Low Category Uncertainty and High Training Potential Instance Learning for Unsupervised Domain Adaptation

Xinyu Zhang[1,2], **Meng Kang**[1,2], **Shuai Lü**[1,2,3*]

[1] Key Laboratory of Symbolic Computation and Knowledge Engineering (Jilin University), Ministry of Education, China
[2] College of Computer Science and Technology, Jilin University, China
[3] College of Software, Jilin University, China
lus@jlu.edu.cn, {zhang_xinyu22, kangmeng20}@mails.jlu.edu.cn

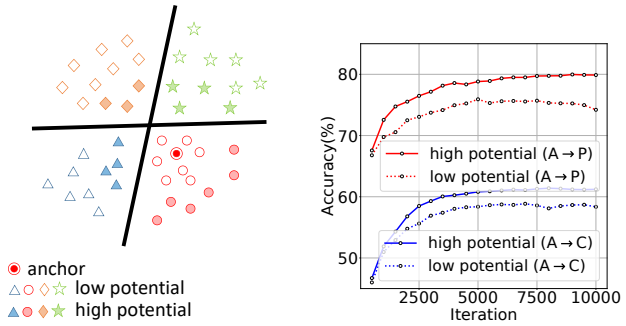
Abstract

Recently, instance contrastive learning achieves good results in unsupervised domain adaptation. It reduces the distances between positive samples and the anchor, increases the distances between negative samples and the anchor, and learns discriminative feature representations for target samples. However, most recent methods for identifying positive and negative samples are based on whether the pseudo-labels of samples and the pseudo-label of the anchor correspond to the same class. Due to the lack of target labels, many uncertain data are mistakenly labeled during the training process, and many low training potential data are also utilized. To address these problems, we propose Low Category Uncertainty and High Training Potential Instance Learning for Unsupervised Domain Adaptation (LUHP). We first propose a weight to measure the category uncertainty of the target sample. We can effectively filter the samples near the decision boundary through category uncertainty thresholds which are calculated by weights. Then we propose a new loss to focus on samples with high training potential. Finally, for anchors with low category uncertainty, we propose a sample reuse strategy to make the model more robust. We demonstrate the effectiveness of LUHP by showing the results of four datasets widely used in unsupervised domain adaptation.

Introduction

Unsupervised Domain Adaptation (UDA) applies knowledge or patterns learned from related fields or tasks to new unlabeled datasets. Some UDA methods (Long et al. 2015, 2017) attempt to learn a metric to quantify domain shift and minimize this metric to align source and target distributions. Recently, some works (Ganin et al. 2016; Hong et al. 2018; Tzeng et al. 2017) introduce the adversarial learning into UDA to implicitly align source and target distributions by learning domain invariant features.

The previous methods focus solely on learning domain agnostic (domain independent) features through global distribution alignment, which has the disadvantage of not considering the more refined class level structure of the target domain data (Sharma, Kalluri, and Chandraker 2021). To address this issue, the method based on instance contrastive learning in UDA is proposed. ILA-DA (Sharma, Kalluri, and Chandraker 2021) defines samples of the same class cross domains as positive pairs, and samples of different classes cross domains as negative pairs. CaCo (Huang et al. 2022) constructs a semantics-aware dictionary that includes source and target samples.



(a) Training Potential (b) Classification Accuracy

Figure 1: Low and High Training Potential Samples.

However, these methods only divide positive and negative samples based on whether the classes are the same, without considering the training potential of samples. As shown in Figure 1 (a), without contrastive training, the distribution of low training potential data in the feature space satisfies: samples with the same class as the anchor are close to the anchor, and samples with different classes from the anchor are far from the anchor. In Figure 1 (b), the model trained with high training potential samples performs better. And the red dashed line shows a decrease in accuracy of the model in the target domain after 5000 iterations. If we focus on low training potential samples, not only does the model fail to learn highly discriminative target features, but it also suffers from overfitting.

In addition, previous methods mainly obtain pseudo-labels by referring to source data (Sharma, Kalluri, and Chandraker 2021). Domain shift between source and target distributions during the initial training stage leads to noisy pseudo-labels which seriously affect the training of the model in the target data. Data with noisy labels have high uncertainty and they are distributed around the decision boundary (Zhang et al. 2021).

*Corresponding author

To address above issues, we propose Low Category Uncertainty and High Training Potential Instance Learning for UDA (LUHP). Firstly, there is a strong correlation between each sample and its neighbors. Inspired by ATDOC (Liang, Hu, and Feng 2021), we use the K-nearest neighbor strategy to identify the pseudo-labels of target samples. Then, in order to filter out data with high uncertainty, we design dynamical weights from the perspective of data distribution in feature embedding space. The category uncertainty weight represents the distance between the feature and the decision boundary. According to weights, we propose category uncertainty threshold to select reliable data as positive and negative samples for the anchor.

In addition, LUHP considers data with high training potential. Unlike previous work (Huang et al. 2022), both positive and negative samples of our method come from the target domain. For each anchor, we identify K positive and negative samples with high potential. Based on Triplet Loss (Schroff, Kalenichenko, and Philbin 2015), we design a new objective function to reduce the distances between the anchor and positive samples, and also to increase the distances between the anchor and negative samples.

Finally, for the current mini-batch, we select samples with high certainty based on the category uncertainty threshold for sample reuse to increase the robustness of the model. Due to LUHP focuses on each target anchor, it is an instance level method.

The main contributions of this paper are summarized as follows:

- We propose category uncertainty weight. By category uncertainty weight, we can obtain category uncertainty thresholds for different classes. According to thresholds, we can filter out the data distributed around the decision boundary and noisy data.
- For samples with low uncertainty, we design a reuse strategy to increase the robustness of the model.
- We design the multi-sample triplet loss based on weight, with the optimization goal of reducing the distances between the anchor and positive samples with high training potential, and also increasing the distances between the anchor and negative samples with high training potential.
- Extensive experiments show that our method achieves state-of-the-art (SOTA) on four well-known datasets.

