x_i \), we can get pseudo centers \( C_f \) by Eq. 6, and further define a Protonet with Pseudo Centers (PPC) \( P_{C_f} \) by Eq. 7.

Table 3 compares the average L2 distance from ideal centers \( C_f \) to labeled target centers \( C_f \) and pseudo centers \( \tilde{C}_f \) over the feature space trained by S+T. The distance between \( C_f \) and \( \tilde{C}_f \) is significantly shorter than the distance between \( C_f \) and \( C_f \), which means the pseudo centers are indeed much closer to the ideal centers.

Taking PPC as the label adaptation model, the modified label \( \tilde{y}_i \) turns out to be:

\[
\tilde{y}_i = (1 - \alpha) \cdot y_i + \alpha \cdot P_{C_f}(x_i)
\]

### 3.3. Source Label Adaptation for SSDA

We propose a label adaptation loss for unlabeled source data to replace the typical source loss with a standard cross entropy loss. For each source instance \( x_i \) with label \( y_i \), we first compute the modified source label \( \tilde{y}_i \) by Eq. 9. Then, the label adaptation loss \( \hat{\mathcal{L}}_a \) is:

\[
\hat{\mathcal{L}}_a(g|S) = \frac{1}{|S|} \sum_{i=1}^{|S|} \mathcal{H}(g(x_i), \tilde{y}_i)
\]

Our framework, Source Label Adaptation (SLA) for SSDA, can be trained by the following loss function:

\[
\mathcal{L}_{\text{SSDA w/ SLA}} = \hat{\mathcal{L}}_a + \mathcal{L}_t + \mathcal{L}_u
\]

\( \mathcal{L}_t \) is the loss function for labeled target data \( L \), which can still be a standard cross entropy loss. In contrast to other widely used methods, we primarily concentrate on improving the usage of source data. Therefore, the loss function for unlabeled target data \( \mathcal{L}_u \) can be derived through any state-of-the-art algorithm, and our framework can be easily coupled with other methods without contradiction.

#### 3.3.1 Implementation Details

**Warmup Stage.** Our label adaptation framework relies on the quality of the predicted pseudo labels. However, the prediction from the initial model can be noisy. Thus, we introduce a hyperparameter \( W \) for warmup to get more stable pseudo labels. During the warmup stage, we train our model normally with original source labels. Specifically, at the \( e \)-th iteration, we compute the modified source label \( \tilde{y}_i^e \) as follows:

\[
\tilde{y}_i^e = \begin{cases} 
  y_i^e & \text{if } e \leq W \\
  (1 - \alpha) \cdot y_i^e + \alpha \cdot P_{C_f}(x_i) & \text{otherwise}
\end{cases}
\]

**Dynamic Update.** The feature space and the predicted pseudo labels constantly evolve during the training phase. By updating the pseudo labels and centers, we can remain the quality of the projected pseudo centers the same. It would be ideal for updating the centers at each iteration. In practice, we update the pseudo labels through Eq. 8 and update centers with the current feature extractor \( f \) through Eq. 6 for every specific interval \( I \). Prior works [14] have addressed a similar issue and proposed to maintain a memory bank for dynamic updates of the estimated centers. However, in our framework, we need to update both the estimated centers and pseudo labels simultaneously. Therefore, we decided to adopt a more straightforward solution to mitigate the demands on time and complexity.

### 4. Experiments

We first sketch our experiment setup, including data sets, competing methods, and parameter settings in Section 4.1. We then present experimental results to validate the superiority of the proposed SLA framework in Section 4.2. We further analyze our proposed framework and highlight the limitation in Section 4.3.

#### 4.1. Experiment Setup

**Datasets.** We evaluate our proposed SLA framework on two sets of SSDA benchmarks, including Office-Home [27] and DomainNet [18]. Office-Home is a mainstream benchmark for both UDA and SSDA. It contains four domains: Art (A), Clipart (C), Product (P), and Real (R), with 65 categories. DomainNet is initially designed for benchmarking Multi-Source Domain Adaptation approaches. [21] pickup four domains: Real (R), Clipart (C), Painting (P), and Sketch (S) with 126 classes to build a cleaner dataset for SSDA. Besides, they focus on seven scenarios instead
of combining all pairs. Our experiments follow the settings in recent works [12,21,30], with the same sampling strategy for both the training set and validation set, and we conduct both 1-shot and 3-shot settings on all datasets.

**Implementation Details.** Our framework can be applied with many state-of-the-art methods, we choose MME [21], and CDAC [12] as our cooperators to validate the efficacy of our method, named MME + SLA and CDAC + SLA, respectively. For a fair comparison, we choose ResNet34 [7] as our backbone. The backbone is pre-trained on ImageNet-1K dataset [4], and the model architecture, batch size, learning rate scheduler, optimizer, weight-decay, and initialization strategy are all followed as previous works [12,21,30]. We follow the same hyper-parameters for MME and CDAC as their suggestions. We set the mix ratio $\alpha$ in Eq. 12 to 0.3 and the temperature parameter $T$ in Eq. 7 to 0.6. The update interval $I$ mentioned in Section 3.3 is 500. The warmup parameter $W$ in Eq. 12 is 5000 for CDAC on DomainNet. After the warmup stage, we refresh the learning rate scheduler so that the label adaptation loss can be updated with a higher learning rate. All hyper-parameters can be properly tuned via the validation process. For each subtask, we conducted the experiments three times. The detailed statistics of our results can be found in our supplementary materials.

