SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

April 2023, Vol. 66 142101:1–142101:14
https://doi.org/10.1007/s11432-021-3535-2

Enhancing unsupervised domain adaptation by exploiting the conceptual consistency of multiple self-supervised tasks

Hui SUN & Ming LI*

National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, *China*

Received 19 October 2021/Revised 13 December 2021/Accepted 16 April 2022/Published online 21 February 2023

Abstract Unsupervised domain adaptation (UDA) aims to transfer the knowledge from a label-rich source domain to an unlabeled target domain. Current approaches mainly focus on aligning the target domain's data distribution with the source domain, but lacking attention to allowing target data to guide which features need to be captured. Source data dominates feature extraction. As a result, target embedding would lose some vital discriminative features and limit UDA in complex tasks. In this paper, we argue that utilizing auxiliary tasks to capture target-intrinsic patterns, which do not depend on source supervision, could further enhance UDA. Multiple auxiliary tasks understand the instance concept from different perspectives. Our findings show that exploiting the conceptual consistency of multiple auxiliary tasks to characterize the common part of these various understandings should reveal the target's hidden ground truth. Furthermore, we propose a novel method named multiple self-supervision conceptual consistency domain adaptation (MSCC). Experiments and analysis on benchmark datasets show the effectiveness of our idea and method.

Keywords deep learning, transfer learning, domain adaptation, domain shift, self-supervised

Citation Sun H, Li M. Enhancing unsupervised domain adaptation by exploiting the conceptual consistency of multiple self-supervised tasks. Sci China Inf Sci, 2023, 66(4): 142101, https://doi.org/10.1007/s11432-021-3535-2

1 Introduction

Machine learning (ML) has achieved remarkable performance in numerous real-world applications. In supervised learning, the reduction of the generalization error extremely depends on massive annotated data. However, manually annotating massive data for real-world applications is always time-consuming, labor-consuming, or even impossible. Hence, transferring knowledge from a similar but label-rich source domain to a label-poor target domain is an intuitive idea. However, the domain shift across domains poses a major obstacle in adopting predictive models for the target domain [1]. To tackle this problem, learning a model that reduces the domain shift across domains is known as domain adaptation (DA). Specifically, when the target domain has no annotation available, it is called unsupervised domain adaptation (UDA). Much effort has been made, such as instance-based adaptation [2], feature-representation adaptation [3], parameter adaptation [4], and relational-knowledge adaptation [5].

Recently, numerous deep domain adaptation approaches that leverage deep neural networks to learn more transferable representations have achieved significant performance. Many deep domain adaptation approaches mainly focus on learning domain-invariant embedding. Domain-level adaptation methods transfer knowledge by confusing the downstream predictor which domain the embedding comes from, e.g., [6–14]. They mainly focus on aligning the marginal distributions $P_s(\mathcal{X}_s)$ and $P_t(\mathcal{X}_t)$ across domains. In these methods, even with perfect domain-level confusion matching, the mismatch of the same category across domains is still inevitable. Domain-level adaptation methods should fail to preserve the discriminative patterns of the target during the aligning process. For example, the target dog's embedding may be mapped near the source cat's embedding. In this case, although the domain-level marginal distribution across domains may already have been aligned, the dog sample still could not be correctly classified

* Corresponding author (email: lim@lamda.nju.edu.cn)

because its embedding lost vital discriminative patterns. For instance, the target's embedding is poor for the downstream classifier trained by the source. The loss of vital discriminative patterns in the target's embedding would limit the performance of UDA.

Many researchers have attempted to improve UDA through a fine-grained category-level adaptation. Refs. [15–19] focused on subdomain adaptation by matching the poster distributions $P_s(\mathcal{X}_s|\mathcal{Y}_s)$ and $P_t(\mathcal{X}_t|\hat{\mathcal{Y}}_t)$, where $\hat{\mathcal{Y}}_t$ is the distribution of target pseudo labels generated by the source predictor. Moreover, Refs. [20, 21] avoided generating non-discriminative features lying in the region near the decision boundary of the source predictor. These category-level adaptation methods preserve the discriminative patterns of target instances during the aligning process, which is helpful for UDA. Nevertheless, these methods are utterly dependent on the source predictor, which is only trained by source supervision. Hence, the target domain could not guide which kind of features need to be captured, which would be dominated by source supervision. When the domain shift is large, these source predictors are arbitrary and dubious on target embedding. So, they can hardly help UDA or even degrade it.

To tackle these problems and further enhance UDA, one of the most crucial questions is whether we can overcome the dominance of source supervision in feature extraction and capture more target-intrinsic discriminative patterns for enhancing UDA. Specifically, extracting target-intrinsic patterns should not utterly rely on the source predictor, which is only trained by source supervision. It should be aware of the concept of the target instance itself. The intrinsic pattern could contain more additional information than the source-dominant feature extraction. Next, we will discuss how to extract and utilize intrinsic patterns of the target.

Considering that there is no available annotation in the target domain, how to capture intrinsic patterns and avoid entirely relying on the source predictor is worth discussing. Many practices have been proposed to adopt some additional auxiliary tasks for better feature extraction, e.g., [22, 23]. Therefore, training well-designed auxiliary tasks with the main (original) task to capture intrinsic patterns in the shared latent feature space is an intuitive and practical idea. Another critical problem is how to use intrinsic patterns captured by auxiliary tasks effectively. Previous studies always trained a separate auxiliary task head on the shared feature space to capture more valuable information [22, 23]. Nonetheless, they did not pay attention to two problems.

(1) Although more valuable information is captured in the shared feature space, the output of the main task head still loses some useful information because the separate main task head does not consider the prediction of auxiliary tasks. Moreover, it may ignore the intrinsic pattern captured by the auxiliary task heads.

(2) Not all intrinsic patterns captured by the auxiliary tasks are useful for the main task. Some invalid noise information would degrade UDA, especially when the domain shift is large.

The intrinsic pattern reflects the model's understanding of the instance concept for solving a specific auxiliary task. The understanding of the instance concept could be characterized in different spaces. In the embedding space, it could be the specific intrinsic pattern. In the auxiliary task output space, it could be the auxiliary task supervision $y \in \overline{\mathcal{Y}}$. In human descriptions, it may be color or shape descriptions. Moreover, if it can be characterized in the main task output space, i.e., describe the understanding of the instance concept by a similarity vector $y \in \mathcal{Y}$, then we call it the conceptual label. Multiple conceptual labels characterize different auxiliary tasks' understanding of the instance concept from different perspectives. Inspired by [24], the common (consistent) part of all conceptual labels from multiple perspectives should strongly correlate with the target's hidden actual ground truth. In other words, a reliable label should be consistent from multiple perspectives. Therefore, exploiting conceptual consistency (conceptual labels' consistency) to imply the common (consistent) understanding in the main task output space should reveal the target's hidden actual ground truth.