Related Work

Unsupervised Domain Adaptation. Recently, UDA methods usually learn domain invariant features through adversarial learning. DANN (Ganin et al. 2016) utilizes the adversarial relationship between the domain discriminator and the feature extractor to learn domain invariant features. Methods (Saito et al. 2018; Li et al. 2021) propose a bi-classifier structure to replace the domain discriminator. DALN (Chen et al. 2022) introduces a discriminator free adversarial learning network for UDA. UTEP (Hu et al. 2022) models the uncertainty of domain discriminators. ELS (Zhang et al. 2023) encourages domain discriminators to output soft probabilities to reduce their confidences and mitigate the impact of noise labels.

UDA methods are usually combined with self-supervised learning (Liang, Hu, and Feng 2021) and mutual information (Zhao et al. 2022). MSGD (Xia, Jing, and Ding 2023) estimates and alleviates domain shift by introducing intermediate domains.

We focus on the finer class level distribution on the target domain and adopt same-class samples drawing and different-class samples separation strategy, which is similar to the purpose of instance contrastive learning.

Instance Contrastive Learning in UDA. Instance Contrastive Learning (Wu et al. 2018) is to learn an embedding space by pulling positive samples closer to the anchor and pushing negative samples farther away from the anchor. The application of instance contrastive learning in UDA is relatively rare, such as CaCo (Huang et al. 2022) and ILA-DA (Sharma, Kalluri, and Chandraker 2021).

CaCo considers instance contrastive learning as a category-aware dictionary look-up task. The category-aware dictionary has two characteristics: 1) The number of key values for each class is the same; 2) The key values include source and target data. CaCo proposes a category contrastive loss, with the goal of minimizing changes within classes and maximizing differences between classes.

ILA-DA proposes instance affinity relationships, which mainly define the category relationships between target and source data. If classes of source and target sample are consistent, the relationship coefficient is 1; Otherwise it is -1. ILA-DA defines samples with a correlation coefficient of 1 with the anchor as positive samples, and samples with a correlation coefficient of -1 with the anchor as negative samples. Finally, ILA-DA proposes Multi-Sample Contrastive Loss (MCS) (Sharma, Kalluri, and Chandraker 2021).

The above methods only identify the positive and negative samples of the anchor based on pseudo-labels, which has two problems: 1) Many samples with low training potential are considered, which brings negative effects on model training; 2) The pseudo-labels of target samples with high category uncertainty always vary, which is harmful for model training. Our method can solve these two problems. Firstly, our proposed multi-sample triplet loss based on weight focuses on high training potential samples. Secondly, we filter out noisy data and data with high uncertainty based on the category uncertainty weight.

Method

The setting in UDA contains two domains: source domain and target domain. We represent the source domain dataset as $\mathcal{D}_s = \{\mathcal{X}_s, \mathcal{Y}_s\} = \{(x_i^s, y_i^s)\}_{i=1}^{N_s}$, where N_s represents the number of source samples. We represent the target domain dataset without label information as $\mathcal{D}_t = \{\mathcal{X}_t\} = \{x_j^t\}_{j=1}^{N_t}$, where N_t represents the number of target samples. The source data and target data share the same label space, and we represent the number of classes as N_c.

The network model in this paper mainly consists of three parts: feature extractor F, domain discriminator D and classifier C. To provide pseudo-labels for target samples, we employ two memory banks: $\mathcal{M}_f$ is used to store target features and $\mathcal{M}_p$ is used to store target prediction probability

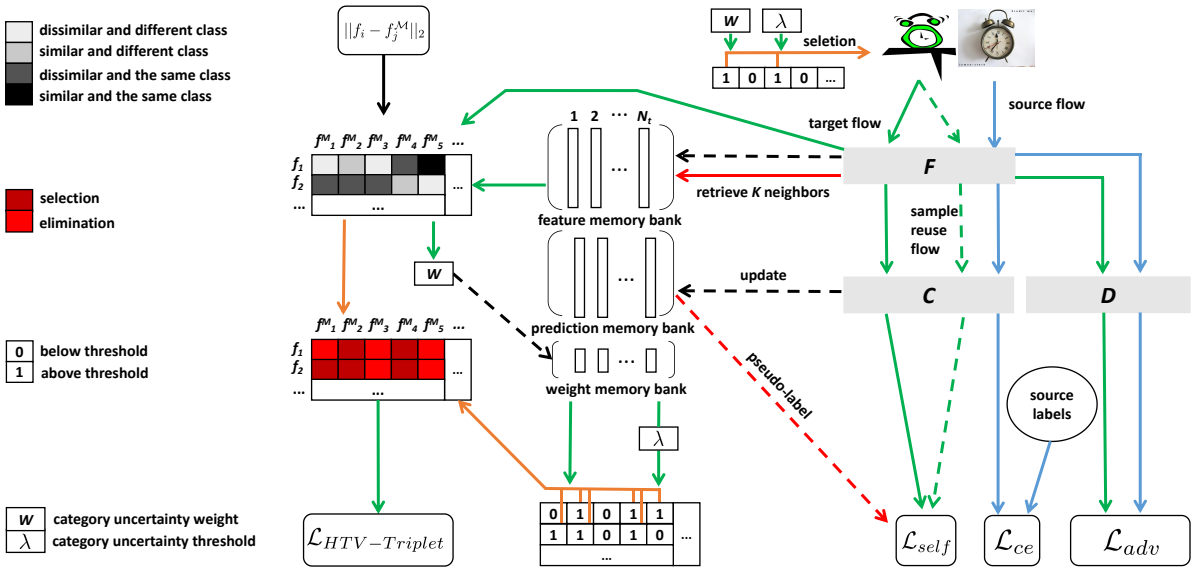


Figure 2: The Framework of LUHP.

vectors (Liang, Hu, and Feng 2021). In addition, since our method assigns a category uncertainty weight to each target sample, we define a memory bank $\mathcal{M}_w$ to store category uncertainty weights. We show the overview of LUHP in Figure 2. Firstly, we obtain pseudo-labels for target samples by K-nearest neighbor strategy. And we compute category uncertainty weights based on features in $\mathcal{M}_f$ and features of target samples in current mini-batch. We calculate category uncertainty thresholds based on category uncertainty weights and update $\mathcal{M}_w$ (**Category Uncertainty Weight**). Secondly, we select samples above thresholds with high training potential to compute the multi-sample triplet loss based on weight $\mathcal{L}_{HTV-Triplet}$ to complete to minimize changes within classes and maximize differences between classes (**Training with High Training Potential Samples**). Thirdly, we perform sample reuse strategy for samples in current mini-batch that are above thresholds (**Self-training and Sample Reuse**). Finally, We minimize the cross-entropy loss $\mathcal{L}_{ce}$ to complete the supervised learning of classifier C on source samples and complete domain alignment by the domain adversaria loss $\mathcal{L}_{adv}$ (**Adversarial Domain Adaption**).