### 4.2. Comparison with State-of-the-Arts

We compare our results with several baselines, including S+T, DANN [5], ENT [6], MME [21], APE [10], CDAC [12], DECOTA [31], MCL [30]. S+T is a baseline method for SSDA, with only source data and labeled target data involved in the training process. DANN is a classic unsupervised domain adaptation method, and [21] reproduces it by training with additional labeled target data. ENT is a standard entropy minimization originally designed for Semi-Supervised Learning, and the reproduction was also done by [21]. Note that for MCL, we only compare with their
Table 6. Accuracy (%) of MCL and MCL + SLA on Office-Home for 3-shot Semi Supervised Domain Adaptation (ResNet34).

<table>
<thead>
<tr>
<th>Method</th>
<th>A→C</th>
<th>A→P</th>
<th>A→R</th>
<th>C→A</th>
<th>C→P</th>
<th>C→R</th>
<th>P→A</th>
<th>P→C</th>
<th>P→R</th>
<th>R→A</th>
<th>R→C</th>
<th>R→P</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCL [30]</td>
<td>67.5</td>
<td>83.9</td>
<td>82.4</td>
<td>71.4</td>
<td>84.3</td>
<td>81.6</td>
<td>69.9</td>
<td>68.0</td>
<td>83.0</td>
<td>75.3</td>
<td>70.1</td>
<td>88.1</td>
<td>77.1</td>
</tr>
<tr>
<td>MCL*</td>
<td>64.1</td>
<td>81.6</td>
<td>80.6</td>
<td>70.3</td>
<td>82.2</td>
<td>79.2</td>
<td>70.6</td>
<td>64.0</td>
<td>81.8</td>
<td>75.3</td>
<td>67.8</td>
<td>86.6</td>
<td>75.3</td>
</tr>
<tr>
<td>MCL + SLA (ours)</td>
<td>64.3</td>
<td>81.6</td>
<td>80.8</td>
<td>70.2</td>
<td>82.6</td>
<td>79.4</td>
<td>70.9</td>
<td>64.2</td>
<td>82.2</td>
<td>75.5</td>
<td>68.0</td>
<td>86.8</td>
<td>75.6</td>
</tr>
</tbody>
</table>

*: Reproduced by ourselves

results on DomainNet. We leave the detailed analysis for MCL on Office-Home in Section 4.3.

**DomainNet.** We show the results on DomainNet dataset with 1-shot and 3-shot settings on Table 4. It is worth noting two things. First, for MME and CDAC, almost all sub-tasks get improvement after applying our SLA framework, except for only two cases where CDAC + SLA performs roughly the same as CDAC. Second, the overall performance of CDAC + SLA for 1-shot and 3-shot settings reaches 75.0% and 76.9%, respectively; both outperform the previous methods and achieve new state-of-the-art results.

**Office-Home.** We show the results on Office-Home dataset with 1-shot and 3-shot settings on Table 5. Similarly, after applying SLA to MME and CDAC, the performances get much better except for only one case under the 3-shot setting. Overall, our framework improves the original works by at least 1.5% under all settings.

4.3. Analysis

**Reproducibility issue for MCL.** MCL [30] designs consistency regularization for SSDA at three different levels and achieves excellent results. However, our experiments cannot fully reproduce their reported numbers. The reproduced results on 3-shot Office-Home dataset are shown in Table 6. After applying our SLA framework, although we can stably improve our reproduction, we are still unable to compete with their reported values. We put our detailed reproducing results into the supplementary materials, and the implementation is available at https://github.com/chester256/MCL.

**The intermediate results in SLA.** In SLA, we build a PPC to provide the view from the target data. PPC can be viewed as a variant of the pseudo-labeling method proposed in [13]. They apply such a method to boost their final performance in their work. If PPC has performed well, a natural question is: *Is it necessary to modify source labels by PPC?*

To reveal the intermediate steps within SLA, we plot the test performance of MME (red), MME + PPC (orange), MME + SLA (blue), and PPC within MME + SLA (purple) during the training phase in Figure 5. Initially, PPC (target view) performs at a higher level. However, if the source labels are not adapted, it will end up converging to the same performance as MME. In contrast, within our SLA framework, the model leverages the benefits of PPC, further producing an enhanced version of PPC, resulting in better overall performance compared to the original MME.