An example of a four-way classification task is shown in Figure 1. Three well-design auxiliary task classifiers share a common train set. Each of them is aware of the specific intrinsic pattern for solving the specific auxiliary task. For instance, the auxiliary task classifier C_0 is aware of the intrinsic pattern of color information ("color" is described by humans, the real intrinsic pattern characterized in the embedding space is more abstract). When the input is a zebra image, the C_0 captures the intrinsic pattern, reflecting the understanding of the input concept, such as "the main colors are white and black". Furthermore, it can be characterized in the main task output space $y \in \mathcal{Y}$ as the conceptual label $P_0(y|x)$: "like Panda and Zebra". It is characterized in the main task output space but not the hidden actual ground truth itself. Similar to C_0, the classifiers C_1 and C_2 capture the intrinsic patterns and characterize the conceptual labels from different perspectives. They have different understanding of the input instance

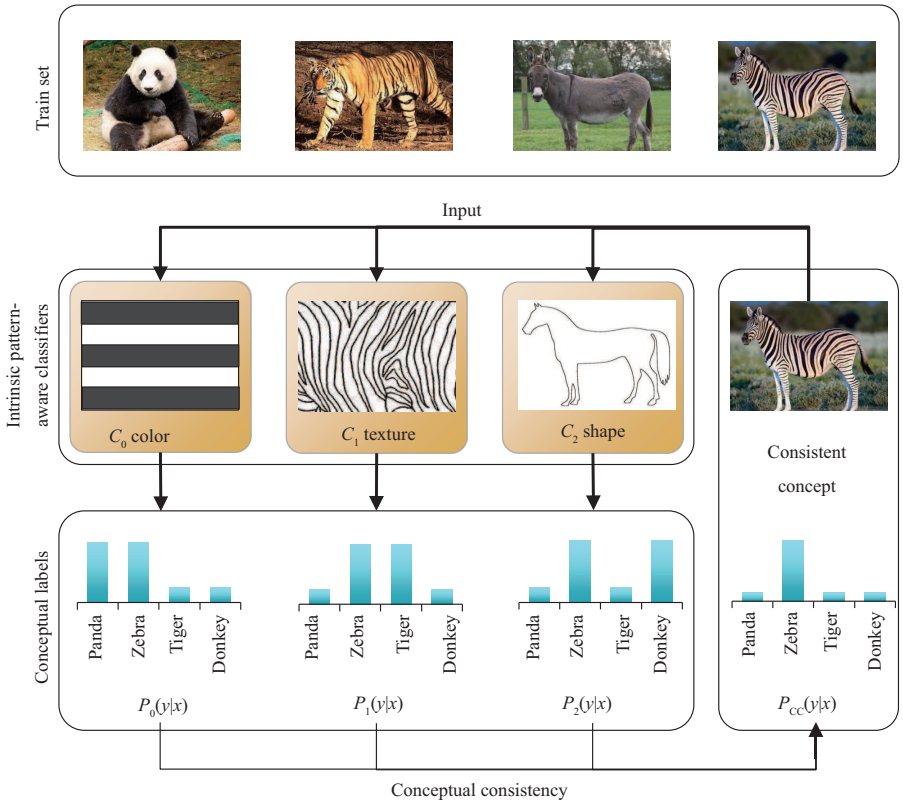

Figure 1 Four-way classification example of the conceptual consistency of multiple intrinsic patterns.

concept. However, the common (consistent) part of these conceptual labels $P_{\mathrm{CC}}(y|x)$ "Zebra", could reveal the hidden actual ground truth, which could be inferred by exploiting the conceptual consistency.

In this paper, we propose a novel framework named multiple self-supervision conceptual consistency domain adaptation (MSCC) to implement our idea for enhancing UDA. We adopt multiple annotating-free self-supervised auxiliary tasks to capture various intrinsic patterns with target self-supervision instead of utterly relying on the source's main task supervision. To obtain the conceptual label, we need to model the correlation between the output space of each auxiliary task and the main task and map the intrinsic pattern into the main task output space. Firstly, we construct the joint labels $(y, \overline{y}^i) \in \mathcal{Y} \times \overline{\mathcal{Y}}_i$ by taking a Cartesian product to the label space of the main task $\mathcal{Y}$. Each (e.g., the i-th) auxiliary task $\overline{\mathcal{Y}}_i$ utilizes the joint labels to train a joint-classifier, which outputs a two-dimensional joint probability $P_i(y, \overline{y}^i|x)$ to synchronously predict the specific auxiliary task and the main task. Secondly, we apply the total probability theorem on the joint-classifier output to characterize the conceptual label $P_i(y|x)$ with the main task prediction together, which preserves the correlation of the main task prediction $P(y|x)$ and specific (i-th) auxiliary task prediction $P_i(\overline{y}^i|x)$. Finally, we exploit the conceptual consistency by maximizing the agreement over conceptual labels to reveal the target's hidden actual ground truth.

Our main contributions are summarized as follows.

(1) We show that overcoming the dominance of source supervision to capture more target-intrinsic patterns in the feature extracting process could enhance UDA.

(2) We introduce a novel idea to leverage captured intrinsic patterns: we characterize the conceptual label and exploit the conceptual consistency to reveal the target's hidden actual ground truth. It is a new attempt at the utilization of auxiliary tasks in UDA.

(3) We propose a novel framework named MSCC, extensively evaluate our method on the standard benchmark datasets, and exceed state-of-the-art results on some of them.

The remainder of this paper is organized as follows. We first review related work in Section 2. Then, we introduce the proposed method in Section 3, including the framework of MSCC, training losses, and the overall training process. We show the experimental results in Section 4. Finally, we conclude this paper with a brief remark in Section 5.

2 Related work

2.1 Source-dominant pattern extraction

Deep domain adaptation attempts to learn a domain-invariant feature to reduce the domain shift, and previous studies mainly focus on the distribution alignment across domains.

Domain-level adaptation. Domain-level adaptation methods are directly motivated by the theory in [25]. They minimize the divergence between domains to lower the upper bound of the generalization error on the target domain. Therefore, domain-level adaptation methods focus on matching the source and target global marginal distributions, i.e., $P_s(\mathcal{X}_s)$ and $P_t(\mathcal{X}_t)$. These methods always use two leading technologies for distribution alignment: statistical moment matching and adversarial learning. There are many statistical moment-matching algorithms, such as deep domain confusion (DDC) [6], deep adaptation network (DAN) [7], residual transfer network (RTN) [8] and multiple kernel learning active learning (MKLAL) [14]. Such methods measure domain discrepancy in reproducing kernel Hilbert space (RKHS) using maximum mean discrepancy (MMD) and its variant. They match the marginal distribution in the shared latent feature space. Meanwhile, the adversarial learning algorithms, such as deep-adversarial neural network (DANN) [9], adversarial discriminative domain adaptation (ADDA) [11], and multi-source adaptation network (MSAN) [12], confuse the embedding cross-domain by adversarial learning. Furthermore, cycle-consistent adversarial domain adaptation (CyCADA) [13] confuses the input data cross-domain by adopting a variant of the generative adversarial networks (GANs) [26], i.e., the Cycle-GANs, to style transfer between the source domain and the target domain. However, these domain-level methods do not consider the relationship between the decision boundary and target features. Thus, they pay no effort to preserve the discriminative patterns of the target during the matching process. The target embedding may lose some vital discriminative patterns for classification, resulting in the mismatch of the same category cross-domain. Hence, the learned target feature would be inadequate for the downstream predictor.