Category Uncertainty Weight

We regard target samples in the mini-batch as anchors. Because target samples are completely unlabeled, it is difficult to identify the target data labels distributed around the decision boundary. We propose category uncertainty weights to distinguish samples distributed in high-density or low-density areas.

Because our method requires neighborhood information, we provide two memory banks $\mathcal{M}_f = \{f_1^{\mathcal{M}}, f_2^{\mathcal{M}}, ..., f_{N_t}^{\mathcal{M}}\}$ and $\mathcal{M}_p = \{p_1^{\mathcal{M}}, p_2^{\mathcal{M}}, ..., p_{N_t}^{\mathcal{M}}\}$ (Liang, Hu, and Feng 2021), where $\mathcal{M}_f$ stores all target features, while $\mathcal{M}_p$ stores all target prediction probability vectors. We use the Euclidean distance between features to retrieve the nearest neighbors of anchors.

Firstly, we use $\mathcal{N}_{k,i}^s$ to represent the the same class K-nearest neighbors of the i-th anchor, and use $\mathcal{N}_{k,i}^h$ to represent different classes K-nearest neighbors of the i-th anchor. Motivated by ATDOC (Liang, Hu, and Feng 2021), in order to provide reliable pseudo-label to the anchor, we adopt a K-nearest neighbor strategy. We directly use the mean vector of prediction probabilities (Liang, Hu, and Feng 2021) of K nearest neighbors as the pseudo prediction probability vector of the i-th anchor,

$$\hat{p}_i = \mathbb{E}_{x_j^t \in \mathcal{N}^{k,i}} p_j^{\mathcal{M}}, \tag{1}$$

where $\mathcal{N}^{k,i}$ represents K-nearest neighbors of x_i^t, $p_j^{\mathcal{M}}$ represents the prediction probability vector of the neighbor of the i-th target sample, and $\hat{p}_i$ represents the pseudo prediction probability vector of the i-th target sample. Therefore, we can directly obtain the pseudo-label of the i-th target sample,

$$\hat{y}_i = \arg\max_c \hat{p}_{i,c}. \tag{2}$$

Then, we define the category uncertainty weight of the i-th anchor as follow,

$$w_i = \frac{\sum_{x_j^t \in \mathcal{N}_{k,i}^h} ||f_i - f_j^{\mathcal{M}}||_2}{\sum_{x_j^t \in \mathcal{N}_{k,i}^s} ||f_i - f_j^{\mathcal{M}}||_2}, \tag{3}$$

where w_i represents the ratio of the sum of the distances between the i-th anchor and the nearest K samples of different classes to the sum of the distances between the i-th anchor and the nearest K samples of the same class. If a sample is distributed around the decision boundary, its nearest neighbors may contain samples with different labels $i.e.$ $w_i \approx 1$. When a sample is closer to the samples of other classes than the samples of the same class, it indicates that the sample is a noisy sample, that is, w_i is much less than 1. When the

sample is distributed around the center of the cluster, it is surrounded by samples with the same label as it, $i.e.\ w_i > 1$. We present a new memory bank:

$$\mathcal{M}_w = \{w_1^{\mathcal{M}}, w_2^{\mathcal{M}}, ..., w_{N_t}^{\mathcal{M}}\}, \tag{4}$$

where $\mathcal{M}_w$ is used to store category uncertainty weights of target samples. We update the $\mathcal{M}_w$ every iteration. Specifically, we update weights of anchors in the current mini-batch in $\mathcal{M}_w$. Category uncertainty weights have two functions: 1) calculate category uncertainty thresholds; 2) select samples for sample reuse.

Finally, we provide the calculation of thresholds. We need to obtain the pseudo-label for each target sample in $\mathcal{M}_p$:

$$\hat{y}_i^{\mathcal{M}} = \arg\max_c p_{i,c}^{\mathcal{M}}, i = 1, 2, ..., N_t. \tag{5}$$

After obtaining pseudo-labels, we can calculate the c-th category uncertainty thresholds:

$$\begin{aligned} \lambda_c^+ &= \mathbb{E}_{\hat{y}_i^{\mathcal{M}}=c} w_i^{\mathcal{M}}, \\ \lambda_c^- &= \mathbb{E}_{\hat{y}_i^{\mathcal{M}}\neq c} w_i^{\mathcal{M}}, i = 1, 2, ..., N_t. \end{aligned} \tag{6}$$

Training with High Training Potential Samples

In order to achieve clustering of target samples at the class level in the feature space, we regard target samples in the mini-batch as the anchors, and pull the anchor closer to high training potential samples in the $\mathcal{M}_f$ of the same class, and push the anchor farther away from high training potential samples in the $\mathcal{M}_f$ of different classes. Specifically, after obtaining category certainty thresholds, we select the thresholds for the corresponding class of the anchor based on its pseudo-label. If the weight of a sample in the $\mathcal{M}_f$ is greater than thresholds, we select it. So far, we can obtain samples with high category certainty, that is, samples far away from the decision boundary. Then, we represent the feature-label pair set $S_{t,i}^h$ of the i-th anchor. It contains high category certainty samples,

$$\begin{aligned} S_{t,i}^h = &\{(f_j^{\mathcal{M}}, \hat{y}_j^{\mathcal{M}})\},\ \ s.t.\ \ (\hat{y}_j^{\mathcal{M}} = \hat{y}_i \wedge w_j^{\mathcal{M}} \geq \lambda_{\hat{y}_i}^+) \\ &\vee (\hat{y}_j^{\mathcal{M}} \neq \hat{y}_i \wedge w_j^{\mathcal{M}} \geq \lambda_{\hat{y}_i}^-), j = 1, 2, ..., N_t. \end{aligned} \tag{7}$$