**Sensitivity study of $\alpha$.** Table 7 shows the sensitivity study results for $\alpha$ using MME + SLA on 3-shot OfficeHome dataset. We selected 0.3 from the best validation performance and kept 0.3 throughout all experiments for simplicity and resource saving. In the mid-range, $\alpha = 0.3$ (favoring source view) or $\alpha = 0.7$ (favoring target view) perform similarly, hinting that it is stable enough for a proper range of choices. It is worth noting that the dynamic adjustment of $\alpha$ is a promising direction. We proposed the warmup stage $\alpha_i = 0$ to the desired value after a period of warming up. More sophisticated scheduling techniques, such as the linear growth approach [9], could potentially replace the warmup parameter and lead to further improvements. We leave exploration of these techniques for future work.

**Illustration of the adapted labels.** We aim to adapt the original source label $y^*_i$ to the ideal label $g^*(x^*_i)$. To demonstrate the change of labels, we visualize the top 3 probabilities of the average adapted source labels on two classes in Figure 6. For Backpack (class 1), ideally-adapted
Figure 6. Top-3 probabilities of the average adapted source labels on 3-shot Office-Home $A \rightarrow C$ with ResNet34. **Top.** Class 1 (backpack). **Bottom.** Class 30 (knives).

Figure 7. Label Adaptation Loss of MME + SLA by first pre-training MME for $W$ iterations on 3-shot Office-Home $A \rightarrow C$ with ResNet34. (Smoothing by EMA with a ratio 0.8.)

S+T should change the source label to 40% Backpack + 10% Kettle + 8% Toys; SLA proposes to change to 30% Backpack + 5% Toys + 4% Kettle, which is closer to the ideally-adapted label than the original label of 100% Backpack. We draw the same conclusion for knives (class 30).

**Warmup issue for MME + SLA.** As described in Section 3.3, our framework relies on the quality of the predicted pseudo labels. Thus, we introduce a warmup stage parameter $W$ to derive a robust model. We can treat the warmup strategy as a two-stage algorithm. Take MME as our backbone method; the algorithm works like:

1. Train a model with normal MME loss for $W$ iteration.
2. Take the model above as a pre-trained model and further applying label adaptation loss.

For the first step, intuitively, we should train the model until the loss converges. That is how we select the warmup stage parameter for CDAC + SLA. However, empirically we found that the performance of MME + SLA will degrade if we train an MME model until it converges. Table 8 shows the sensitivity test of $W$ using MME + SLA on Office-Home dataset. We can observe that no matter the 1-shot or 3-shot settings, the performance is generally getting worse as the number of warmup stages increases. To analyze the effect, we first pre-train a normal MME for $W$ iterations, then observe the label adaptation loss of MME + SLA. Figure 7 plots the label adaptation loss of MME + SLA by first pre-training MME for $W$ iterations. We can observe that when $W = 5000$, the initial label adaptation loss has already been close to 0. Doing label adaptation in the situation is almost equivalent to not doing so, as we mentioned in Section 3.1.

**Limitation.** Our SLA framework might not be helpful if the label adaptation loss approaches 0. Although we have applied the Protonet with Pseudo Centers to avoid the issue, the loss will converge to 0 in MME + SLA. We leave the analysis of the reason for the convergence as a future work. On the other hand, we argue that it is unnecessary to discuss the reason in our proposed scope since we can make a trade-off by carefully tuning the warmup parameter $W$, and the issue turns out to be part of the hyper-parameters selection.

<table>
<thead>
<tr>
<th>$W$</th>
<th>1-shot</th>
<th>3-shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>62.09</td>
<td>65.90</td>
</tr>
<tr>
<td>1000</td>
<td>61.95</td>
<td>64.99</td>
</tr>
<tr>
<td>2000</td>
<td>61.37</td>
<td>64.72</td>
</tr>
<tr>
<td>3000</td>
<td>61.53</td>
<td>64.87</td>
</tr>
<tr>
<td>5000</td>
<td>61.79</td>
<td>64.68</td>
</tr>
</tbody>
</table>

Table 8. Accuracy (%) for different warmup stage $W$ of MME + SLA on Office-Home $A \rightarrow C$ with ResNet34.

5. Conclusion

In this work, we present a general framework, Source Label Adaptation for Semi-Supervised Domain Adaptation. Our work highlights that the usage of source data should be revisited carefully. We argue that the original source labels might be noisy from the perspective of target data. We approach Domain Adaptation as a Noisy Label Learning problem and correct source labels with the predictions from Protonet with Pseudo Centers. Our approach mainly addresses an issue that is orthogonal to other existing works, which focus on improving the usage of unlabeled data. The empirical results show that we can apply our framework to several state-of-the-art algorithms for SSDA and further boost their performances.

**Acknowledgement.** We thank the anonymous reviewers for valuable comments. This work is supported by the Ministry of Science and Technology in Taiwan via the grant MOST 111-2628-E-002-018. We thank to National Center for High-performance Computing (NCHC) in Taiwan for providing computational and storage resources.
References