Category-level adaptation. Recently, category-level adaptation has been proposed, and several researchers pay attention to preserving the discriminative patterns of the target domain during the matching process. Moving semantic transfer network (MSTN) [15], conditional domain adversarial networks (CDAN) [16], multi-adversarial domain adaptation (MADA) [17], and neural embedding matching (NEM) [19] match the poster distributions $P_s(\mathcal{X}_s|\mathcal{Y}_s)$ and $P_t(\mathcal{X}_t|\hat{\mathcal{Y}}_t)$ in the embedding space, where $\hat{\mathcal{Y}}_t$ is the distribution of target pseudo labels generated by the source predictor. Moreover, adversarial dropout regularization (ADR) [20] and maximum classifier discrepancy (MCD) [21] avoid generating non-discriminative features lying in the region near the decision boundary of the source predictor, i.e., encourage the feature extractor to output more discriminative patterns for the target instance. These category-level adaptation methods have shown that the source predictor can guide the model to preserve more discriminative patterns of the target instance during the matching process. Moreover, it is helpful for UDA. Nevertheless, these strategies are utterly dependent on the source predictor, which is only trained by source supervision. In other words, feature extraction is still dominated by source supervision. When the domain shift is large, the source predictor still performs poorly for the target instance. Their outputs on the target domain are arbitrary and dubious, so they can hardly help UDA or even degrade UDA results from the wrong prediction.

2.2 Target-intrinsic pattern extraction

This paper assumes that some discriminative target-intrinsic patterns would be lost in the above source-dominant pattern extraction methods, and some target-intrinsic patterns could enhance UDA, especially when the domain shift is large. Our proposed method, i.e., MSCC, tries to overcome the dominance of source supervision in feature extraction. MSCC makes the target domain guide which kind of features are captured, rather than just aligning the distribution of target features with the source features. MSCC adopts auxiliary tasks on the target domain to directly capture intrinsic patterns without utterly relying on the source data. These captured intrinsic patterns could be additional information to the global marginal distribution and break the limitation of source supervision.

The most relevant methods are deep reconstruction classification network (DRCN) [22] and [23]'s method. They have tried to use auxiliary tasks to benefit feature extraction. However, they add a separate head on the shared embedding space for the auxiliary task, which still has two fatal drawbacks.

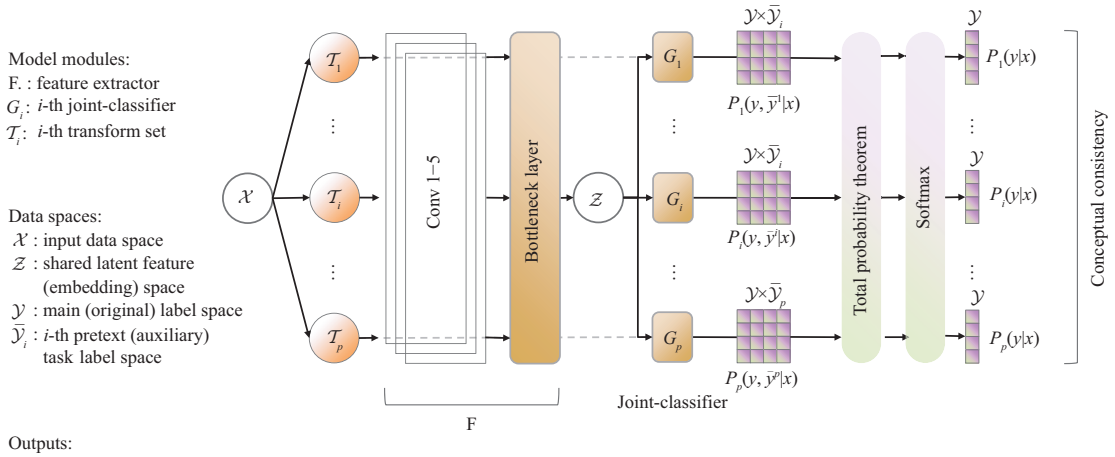

Figure 2 The framework of our multiple self-supervision conceptual consistency domain adaptation (MSCC).

(1) Although more valuable patterns would be captured in the shared embedding space, the final output of the main task head maybe still lose some useful information.

(2) Not all captured intrinsic patterns are useful for the main task. Some invalid noise information would degrade UDA, especially when the domain shift is large.

In short, the separate main task predictor could not correctly utilize captured intrinsic patterns. This problem has not been researched in detail. We propose a new method (MSCC) to extract and utilize intrinsic patterns by auxiliary tasks and overcome the above drawbacks by exploiting the conceptual consistency of multiple self-supervised auxiliary tasks.

3 Method

In this section, we will introduce MSCC about how to extract intrinsic patterns from the unlabeled target domain and apply them to enhance UDA. The framework of MSCC is shown in Figure 2.

We adopt multiple annotating-free self-supervised auxiliary tasks to capture intrinsic patterns from different perspectives. Different data augmentation transformations are applied to the input samples, and then the applied transformation types can be used as annotating-free labels to construct an auxiliary task. The model understands the instance concept from the specific perspective to work out the auxiliary task, and the understanding would be reflected in the captured intrinsic pattern. The detail about extracting intrinsic patterns by auxiliary tasks will be covered in Subsection 3.2.

We preserve the correlation between each auxiliary task output space and the main task output space for characterizing conceptual labels. Firstly, we construct joint labels by taking Cartesian product to the label space of the main task and each auxiliary task. Then, we utilize the joint labels to train the joint-classifier to output a two-dimensional joint probability for predicting the specific auxiliary task and the main task synchronously. Secondly, we apply the total probability theorem on the joint-classifier output, i.e., the joint probability, to characterize the conceptual label in the main task label space. Thirdly, we exploit the conceptual consistency by maximizing the agreement among all conceptual labels to imply the common (consistent) part understanding of the instance concept, which should have a strong correlation with the target's hidden actual ground truth. The detail about joint-classifier, conceptual label, and conceptual consistency will be discussed in Subsection 3.3.

3.1 Formulation and notation

Unsupervised domain adaptation specifically addresses the situation where there are labeled source domain dataset $\mathcal{D}_s : (\mathcal{X}_s, \mathcal{Y}_s) = \{x_i^s, y_i^s\}_{i=1}^{n_s}$ with n_s labeled instances and only unlabeled target dataset $\mathcal{D}_t : \mathcal{X}_t = \{x_i^t\}_{i=1}^{n_t}$ with n_t instances available for training. Moreover, the source domain and the target domain have the same label space, i.e., $\mathcal{Y}_s = \mathcal{Y}_t$. However, there is domain shift; cross-domain data are

drawn from different distributions $P_s(x^s, y^s) \neq P_t(x^t, y^t)$. UDA aims to reduce the domain shift and transfer the knowledge from source data to improve the evaluation performance on target data.