We regard samples in $S_{t,i}^h$ with the same class as the i-th anchor as positive sample set, and samples in $S_{t,i}^h$ with different classes as negative sample set. Then we arrange the positive sample set in descending order of distances from features of samples to the feature of the anchor, and the negative sample set in ascending order of distances from features of samples to the feature of the anchor. We take the first K elements from the positive and negative sample sets respectively, which represent samples with high training potential. Finally, we represent the high training potential samples of the i-th anchor as $\mathcal{N}_{K,i}^+$ and $\mathcal{N}_{K,i}^-$. Based on Triplet Loss (Schroff, Kalenichenko, and Philbin 2015), we first propose the i-th single-sample based $\mathcal{L}_{sHTV-Triplet}$,

$$\begin{aligned} \mathcal{L}_{sHTV-Triplet,i} = \sum_{x_j^t \in \mathcal{N}_{K,i}^+, x_k^t \in \mathcal{N}_{K,i}^-} &\max(||f_i - f_j^{\mathcal{M}}||_2 \\ &- ||f_i - f_k^{\mathcal{M}}||_2 + margin, 0). \end{aligned} \tag{8}$$

For all the experiments in this paper, $margin = 0.3$ (Schroff, Kalenichenko, and Philbin 2015).

Then, we define the multi-sample triplet loss based on weight,

$$\mathcal{L}_{HTV-Triplet} = \sum_{i=1}^{B} \hat{p}_{i,\hat{y}_i} \cdot \mathcal{L}_{sHTV-Triplet,i}, \tag{9}$$

where B represents the batch size of anchors. And we consider the component of the prediction probability vector corresponding to $\hat{y}_i$. If this component value is relatively large, we believe that $\hat{y}_i$ has a high confidence level; Otherwise, the confidence level is low. The low confidence level may reflect the unreliable pseudo-labels generated by K-nearest neighbor strategy. So we assgin a low weight to $\mathcal{L}_{sHTV-Triplet,i}$. Finally, we achieve a compact distribution of target samples in the same class and separate target samples from different classes by minimizing the $\mathcal{L}_{HTV-Triplet}$.

Self-training and Sample Reuse

For learning local neighborhood knowledge of target samples, similar to $\mathcal{L}_{HTV-Triplet}$, we consider using the maximum component of the pseudo prediction probability vector as the confidence weight. We consider a weighted cross-entropy loss (Liang, Hu, and Feng 2021),

$$\mathcal{L}_{KNN} = -\frac{1}{B} \sum_{i=1}^{B} \hat{p}_{i,\hat{y}_i} \cdot \log C(F(x_i^t))_{\hat{y}_i}. \tag{10}$$

According to Eq. 3 and Eq. 6, we can calculate the category uncertainty weight w_i of the i-th target sample and category uncertainty thresholds. Then, we can select samples with low category uncertainty from the B target samples in current iteration for sample reuse,

$$S_t^{lu} = \{(x_i^t, \hat{y}_i)\},\ \ s.t.\ \ \hat{y}_i = c \wedge w_i \geq \lambda_c^+, i = 1, 2, ..., B. \tag{11}$$

Utilizing the Mixup (Zhang et al. 2018) strategy, we randomly mix two target samples x_i^t and x_j^t in S_t^{lu}, as well as their pseudo-labels $\hat{y}_i$ and $\hat{y}_j$,

$$\begin{aligned} x_i^{new} &= \gamma \cdot x_i^t + (1 - \gamma) \cdot x_j^t, \\ \hat{y}_i^{new} &= \gamma \cdot \hat{y}_i + (1 - \gamma) \cdot \hat{y}_j, \end{aligned} \tag{12}$$

where $\gamma \sim Beta(\epsilon, \epsilon)$, we set $\epsilon = 1.0$ (Zhang et al. 2018). We obtain a set $S^{new} = \{(x_i^{new}, \hat{y}_j^{new})\}_{i=1}^{N^{new}}$. $N^{new} = \mid S_t^{lu} \mid = \mid S^{new} \mid$. Then, we input samples in S^{new} into the feature extractor and classifier to obtain the prediction vectors. Finally, we directly adopt the cross-entropy loss,

$$\mathcal{L}_{reuse} = -\frac{1}{N^{new}} \sum_{i=1}^{N^{new}} \hat{y}_i^{new} \cdot \log(C(F(x_i^{new}))). \tag{13}$$

$\mathcal{L}_{KNN}$ and $\mathcal{L}_{reuse}$ can be considered as self-supervised losses, so we propose a new self-supervised loss,

$$\mathcal{L}_{self} = \mathcal{L}_{reuse} + \mathcal{L}_{KNN}. \tag{14}$$

Finally, update strategy for $\mathcal{M}_f$ and $\mathcal{M}_p$ is similar to the update strategy of $\mathcal{M}_w$. We use target features and target probability predictions from the current mini-batch to update features and probability predictions stored in $\mathcal{M}_f$ and $\mathcal{M}_p$ corresponding to target samples in the current mini-batch.

Adversarial Domain Adaption

Since the source data is richly labeled, we directly train the classifier C through a cross-entropy loss,

$$\mathcal{L}_{ce} = \mathbb{E}_{(x_i^s, y_i^s)\sim D^s}[-y_i^s \cdot \log(C\left(F\left(x_i^s\right)\right))]. \tag{15}$$

However, due to domain shift between the source and target domains, directly deploying the training model on the source domain to the target domain is not effective. The domain discriminator D (Ganin et al. 2016) can solve this problem,

$$\begin{aligned}\mathcal{L}_{adv} = &- \mathbb{E}_{x_i^s\sim D^s}[\log D(F(x_i^s))] - \\ &\mathbb{E}_{x_i^t\sim D^t}[\log(1 - D\left(F\left(x_i^t\right)\right))].\end{aligned} \tag{16}$$

Overall Loss

We give the overall loss function of LUHP,

$$\mathcal{L} = \mathcal{L}_{ce} + \mathcal{L}_{adv} + \alpha \cdot \eta\mathcal{L}_{self} + \eta\mathcal{L}_{HTV-Triplet}, \tag{17}$$

where α represents the trade-off coefficient of $\mathcal{L}_{self}$, and η represents a linearity coefficient (Liang, Hu, and Feng 2021) that increases from 0 to 1 from the beginning to the end of training. This paper mainly proposes two losses, $\mathcal{L}_{self}$ and $\mathcal{L}_{HTV-Triplet}$. $\mathcal{L}_{self}$ consists of two parts, $\mathcal{L}_{reuse}$ and $\mathcal{L}_{KNN}$. By minimizing $\mathcal{L}_{KNN}$, we complete the learning of local neighborhood knowledge for the target sample; By minimizing $\mathcal{L}_{reuse}$, we achieve reuse of low category uncertain target samples, thereby increasing the robustness of the model; By minimizing $\mathcal{L}_{HTV-Triplet}$, we complete contrastive training between the anchor and high training potential samples. For better understanding, we present the overall procedure of LUHP in Algorithm 1.