The framework of MSCC is shown in Figure 2. $\mathcal{X} = \mathcal{X}_s$ or $\mathcal{X}_t$ is the original input space, $\mathcal{Z}$ is the shared latent feature space, and $\mathcal{Y} = \mathcal{Y}_s = \mathcal{Y}_t$ means the main task label space. Besides, there are p self-supervised auxiliary tasks with p transform sets $\{\mathcal{T}_1, \ldots, \mathcal{T}_i, \ldots, \mathcal{T}_p\}$, and each transform-set has $|\mathcal{T}_i|$ transform functions: $\mathcal{T}_i = \{t_1^i, t_2^i, \ldots, t_{|\mathcal{T}_i|}^i\}$, and its transform type label space is $\overline{\mathcal{Y}}_i = \{\overline{y}_1^i, \overline{y}_2^i, \ldots, \overline{y}_{|\mathcal{T}_i|}^i\}$. For the i-th self-supervised auxiliary task, the model needs to predict which transform functions have been applied in the i-th transform-set $\mathcal{T}_i$. And for each (i-th) auxiliary task, there is a joint-classifier $G_i : \mathcal{Z} \to \mathcal{Y} \times \overline{\mathcal{Y}}_i$, which outputs a two-dims joint probability $P_i(y, \overline{y}^i|x)$ to predict the main task and auxiliary task synchronously, where $\mathcal{Y} \times \overline{\mathcal{Y}}_i$ means Cartesian product of the main task label space $\mathcal{Y}$ and the i-th auxiliary task label space $\overline{\mathcal{Y}}_i \in \{\overline{y}_1^i, \overline{y}_2^i, \ldots, \overline{y}_{|\mathcal{T}_i|}^i\}$. Moreover, $P_i(y|x)$ means the conceptual label from the i-th joint-classifier. Besides, we write $C_i = G_i \circ F$, where $F : \mathcal{X} \to \mathcal{Z}$ is a shared feature extractor.

3.2 Extracting intrinsic patterns

Considering there is no available label in the target domain, we adopt the transform-based self-supervised tasks as auxiliary tasks. These kinds of tasks need not annotate manually. Instead, their supervision could be generated automatically. Transform-based self-supervised tasks aim to capture valuable patterns by predicting the type of applied transformations that should be helpful for the main task.

(1) Rotation prediction [27]. An input image is rotated in 90-degree increments, i.e., 0°, 90°, 180°, 270°. The self-supervised task is to predict the angle of rotation as a four-way classification problem.

(2) Color permutation [28]. The channel permutation of the input image is randomly shuffled. The self-supervised task is a classification task about predicting the channel permutation.

(3) Patch location prediction [23]. Patches are randomly cropped out of an input image. The self-supervised task is to predict the position of the patch.

(4) Flip prediction. The input image is randomly flipped vertically. The self-supervised task is to predict whether the image is flipped or not.

The different self-supervised tasks have different understandings of the instance concept and capture the specific intrinsic pattern. For example, rotation prediction may enhance the awareness of the spatial patterns, while color permutation may enhance the preserving of color patterns, and other self-supervised tasks may capture more abstract patterns which are difficult to describe by a human. Whatever, these self-supervised tasks can capture more intrinsic patterns in the shared feature space, which is always helpful to the main task. Most importantly, the target self-supervisions are independent of the source data. So the feature extraction of the target could break the limitation of source supervision and learn more intrinsic discriminative patterns of the target.

The following is how to use the captured intrinsic patterns as additional information to enhance UDA.

3.3 Exploiting the conceptual consistency

Numerous practices have shown that self-supervised auxiliary tasks can capture more helpful features in the shared embedding space [29,30]. Previous self-supervised learning methods always add a separate head on the shared feature space and output $P(\overline{y}|x)$ to solve the self-supervised task independently [23,27,29]. These methods still have two apparent limitations.

(1) Although more intrinsic patterns would be captured in the shared feature space, the final output of the main task head maybe still lose some useful information since the separate main task head does not consider the prediction of auxiliary tasks, and it may ignore the intrinsic pattern captured by the auxiliary task heads.

(2) The captured intrinsic pattern reflects the understanding of the instance concept by the specific self-supervised auxiliary task. Nevertheless, it is not a direct understanding of the main task's hidden actual ground truth. The auxiliary tasks may capture some noise information and degrade the main task.

How to use the additional information captured by the auxiliary tasks remains to be researched carefully. This paper will introduce a novel approach to utilizing the auxiliary tasks: exploiting the conceptual consistency of multiple self-supervised auxiliary tasks.

To break the first limitation, MSCC models the correlation between the main task and each auxiliary task in their output spaces, and then we can bridge the captured intrinsic pattern and the main task

output space. Some researchers have considered the correlation between multi-task predictors: Ref. [31] adopted lasso training. Meanwhile, Refs. [28, 32] designed a correlative predictors' architecture.

3.3.1 *Joint-classifier*

In MSCC, we train a joint-classifier that preserves the correlation between the output of the main task $P(y|x)$ and the specific self-supervised (i-th) auxiliary task $P_i(\overline{y}^i|x)$, while previous methods separate them. Firstly, we take Cartesian product to the label space of the main task $\mathcal{Y}$ and each auxiliary task $\overline{\mathcal{Y}}_i$ for constructing the joint label space $\mathcal{Y} \times \overline{\mathcal{Y}}_i$. For example, the main task label space is $\mathcal{Y} = \{\text{cat}, \text{dog}\}$, and the flip prediction self-supervised auxiliary task label space is $\overline{\mathcal{Y}}_{\text{flip}} = \{\text{Flip}, \text{no Flip}\}$, then the joint label space of them would contain four two-dimensional labels: $\mathcal{Y} \times \overline{\mathcal{Y}}_{\text{flip}} = \{(\text{cat}, \text{Flip}), (\text{cat}, \text{no Flip}), (\text{dog}, \text{Flip}), (\text{dog}, \text{no Flip})\}$. Secondly, we train a joint-classifier outputs two-dimensional joint probability to predict the joint label, i.e., $P_i(y, \overline{y}^i|x) = G_i(t^i(x))$, where t^i is a transform function sampled from the i-th transform set $\mathcal{T}_i = \{t_1^i, t_2^i, \ldots, t_{|\mathcal{T}_i|}^i\}$. So, the loss of the i-th joint-classifier (for the main task and the i-th auxiliary task, also the i-th transform set) is

$$\mathcal{L}_{\text{Joint}}(\mathcal{D}, \mathcal{T}_i) = \frac{1}{n|\mathcal{T}_i|} \sum_{j=1}^{n} \sum_{k=1}^{|\mathcal{T}_i|} \mathcal{L}_{\text{CE}}(C_i(t_k^i(x_j)), (y_j, \overline{y}_k^i)). \tag{1}$$

where $t_k^i(\cdot)$ means the k-th transform function in i-th transform set $\mathcal{T}_i$. $(y_j, \overline{y}_k^i)$ is the joint label, where $y_j \in \mathcal{Y}$ is the main task label of the j-th sample. And, $\overline{y}_k^i \in \overline{\mathcal{Y}}_i$ is the i-th auxiliary task label. Specifically, $\overline{y}_k^i$ reveals that the k-th transform function in $\mathcal{T}_i$ has been applied, i.e., $\overline{y}_k^i = k$. Moreover, $\mathcal{L}_{\text{CE}}$ is the cross-entropy loss function. Besides, to calculate this loss on target domain, we need a few pseudo labels of the target data, which will be discussed in Subsection 3.4.