Algorithm 1: LUHP

Input: Source samples $\{(x_i^s, y_i^s)\}_{i=1}^{N_s}$; Target samples $\{x_i^t\}_{i=1}^{N_t}$; Trade-off parameter α; Linearity coefficient η; Number of neighbors K; Batch size B; max_iteration.

Output: Optimal parameters θ_F, θ_C of feature extractor F and classifier C.

Randomly initialize θ_C, randomly initialize each feature vector and prediction vector in $\mathcal{M}_f$ and $\mathcal{M}_p$, initialize each category uncertainty weight in $\mathcal{M}_w$ with 1. Initialize θ_f using pre-trained model on ImageNet.
for $epoch = 1$ to max_iteration **do**
  Randomly sample a mini-batch of B source samples and B target samples.
  Obtain pseudo-labels $\hat{p}$ of the target samples by Eq. 1, calculate category uncertainty thresholds $\lambda = [\lambda_1^{\pm}, \lambda_2^{\pm}, ..., \lambda_{N_c}^{\pm}]$, obtain the set $S_t^h = [S_{t,1}^h, S_{t,2}^h, ..., S_{t,B}^h]$ containing sample features with low category uncertainty in $\mathcal{M}_f$ and labels by Eq. 7.
  Calculate the single-sample loss $\mathcal{L}_{sHTV-Triplet}$ by Eq. 8, then calculate the multi-sample triplet loss based on weight $\mathcal{L}_{HTV-Triplet}$ by Eq. 9.
  Obtain the set S_t^{lu} containing target samples with low category uncertainty and pseudo-labels by Eq. 11, generate new samples x^{new} and labels $\hat{y}^{new}$ using samples and labels in S_t^{lu} by Eq. 12 for sample reuse.
  Calculate $\mathcal{L}_{KNN}$, $\mathcal{L}_{reuse}$ by Eq. 10 and Eq. 13.
  Calculate $\mathcal{L}_{ce}$ by Eq. 15 and calculate $\mathcal{L}_{adv}$ by Eq. 16.
  Update θ_F, θ_C by minimizing $\mathcal{L}$.
  Update $\mathcal{M}_f$, $\mathcal{M}_p$ and $\mathcal{M}_w$.
end for

Experiment

Dataset and Implementation Details. We evaluate our method on Office-31 (Saenko et al. 2010), Office-Home (Venkateswara et al. 2017), VisDA-2017 (Peng et al. 2017) and DomainNet (Peng et al. 2019). For DomainNet, We adopt the settings in BIWAA-I (Westfechtel et al. 2023) and KUDA (Sun, Lu, and Ling 2022). Specifically, we select 40 common classes in the original dataset from 4 domains. We use ResNet (He et al. 2016) pre-trained on ImageNet as the F and use one fully connected layer as C. We set α = 0.2 for Office-31, and set α = 0.5 for other benchmarks. The value of K is 5. Other implementation details are shown in Appendix. Code is available at https://github.com/zxyzxyhh/LUHP.

Comparison Performance

We present the results of our method and other baselines on four benchmarks in Tables 1-4. The bold font in each column represents the best accuracy on the corresponding task.

Results on Office-31. Our method has strong competitiveness on Office-31. The average accuracy of LUHP is close to optimal, only 0.1% lower than the result of MSGD. Our method has outstanding performance on difficult tasks W→A and D→A, due to the filtering of early uncertain target samples by category uncertainty thresholds and the training of high training potential samples. Specifically, similar to the setting in ATDOC (Liang, Hu, and Feng 2021), we calculate the average accuracy Avg.† for four tasks: A→W, A→D, D→A, and W→A, and LUHP achieves the best result.

Results on Office-Home. The results of our method on each task on this dataset are the best. Instance level contrastive learning based samples with high training potential helps model to learn the class level distribution of target features. After obtaining the target distribution, we align source and target distributions at the class level. Compared to SOTA, the average accuracy of our method on Office-Home improves by 2.8%.

Results on VisDA-2017. The average accuracy of our method is 84.6%. Compared to the instance level contrastive learning method CaCo, the results of our method improves by 3%. Unlike CaCo, our proposed $\mathcal{L}_{HTV-Triplet}$ only focuses on samples with high training potential, so the trained model performs better on the target domain.

Results on DomainNet. The baselines mainly come from BIWAA-I (Westfechtel et al. 2023) and COAL (Tan, Peng, and Saenko 2020). The results of our method exceeded SOTA. For 12 tasks on DomainNet, our method works best on 6 tasks. Our breakthrough on DomainNet is mainly due to three parts of LUHP: Firstly, we use category uncertainty