As for the second limitation, the self-supervised auxiliary task understands the instance concept from a specific perspective. Furthermore, the understanding of the instance concept could be characterized in the different spaces. For example, in the shared feature space, it could be the intrinsic pattern. In the self-supervised auxiliary task label space $\overline{\mathcal{Y}}$, it could be the specific self-supervised task label $\overline{y}^i \in \overline{\mathcal{Y}}_i$. In human's description space, it could be which color or shape it is. MSCC tries to characterize it in the main task output space, i.e., describe the understanding of instance concept through a similarity vector $y \in \mathcal{Y}$ in the main task label space. We call this similarity vector conceptual label.

3.3.2 *Conceptual label*

It is not easy to characterize the conceptual label directly in practice. Fortunately, the i-th joint-classifier outputs a two-dimensional joint probability, i.e., $P_i(y, \overline{y}^i|x)$. We can apply total probability theorem on the output of the joint-classifier to calculate the conceptual label, which is a similarity vector in the main task label space:

$$P_i(y|x) = \int_{\overline{y}^i} P_i(y, \overline{y}^i|x) d_{\overline{y}^i} = \int_{\overline{y}^i} P_i(y|\overline{y}^i, x) P_i(\overline{y}^i|x) d_{\overline{y}^i}. \tag{2}$$

Total probability theorem does not need the independence between two events so that it could preserve the correlation between the main task prediction $P(y|x)$ and the specific (i-th) auxiliary task prediction $P_i(\overline{y}^i|x)$. In this way, MSCC is aware of the specific intrinsic pattern for solving the auxiliary task and "project" it into the main task label space as the conceptual label, i.e., preserves the knowledge learned from the auxiliary task in the conceptual label.

In practice, we apply total probability theorem on pre-softmax activations (i.e., logits) layer. In the i-th joint-classifier, the conditional probability on pre-softmax activations (i.e., logits) is defined as

$$P_i(y|\overline{y}_k^i, x) = P_i(y|\overline{y}^i = k, x) = C_i(t_k^i(x))[:, k]. \tag{3}$$

In each self-supervised auxiliary task, we randomly apply each transform function with equal probability:

$$P_i(\overline{y}_k^i|x) = P_i(\overline{y}^i = k|x) = \frac{1}{|\mathcal{T}_i|}, \quad \forall k \in \{1, 2, \ldots, |\mathcal{T}_i|\}. \tag{4}$$

Therefore, the i-th conceptual label characterized by the i-th self-supervised auxiliary task is

$$P_i(y|x) = \text{softmax}\left(\frac{1}{|\mathcal{T}_i|}\sum_{k=1}^{|\mathcal{T}_i|} C_i(t_k^i(x))[:,k]\right). \tag{5}$$

3.3.3 *Conceptual consistency*

We have characterized multiple conceptual labels to manifest the understanding of the instance concept from different self-supervised auxiliary tasks' perspectives. As discussed above, these auxiliary tasks do not directly understand the main task label. In other words, the conceptual label is not the hidden actual ground truth itself. The case in Figure 1 also shows it. Inspired by [24], the consistent (common) part of all conceptual labels from multiple perspectives should strongly correlate with the target's hidden actual ground truth. There is only a consistent actual ground truth for every instance. Hence, exploiting the conceptual consistency (conceptual labels' consistency) to obtain the consistent concept of multiple conceptual labels should reveal the target's hidden actual ground truth. MSCC maximizes the agreement on multiple conceptual labels of the target instance to get a reliable label for the main task.

The conceptual consistency loss is

$$\mathcal{L}_{\text{CC}}(\mathcal{D}_t) = \frac{1}{p^2 n_t}\sum_{i=1}^{p}\sum_{j=1}^{p}\sum_{k=1}^{n_t}\mathcal{L}_{\text{CE}}(P_i(y|x_k^t), P_j(y|x_k^t)). \tag{6}$$

3.4 Pseudo labeler

For constructing joint labels of the target domain, we need a few main task pseudo labels of the target domain. Generating pseudo labels for the unlabeled instance is a common technique in machine learning tasks that lack labels, e.g., semi-supervised learning (including positive and unlabeled learning) [24, 33], unsupervised domain adaptation [34, 35]. Category-level adaptation approaches [15–17, 20, 21] depend on pseudo labels from the source predictor utterly. In contrast, MSCC leverages a part of pseudo labels to bridge the main task and the additional target-intrinsic patterns captured by self-supervised auxiliary tasks. Previous domain adaptation method joint adaptation network (JAN) [36] would be directly applied as a base pseudo labeler in this paper. We add a separate pseudo labeler on the shared features space: $G_{\text{PL}} : \mathcal{Z} \to \mathcal{Y}$. And, we write $C_{\text{PL}} = G_{\text{PL}} \circ F$. Of course, MSCC can use most previous adaptation methods as the base pseudo labeler and enhance the pseudo labeler further. Moreover, MSCC would have a better performance by combining with a better pseudo labeler in the future.

There are two losses in the JAN. One of them trains the classifier on the source data:

$$\mathcal{L}_{\text{PL}}(\mathcal{D}_s) = \frac{1}{n_s}\sum_{i=1}^{n_s}\mathcal{L}_{\text{CE}}(C_{\text{PL}}(x_i^s), y_i^s). \tag{7}$$

Moreover, the other one is transfer loss: joint maximum mean discrepancy (JMMD), a variant of MMD, which aligns the joint distribution of the feature and the classifier output across domains:

$$\begin{aligned}\mathcal{L}_{\text{trans}}(\mathcal{D}_s, \mathcal{D}_t) = &\frac{1}{n_s^2}\sum_{i=1}^{n_s}\sum_{j=1}^{n_s} k_z(z_i^s, z_j^s)k_y(\hat{y}_i^s, \hat{y}_j^s)\\ &+ \frac{1}{n_t^2}\sum_{i=1}^{n_t}\sum_{j=1}^{n_t} k_z(z_i^t, z_j^t)k_y(\hat{y}_i^t, \hat{y}_j^t)\\ &- \frac{2}{n_s n_t}\sum_{i=1}^{n_s}\sum_{j=1}^{n_t} k_z(z_i^s, z_j^t)k_y(\hat{y}_i^s, \hat{y}_j^t),\end{aligned} \tag{8}$$

where $z = F(x)$, $\hat{y} = \text{softmax}(G_{\text{PL}}(z))$, and $k_\cdot(\cdot,\cdot)$ is Gaussian kernel function.

The target predictions near the class boundary are ambiguous, which may degrade UDA. Therefore, we design a pseudo label select function to select pseudo labels $\hat{y}_i^t = \text{softmax}(C_{\text{PL}}(x_i^t))$ in high-confidence:

$$\mathbb{S}_\tau(\hat{y}) = \begin{cases}1, & \max(\hat{y}) \geqslant \tau,\\ 0, & \text{otherwise.}\end{cases} \tag{9}$$

We only annotate the target samples with high-confident pseudo labels: $\hat{\mathcal{D}}_t = \{(x_i^t, \hat{y}_i^t) |\ \mathbb{S}_\tau(\hat{y}_i^t) = 1\}$.

3.5 Overall training process

In the training process, we online train the pseudo labeler with the joint-classifiers together. Thus, MSCC just needs to optimize the total loss:

$$\begin{aligned}\mathcal{L} = &\mathcal{L}_{\mathrm{PL}}(\mathcal{D}_s \cup \tilde{\mathcal{D}}_s \cup \hat{\mathcal{D}}_t \cup \tilde{\mathcal{D}}_t) + \mathcal{L}_{\mathrm{trans}}(\mathcal{D}_s \cup \tilde{\mathcal{D}}_s, \hat{\mathcal{D}}_t \cup \tilde{\mathcal{D}}_t) \\ &+ \frac{1}{p}\sum_{i=1}^{p} \mathcal{L}_{\mathrm{Joint}}(\mathcal{D}_s, \mathcal{T}_i) + \frac{1}{p}\sum_{i=1}^{p} \mathcal{L}_{\mathrm{Joint}}(\hat{\mathcal{D}}_t, \mathcal{T}_i) \\ &+ \lambda_{\mathrm{CC}}\mathcal{L}_{\mathrm{CC}}(\mathcal{D}_t). \end{aligned} \tag{10}$$

In the first line, they are the training losses of pseudo labeler on source data $\mathcal{D}_s$ and a part of target data with pseudo labels $\hat{\mathcal{D}}_t$, and their MixUp [37] data $\tilde{\mathcal{D}}_s \cup \tilde{\mathcal{D}}_t$. The first term means the supervised training loss, and the second term denotes the transfer loss to match the cross-domain distributions. In the second line, it is the joint-classifiers training losses. These two terms respectively represent the average supervised training loss of joint-classifiers on source data $\mathcal{D}_s$ and a part of the target data with pseudo labels $\hat{\mathcal{D}}_t$. Finally, the last term in the third line means the conceptual consistency loss on all target instances $\mathcal{D}_t$.

We use mixed data $\tilde{\mathcal{D}}_s$ and $\tilde{\mathcal{D}}_t$ calculated by MixUp [37] to train the pseudo labeler. MixUp is an unconventional data enhancement method that uses linear interpolation to construct new training samples and labels. Vanilla MixUp and its variants have been widely used to augment visual data [37, 38].

The way we apply MixUp cross-domain for UDA is

$$\begin{aligned}&\lambda \sim \mathrm{Beta}(\alpha, \alpha), \quad \lambda = \max(\lambda, 1-\lambda), \\ &(\tilde{\mathcal{D}}_s, \tilde{\mathcal{D}}_t) = \lambda(\mathcal{D}_s, \hat{\mathcal{D}}_t) + (1-\lambda)\,\mathrm{Shuffle}((\mathcal{D}_s, \hat{\mathcal{D}}_t)). \end{aligned} \tag{11}$$

MixUp cross-domain here has four benefits: (1) data augmentation; (2) smoothing the pseudo labels and enhancing fault tolerance; (3) mixing source and target instance to confuse the classifier in input directly and easily, while most previous UDA methods confuse it by matching the distributions cross-domain in the embedding space; (4) filling the gap between source and target by augmenting the linear interpolation data. For instance, in maximum mean discrepancy, the large domain shift will result in all $k(z_i^s, z_j^t)$ tiny, making the distribution alignment to be a challenge. MixUp can alleviate this problem.

4 Experiments

We implement MSCC in PyTorch based on the Transfer-Learning-library [39]. We extensively evaluate MSCC and compare it with state-of-the-art domain adaptation methods on the standard benchmark visual datasets, including Office-31, Office-Home, and VisDA-2017. Following standard evaluation protocols for UDA as [36], we use all labeled source instances and all unlabeled target instances to compare the average classification accuracy. Specifically, on the small dataset Office-31, we conduct three random experiments.

4.1 Setup

Datasets.

• Office-31 is a standard benchmark dataset for domain adaptation in computer vision, comprising 31 categories collected from three distinct domains: Amazon (A), which contains 2817 images downloaded from amazon.com, Webcam (W) with 795 images and DSLR (D) with 498 images, which taken by web camera and digital SLR camera under different settings. We evaluate all methods across six transfer tasks as A $\rightarrow$ D, A $\rightarrow$ W, D $\rightarrow$ A, D $\rightarrow$ W, W $\rightarrow$ A, W $\rightarrow$ D in [6, 36].

• Office-Home is a better organized but more difficult dataset than Office-31, which consists of 15588 images in 65 object classes in office and home settings, forming four extremely dissimilar domains: artistic images (A), clip art (C), product images (P) and realworld images (R).

• VisDA-2017 is a challenging simulation-to-real dataset, with two very distinct domains: synthetic, rendering 3D model from different angles and with different lighting conditions; real, natural images. It contains 152397 synthetic images and 55388 real images across 12 classes.

Baselines. For Office-31 datasets, we compare MSCC with ResNet50 (source only) [40], DDC, DAN, RTN, DANN, JAN, ADDA, MADA, CDAN, manifold embedded distribution alignment (MEDA) [18], selective pseudo-labeling (SPL) [35], and FixBi [41]. For Office-Home, we compare MSCC with ResNet50 (source only), DAN, DANN, JAN, CDAN, MEDA, SPL, and FixBi. Besides, for VisDA-2017, we compare MSCC with ResNet101 (source only), DAN, DANN, JAN, CDAN, MCD, ADR, and FixBi. All results are cited from [16] or their original articles, except for the results of CDAN on VisDA-2017 and the results of FixBi. We run the official code of CDAN[1)] on VisDA-2017 ourselves. The FixBi is a fine-tune method that needs a base model to pre-train the parameters. For fairness, we pre-train JAN by the official code[1)] and run the official code of FixBi[2)] based on these pre-trained parameters.