Method	A→W	D→W	W→D	A→D	D→A	W→A	Avg.	Avg.†
ResNet (He et al. 2016)	68.4	96.7	99.3	68.9	62.5	60.7	76.1	65.1
DANN (Ganin et al. 2016)	82.0	96.9	99.1	79.7	68.2	67.4	82.2	74.3
BNM (Cui et al. 2020)	91.5	98.5	**100.0**	90.3	70.9	71.6	87.1	81.1
BCDM (Li et al. 2021)	95.4	98.6	**100.0**	93.8	73.1	73.0	89.0	83.8
ATDOC (Liang et al. 2021)	94.3	98.9	**100.0**	94.4	75.6	75.2	89.7	84.9
ILA-DA (Sharma et al. 2021)	**95.7**	**99.3**	**100.0**	93.4	72.1	75.4	89.3	84.2
CaCo (Huang et al. 2022)	90.4	98.9	**100.0**	92.8	73.7	72.5	88.1	82.4
DALN (Chen et al. 2022)	95.2	99.1	**100.0**	95.4	76.4	76.5	90.4	85.9
UTEP (Hu et al. 2022)	94.7	99.0	**100.0**	94.4	77.0	74.5	89.9	85.2
ELS (Zhang et al. 2023)	93.6	99.0	**100.0**	93.4	**78.7**	77.5	90.4	85.8
BIWAA-I (Westfechtel et al. 2023)	95.6	99.0	**100.0**	94.4	75.9	77.3	90.5	85.8
MSGD (Xia et al. 2023)	95.5	99.2	**100.0**	**95.6**	77.3	77.0	**90.8**	**86.4**
LUHP (Ours)	94.2	98.6	**100.0**	95.2	77.7	**78.6**	90.7	**86.4**

Table 1: Accuracy (%) on Office-31 for UDA (ResNet-50).

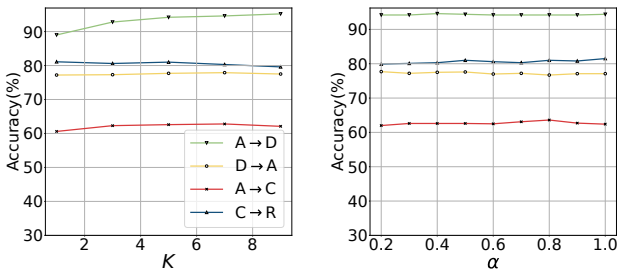
Method	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	Avg.
ResNet (He et al. 2016)	34.9	50.0	58.0	37.4	41.9	46.2	38,5	31.2	60.4	53.9	41.2	59.9	46.1
DANN (Ganin et al. 2016)	45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	63.2	51.8	76.8	57.6
BNM (Cui et al. 2020)	56.2	73.7	79.0	63.1	73.6	74.0	62.4	54.8	80.7	72.4	58.9	83.5	69.4
BCDM (Li et al. 2021)	52.5	73.1	77.2	64.4	69.7	71.8	60.4	50.7	77.2	72.4	57.8	81.2	67.4
ATDOC (Liang et al. 2021)	58.3	78.8	82.3	69.4	78.2	78.2	67.1	56.0	82.7	72.0	58.2	85.5	72.2
DALN (Chen et al. 2022)	57.8	79.9	82.0	66.3	76.2	77.2	66.7	55.5	81.3	73.5	60.4	85.3	71.8
UTEP (Hu et al. 2022)	57.2	75.9	79.6	63.4	72.8	73.7	64.6	55.4	79.8	74.0	61.1	84.2	70.1
ELS (Zhang et al. 2023)	58.2	79.7	82.5	67.5	77.2	77.2	64.6	57.9	82.2	75.4	63.1	85.5	72.6
BIWAA-I (Westfechtel et al. 2023)	56.3	78.4	81.2	68.0	74.5	75.7	67.9	56.1	81.2	75.2	60.1	83.8	71.5
MSGD (Xia et al. 2023)	58.7	76.9	78.9	70,1	76.2	76.6	69.0	57.2	82.3	74.9	62.7	84.5	72.4
LUHP (Ours)	**63.0**	**80.2**	**83.9**	**72.5**	**81.7**	**81.0**	**70.5**	**60.7**	**84.0**	**75.9**	**65.2**	**85.6**	**75.4**

Table 2: Accuracy (%) on Office-Home for UDA (ResNet-50).

thresholds to filter out uncertain samples in order to prevent the impact of noise on model training; Secondly, the proposed $\mathcal{L}_{HTV-Triplet}$ helps the model focus on high training potential samples to achieve clustering of target data; Finally, for target samples with low class uncertainty, we adopt a reuse strategy, which helps to improve the robustness of the model in the target domain.

Ablation Study and Analysis

Parameter Sensitivity. For parameter sensitivity testing, We take D→A and A→D tasks in Office-31, A→C and C→R tasks on Office-Home. In Figure 3 (a), we evaluate the first hyperparameter K of our method. K firstly represents the number of neighbors in the K-nearest neighbor strategy of the target sample. Secondly, K is used for us to obtain category uncertainty weights. Finally, for $\mathcal{L}_{HTV-Triplet}$, K represents the number of positive and negative samples of the anchor. If the K is very small, the quality of the model is easily influenced by individual samples. If the K is very large, samples with noisy labels can likely be considered. When the K is 5, we can find that the model performs well on most tasks. So we set the K to 5. In Figure 3 (b), we evaluated the second hyperparameter α, which is responsible for balancing the importance of $\mathcal{L}_{HTV-Triplet}$. We set the value of α to 0.2 to 1, and we can find that the method is stable and the model is not sensitive to α.



(a) K (b) α

Figure 3: Parameter Sensitivity.

Ablation study of different components. In Table 5, we use $\mathcal{L}_K$, $\mathcal{L}_T$, $weight_c$ and $\mathcal{L}_r$ to represent $\mathcal{L}_{KNN}$, $\mathcal{L}_{HTV-Triplet}$, category uncertainty weight and $\mathcal{L}_{reuse}$. We conduct ablation experiments on Office-Home and DomainNet by isolating different components. $\mathcal{L}_{HTV-Triplet}$ can achieve the anchor close to high training potential samples of the same class, and push away high training potential samples of different classes, which helps the model learn discriminative features. The results in rows 6 and 7 of the Table 5 demonstrate the effectiveness of $\mathcal{L}_{HTV-Triplet}$. Cate-

Method	plane	bcycl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	trunk	Avg.
ResNet (He et al. 2016)	55.1	53.3	61.9	59.1	80.6	17.9	79.9	31.2	81.0	26.5	73.5	8.5	52.4
DANN (Ganin et al. 2016)	81.9	77.7	82.8	44.3	81.2	29.5	65.1	28.6	51.9	54.6	82.8	7.8	57.4
BNM (Cui et al. 2020)	89.6	61.5	76.9	55.0	89.3	69.1	81.3	65.5	90.0	47.3	**89.1**	30.1	70.4
BCDM (Li et al. 2021)	95.1	**87.6**	81.2	**73.2**	92.7	**95.4**	86.9	82.5	95.1	84.8	88.1	39.5	83.4
ATDOC (Liang et al. 2021)	93.7	83.0	76.9	58.7	89.7	95.1	84.4	71.4	89.4	80.0	86.7	55.1	80.3
CaCo (Huang et al. 2022)	91.4	80.6	80.0	56.5	89.5	89.4	82.8	79.9	88.8	86.8	87.3	**66.0**	81.6
DALN (Chen et al. 2022)	96.0	86.3	74.3	50.0	92.4	94.7	83.5	76.4	91.0	87.2	88.4	47.4	80.6
UTEP (Hu et al. 2022)	94.7	75.4	83.2	60.1	93.7	95.3	**93.1**	82.6	94.3	89.8	84.6	41.1	82.3
MSGD (Xia et al. 2023)	**97.5**	83.4	**84.4**	69.4	**95.9**	94.1	90.9	75.5	**95.5**	**94.6**	88.1	44.9	84.5
LUHP (Ours)	94.4	84.0	78.0	61.8	94.8	95.2	87.6	**82.7**	92.0	93.3	87.3	63.5	**84.6**

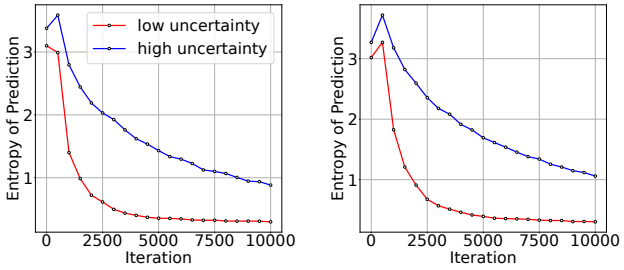
Table 3: Accuracy(%) on VisDA-2017 for UDA (ResNet-101).

Method	R→C	R→P	R→S	C→R	C→P	C→S	P→R	P→C	P→S	S→R	S→C	S→P	Avg.
ResNet (He et al. 2016)	58.8	67.9	53.1	76.7	53.6	53.0	84.4	55.6	60.2	74.6	54.6	57.8	62.5
ETN (Cao et al. 2019)	69.2	72.1	63.6	86.5	65.3	63.3	85.0	65.7	68.8	84.9	72.2	69.0	74.0
BSP (Chen et al. 2019)	67.3	73.5	69.3	86.5	67.5	70.9	86.8	70.3	68.8	84.3	72.4	71.5	74.1
DANN (Ganin et al. 2016)	63.4	73.6	72.6	86.5	65.7	70.6	86.9	73.2	70.2	85.7	75.2	70.0	74.5
COAL (Tan et al. 2020)	73.9	75.4	70.5	89.6	70.0	71.3	89.8	68.0	70.5	88.0	73.2	70.5	75.9
InstaPBM (Li et al. 2020)	80.1	75.9	70.8	89.7	70.2	72.8	89.6	74.4	72.2	87.0	79.7	71.8	77.8
BIWAA-I (Westfechtel et al. 2023)	79.9	75.2	75.4	87.9	72.1	75.7	88.9	77.8	**76.7**	88.8	**80.5**	74.5	79.4
KUDA (Sun et al. 2022)	**83.6**	77.5	75.3	**91.5**	76.4	**77.0**	**91.7**	**82.3**	76.3	89.7	80.2	70.3	81.0
LUHP (Ours)	79.6	**82.8**	**79.3**	91.1	**79.7**	76.5	90.2	77.2	**76.7**	**91.2**	80.3	**79.5**	**82.0**

Table 4: Accuracy(%) on DomainNet for UDA (ResNet-50).

$\mathcal{L}_K$	$\mathcal{L}_T$	$weight_c$	$\mathcal{L}_r$	Office-Home	DomainNet
×	×	×	×	57.6	74.5
✓	×	×	×	73.7	80.3
✓	✓	×	×	74.1	80.3
✓	✓	✓	×	74.4	81.1
✓	✓	×	✓	74.0	81.1
✓	×	✓	✓	74.7	81.1
✓	✓	✓	✓	**75.4**	**82.0**

Table 5: Ablation Study on LUHP.



(a) A→C (b) A→P

Figure 4: Entropy of Prediction Probability.

gory uncertainty weights allow us to focus on the target samples far away from the decision boundary. By results in rows 5 and 7 of the Table 5, we observe that not considering category uncertainty weights can introduce samples with high uncertainty which have a significant negative impact on the model. Then, we find that the sample reuse strategy brings great improvement to the robustness of the model from results in rows 4 and 7 of the Table 5.

Effectiveness of Category Uncertainty Weights. In Figure 4, we show the results on A→C and A→P tasks in Office-Home. Specifically, we split target samples for each iteration into two parts based on category uncertainty thresholds: high category uncertainty samples and low category certainty samples. Then, we compare the means of the entropy of predictions of two kinds of samples. Samples with high category uncertainty have high entropy. Samples with high entropy are considered to be distributed around the decision boundary (noisy data). Results in Figure 4 indicate that by category uncertainty weights we are able to assign high weights to samples far from the decision boundary.

Conclusion

In this paper, we propose LUHP. Firstly, we consider the K-nearest neighbor strategy to provide pseudo-labels for target samples. Then, we propose category uncertainty weights to calculate category uncertainty thresholds. According to thresholds, we can remove the uncertain samples around the decision boundary. In addition, we propose the multi-sample triplet loss based on weight for generating class level discriminative target features. Finally, we utilize target samples with low category uncertainty for a reuse strategy to improve the robustness of the model.

Acknowledgments

We sincerely thank the anonymous reviewers for their careful work and thoughtful suggestions, which have greatly improved this article. This work was supported by the Natural Science Research Foundation of Jilin Province of China under Grant Nos. YDZJ202201ZYTS423 and 20220101106JC.

References

Cao, Z.; You, K.; Long, M.; Wang, J.; and Yang, Q. 2019. Learning to Transfer Examples for Partial Domain Adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2985–2994.