Implementation detail. We employ ResNet [40] pre-trained on ImageNet as the backbone. Moreover, we select rotation prediction, patch location prediction, and flip prediction as transform-based self-supervised auxiliary tasks. And following the setup in [7, 9], we use mini-batch stochastic gradient descent (SGD) to optimize the total loss 30 epochs with 500 batch iterations in every epoch, and the learning rate annealing strategy: the learning rate is adjusted by $\eta_p = \eta_0(1 + \alpha p)^{-\beta}$, where p is the training progress linearly changing from 0 to 1, and $\eta_0 = 0.01$, $\alpha = 10$, $\beta = 0.75$. Besides, the conceptual consistency loss trade-off λ_{CC} is ramp-up with Gaussian function $\exp(-\delta(1-p)^2)$ where $\delta = 5$ and p is linearly changing from 0 to 1 when the epoch from 5 to 15. This strategy has been used in many semi-supervised consistency loss trade-off settings [38, 42]. Furthermore, we adopt a reasonable pseudo label threshold $\tau = 0.9$ in (9).

4.2 Comparison with state-of-the-art methods

In this subsection, we compare MSCC with the state-of-the-art adaptation methods on three standard benchmark datasets. The classification results on Office-31, Office-Home, and VisDA-2017 are reported in Tables 1–3. MSCC significantly outperforms all methods we compared on most tasks, especially on the challenging datasets Office-Home and VisDA-2017. It corroborates the effectiveness of MSCC.

The results reveal several insightful observations.

(1) Deep transfer learning methods substantially outperform standard deep learning methods (source only). It validates that explicitly reducing the cross-domain discrepancy by embedding domain adaptation modules into deep networks can learn more transferable features.

(2) The recent category-level adaptation methods, e.g., CDAN, MADA, MCD, ADR, and our MSCC, pay more attention to the discriminative patterns of the target in addition to the global marginal distribution information. They have achieved better performance than prior domain-level adaptation methods that only use the target data in domain-level distribution alignment. It demonstrates some discriminative patterns of target instances, in addition to the global marginal distribution information, which is beneficial for domain adaptation.

(3) Comparing MSCC with the source-dominant feature extraction methods, e.g., CDAN, MADA, MEDA, SPL, MCD, and ADR, we achieve better performance. It verifies that the intrinsic patterns captured in target data themselves could further enhance UDA. Besides, it validates the algorithm for extracting the intrinsic patterns through self-supervised auxiliary tasks, and utilizing them by conceptual consistency in MSCC is effective.

(4) It is noteworthy that our method promotes the classification accuracy substantially on the challenging datasets Office-Home and VisDA-2017. Office-Home contains more categories, and the domains in VisDA-2017 are very distinct; i.e., the domain shift is huge. It reveals that source-dominated feature extraction on target samples is poor in these complex UDA situations. Thus, correctly extracting and utilizing the target-intrinsic patterns in complex domain adaptation problems is vital and effective.

(5) In addition, it needs to be pointed out that MSCC outperforms the other methods obviously on Office-Home while adopting domain C as the target domain since the domain shift between domain C and the remaining three domains is quite large. It validates the effectiveness of intrinsic patterns in challenging tasks again. However, most methods can well handle the case when domain C is the source domain. We think this may be because the style of the remaining three domains and ImageNet is similar, and most methods use ImageNet to pre-train the backbone network.

(6) The most exciting is that compared with the base pseudo labeler method, i.e., JAN, MSCC has apparent improvements on all datasets: on Office-31, the improvement is 6.3%, on the Office-Home, it is

1) https://github.com/thuml/Transfer-Learning-Library.

2) https://github.com/NaJaeMin92/FixBi.

Table 1 Accuracy (%) on Office-31 for unsupervised domain adaptation (backbone: ResNet-50)[a)]

Method	A → D	A → W	D → A	D → W	W → A	W → D	Average
ResNet50	68.9	68.4	62.5	96.7	60.7	99.3	76.1
DDC	76.5	75.6	62.2	96.0	61.5	98.2	78.3
DAN	78.6	80.5	63.6	97.1	62.8	99.6	80.4
RTN	77.5	84.5	66.2	96.8	64.8	99.4	81.6
DANN	79.7	82.0	68.2	96.9	67.4	99.1	82.2
ADDA	77.8	86.2	69.5	96.2	68.9	98.4	82.9
MADA	87.8	90.0	70.3	97.4	66.4	99.6	85.2
CDAN	89.8	93.1	70.1	98.2	68.0	**100.0**	86.6
MEDA	86.3	86.0	72.1	97.1	73.2	99.2	85.7
SPL	93.0	92.7	**76.4**	**98.7**	**76.8**	99.8	89.6
JAN	84.7	85.4	68.6	97.4	70.0	99.8	84.3
FixBi+JAN	87.0	92.6	68.6	94.6	69.3	97.4	84.9
MSCC+JAN	**95.0**	**97.4**	76.2	**98.7**	76.1	**100.0**	**90.6**

a) Results in bold indicate the best experimental results in the current transfer task.

Table 2 Accuracy (%) on Office-Home for unsupervised domain adaptation (backbone: ResNet-50)[a)]

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	Average
ResNet50	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
DAN	43.6	57.0	67.9	45.8	56.5	60.4	44.0	43.6	67.7	63.1	51.5	74.3	56.3
DANN	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
CDAN	49.0	69.3	74.5	54.4	66.0	68.4	55.6	48.3	75.9	68.4	55.4	80.5	63.8
MEDA	54.9	75.9	77.2	58.1	73.3	71.5	59.0	52.6	77.8	67.9	57.6	81.8	67.3
SPL	54.5	77.8	**81.9**	65.1	**78.0**	**81.1**	66.0	53.1	**82.8**	69.9	55.3	86.0	71.0
JAN	45.9	61.2	68.9	50.4	59.7	61.0	45.8	43.4	70.3	63.9	52.4	76.8	58.3
FixBi+JAN	54.5	72.0	76.5	59.7	67.0	69.2	59.9	51.6	77.2	70.6	58.9	81.4	66.5
MSCC+JAN	**61.8**	**78.6**	**81.9**	**70.3**	77.5	79.0	**67.8**	**61.7**	82.3	**76.1**	**66.9**	**86.3**	**74.2**

a) Results in bold indicate the best experimental results in the current transfer task.