Chen, L.; Chen, H.; Wei, Z.; Jin, X.; Tan, X.; Jin, Y.; and Chen, E. 2022. Reusing the Task-specific Classifier as a Discriminator: Discriminator-free Adversarial Domain Adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 7171–7180.

Chen, X.; Wang, S.; Long, M.; and Wang, J. 2019. Transferability vs. Discriminability: Batch Spectral Penalization for Adversarial Domain Adaptation. In *International Conference on Machine Learning*, 1081–1090.

Cui, S.; Wang, S.; Zhuo, J.; Li, L.; Huang, Q.; and Tian, Q. 2020. Towards Discriminability and Diversity: Batch Nuclear-Norm Maximization Under Label Insufficient Situations. In *IEEE Conference on Computer Vision and Pattern Recognition*, 3940–3949.

Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; and Lempitsky, V. S. 2016. Domain-Adversarial Training of Neural Networks. *Journal of Machine Learning Research*, 17: 59:1–59:35.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Hong, W.; Wang, Z.; Yang, M.; and Yuan, J. 2018. Conditional Generative Adversarial Network for Structured Domain Adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1335–1344.

Hu, J.; Zhong, H.; Yang, F.; Gong, S.; Wu, G.; and Yan, J. 2022. Learning Unbiased Transferability for Domain Adaptation by Uncertainty Modeling. In *European Conference on Computer Vision*, 223–241.

Huang, J.; Guan, D.; Xiao, A.; Lu, S.; and Shao, L. 2022. Category Contrast for Unsupervised Domain Adaptation in Visual Tasks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1193–1204.

Li, B.; Wang, Y.; Che, T.; Zhang, S.; Zhao, S.; Xu, P.; Zhou, W.; Bengio, Y.; and Keutzer, K. 2020. Rethinking Distributional Matching Based Domain Adaptation. *arXiv preprint arXiv*, 2006.13352.

Li, S.; Lv, F.; Xie, B.; Liu, C. H.; Liang, J.; and Qin, C. 2021. Bi-Classifier Determinacy Maximization for Unsupervised Domain Adaptation. In *AAAI Conference on Artificial Intelligence*, 8455–8464.

Liang, J.; Hu, D.; and Feng, J. 2021. Domain Adaptation With Auxiliary Target Domain-Oriented Classifier. In *IEEE Conference on Computer Vision and Pattern Recognition*, 16632–16642.

Long, M.; Cao, Y.; Wang, J.; and Jordan, M. I. 2015. Learning Transferable Features with Deep Adaptation Networks. In *International Conference on Machine Learning*, 97–105.

Long, M.; Zhu, H.; Wang, J.; and Jordan, M. I. 2017. Deep Transfer Learning with Joint Adaptation Networks. In *International Conference on Machine Learning*, 2208–2217.

Peng, X.; Bai, Q.; Xia, X.; Huang, Z.; Saenko, K.; and Wang, B. 2019. Moment Matching for Multi-Source Domain Adaptation. In *IEEE International Conference on Computer Vision*, 1406–1415.

Peng, X.; Usman, B.; Kaushik, N.; Hoffman, J.; Wang, D.; and Saenko, K. 2017. VisDA: The Visual Domain Adaptation Challenge. *arXiv preprint arXiv*, 1710.06924.

Saenko, K.; Kulis, B.; Fritz, M.; and Darrell, T. 2010. Adapting Visual Category Models to New Domains. In *European Conference on Computer Vision*, 213–226.

Saito, K.; Watanabe, K.; Ushiku, Y.; and Harada, T. 2018. Maximum Classifier Discrepancy for Unsupervised Domain Adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 3723–3732.

Schroff, F.; Kalenichenko, D.; and Philbin, J. 2015. FaceNet: A Unified Embedding for Face Recognition and Clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, 815–823.

Sharma, A.; Kalluri, T.; and Chandraker, M. 2021. Instance Level Affinity-Based Transfer for Unsupervised Domain Adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 5361–5371.

Sun, T.; Lu, C.; and Ling, H. 2022. Prior Knowledge Guided Unsupervised Domain Adaptation. In *European Conference on Computer Vision*, 639–655.

Tan, S.; Peng, X.; and Saenko, K. 2020. Class-Imbalanced Domain Adaptation: An Empirical Odyssey. In *European Conference on Computer Vision*, 585–602.

Tzeng, E.; Hoffman, J.; Saenko, K.; and Darrell, T. 2017. Adversarial Discriminative Domain Adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2962–2971.

Venkateswara, H.; Eusebio, J.; Chakraborty, S.; and Panchanathan, S. 2017. Deep Hashing Network for Unsupervised Domain Adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 5385–5394.

Westfechtel, T.; Yeh, H.; Meng, Q.; Mukuta, Y.; and Harada, T. 2023. Backprop Induced Feature Weighting for Adversarial Domain Adaptation with Iterative Label Distribution Alignment. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, 392–401.

Wu, Z.; Xiong, Y.; Yu, S. X.; and Lin, D. 2018. Unsupervised Feature Learning via Non-Parametric Instance Discrimination. In *IEEE Conference on Computer Vision and Pattern Recognition*, 3733–3742.

Xia, H.; Jing, T.; and Ding, Z. 2023. Maximum Structural Generation Discrepancy for Unsupervised Domain Adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3): 3434–3445.

Zhang, B.; Wang, Y.; Hou, W.; Wu, H.; Wang, J.; Okumura, M.; and Shinozaki, T. 2021. FlexMatch: Boosting Semi-Supervised Learning with Curriculum Pseudo Labeling. In *Annual Conference on Neural Information Processing Systems*, 18408–18419.

Zhang, H.; Cissé, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2018. mixup: Beyond Empirical Risk Minimization. In *International Conference on Learning Representations*.

Zhang, Y.; Wang, X.; Liang, J.; Zhang, Z.; Wang, L.; Jin, R.; and Tan, T. 2023. Free Lunch for Domain Adversarial Training: Environment Label Smoothing. In *International Conference on Learning Representations*.

Zhao, H.; Ma, C.; Chen, Q.; and Deng, Z. 2022. Domain Adaptation via Maximizing Surrogate Mutual Information. In *International Joint Conference on Artificial Intelligence*, 1700–1706.