Table 3 Accuracy (%) on VisDA-2017 for unsupervised domain adaptation (backbone: ResNet-101)[a)]

Method	AirPlane	Bicycle	Bus	Car	Horse	Knife	Motorcycle	Person	Plant	Skateboard	Train	Truck	Average
ResNet101	72.3	6.1	63.4	**91.7**	52.7	7.9	80.1	5.6	90.1	18.5	78.1	25.9	49.4
DAN	68.1	15.4	76.5	87.0	71.1	48.9	82.3	51.5	88.7	33.2	88.9	42.2	62.8
DANN	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
CDAN	93.6	82.3	66.2	80.6	92.7	10.7	87.1	70.0	**94.6**	38.4	76.6	47.2	70.0
MCD	87.0	60.9	83.7	64.0	88.9	79.6	84.7	76.9	88.6	40.3	83.0	25.8	71.9
ADR	87.8	79.5	83.7	65.3	92.3	61.8	88.9	73.2	87.8	60.0	85.5	32.3	74.8
JAN	75.7	18.7	82.3	86.3	70.2	56.9	80.5	53.8	92.5	32.2	84.5	**54.5**	65.7
FixBi+JAN	91.7	67.8	78.9	46.5	82.3	**95.0**	85.6	74.8	78.1	73.7	86.5	43.3	75.4
MSCC+JAN	**96.2**	**86.5**	**88.8**	62.8	**96.3**	93.8	**94.2**	**78.3**	92.5	**94.8**	**93.6**	51.3	**85.8**

a) Results in bold indicate the best experimental results in the current transfer task.

15.9%, and on the VisDA-2017 it is 20.1%. The improvement in the more complex tasks is more evident than in the easy tasks, which once again illustrates the importance and effectiveness of MSCC in complex UDA. Furthermore, the JAN is not the best approach in prior methods, and adopting the better base pseudo labeler in MSCC should have better performance. However, this paper mainly wants to validate that our framework can enhance the base adaptation method better. And in future work, we may use a better base pseudo labeler to improve UDA's ultimate performance further.

4.3 Effectiveness experiments

We conduct effectiveness experiments by ablation study and show the result in Tables 4 and 5.

As shown in Table 4, there are three main components: conceptual consistency (CC), self-supervised auxiliary tasks (SS), and MixUp. Moreover, there are four ablation study experiments (ID 3–6).

Exp. 3: without MixUp data.

Table 4 Effectiveness of conceptual consistency (CC), self-supervised auxiliary tasks (SS), and MixUp on Office-Home (backbone: ResNet-50)[a)]

ID	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	Average	Δ
1	ResNet50	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	–
2	JAN	45.9	61.2	68.9	50.4	59.7	61.0	45.8	43.4	70.3	63.9	52.4	76.8	58.3	–
3	w/o MixUp	58.2	77.3	81.6	65.7	77.2	77.4	65.6	61.0	**83.6**	75.2	65.9	**86.7**	73.0	−1.2
4	w/o CC	56.9	74.6	77.8	65.3	73.8	73.7	66.2	58.5	80.1	**76.2**	63.2	84.9	70.9	−3.3
5	w/o SS, CC	57.5	73.0	77.9	66.7	74.2	75.3	65.3	57.2	79.5	74.3	63.0	83.2	70.6	−3.6
6	w/o SS, CC with Ens	57.8	75.5	77.2	64.9	76.8	74.5	64.9	57.7	79.8	73.0	63.4	83.8	70.8	−3.4
7	MSCC	**61.8**	**78.6**	**81.9**	**70.3**	**77.5**	**79.0**	**67.8**	**61.7**	82.3	76.1	**66.9**	86.3	**74.2**	–

a) Results in bold indicate the best experimental results in the current transfer task.

Table 5 Effectiveness when choosing different pseudo labelers on Office-Home (backbone: ResNet-50)[a)]

Pseudo labeler	Protocol	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	Average	Δ
ResNet50	Original	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	–
	SH	51.4	68.2	74.9	56.7	68.6	68.1	55.6	50.5	75.0	68.4	58.3	81.5	64.8	+18.7
	MSCC	**58.7**	**77.2**	**80.3**	**68.3**	**75.6**	**77.9**	**66.5**	**62.8**	**82.4**	**75.0**	**66.1**	**86.5**	**73.1**	+27.0
DAN	Original	43.6	57.0	67.9	45.8	56.5	60.4	44.0	43.6	67.7	63.1	51.5	74.3	56.3	–
	SH	56.2	73.3	78.4	62.2	73.3	71.5	63.2	57.1	80.3	**75.1**	62.0	85.2	69.8	+13.5
	MSCC	**59.5**	**78.3**	**80.4**	**68.5**	**77.7**	**78.5**	**66.0**	**61.6**	**81.9**	74.7	**66.5**	**85.8**	**73.3**	+17.0
JAN	Original	45.9	61.2	68.9	50.4	59.7	61.0	45.8	43.4	70.3	63.9	52.4	76.8	58.3	–
	SH	56.9	74.6	77.8	65.3	73.8	73.7	66.2	58.5	80.1	76.2	63.2	84.9	70.9	+12.6
	MSCC	**61.8**	**78.6**	**81.9**	**70.3**	**77.5**	**79.0**	**67.8**	**61.7**	**82.3**	**76.1**	**66.9**	**86.3**	**74.2**	+15.9

a) Results in bold indicate the best experimental results in the current transfer task.

Exp. 4: without conceptual consistency, but adding separate heads for auxiliary tasks.

Exp. 5: without conceptual consistency and self-supervised auxiliary tasks, but withholding all of the various transformations for data augmentation.

Exp. 6: without conceptual consistency and self-supervised auxiliary tasks, but withholding all of the various transformations for data augmentation. Moreover, in this experimental setup, we add classifiers for every transform set to predict the main task and then assemble their results.

The last column is the performance loss of the ablation experiment compared with the full MSCC.

According to the result in Table 4, there are some meaningful observations.

(1) Exp. 3 vs. Exp. 7 and Exp. 4 vs. Exp. 7 respectively show the effectiveness of MixUp and the conceptual consistency.

(2) Exp. 4 vs. Exp. 5 reveals that when domain shift is large, the learned intrinsic patterns may be invalid or even bring extra noise, which degrades UDA performance. So the improvement is tiny when adding separate heads for the self-supervised auxiliary tasks but without exploiting the conceptual consistency. It verifies the necessity of conceptual consistency.

(3) Exp. 5 vs. Exp. 6 indicates that when maximizing the agreement of all predictions, the improvement is mainly brought by exploiting the conceptual consistency of multiple self-supervised tasks rather than implicitly assembling multiple predictions.

(4) The degradation of all ablation experiments shows that conceptual consistency is the most vital component in MSCC.

Furthermore, we compare MSCC with the separate head (SH) protocol (adding separate heads for the auxiliary tasks [23]) on different base pseudo labelers. We replace the joint-classifier in MSCC with a separate head for the specific self-supervised auxiliary task to calculate the SH results. And the last column is the performance improvement compared with the original base pseudo labeler.

We analyze the results in Table 5.

(1) MSCC and SH both enhance all base pseudo labelers, which validates our method is effective, and the self-supervised auxiliary tasks could always help the better feature extraction for domain adaptation.

(2) Whatever the base pseudo labeler is, MSCC is better than SH, especially in the crude pseudo labeler, e.g., ResNet50 (source only). It indicates that exploiting the conceptual consistency is a better strategy to utilize self-supervision than adding a separate head for auxiliary tasks.

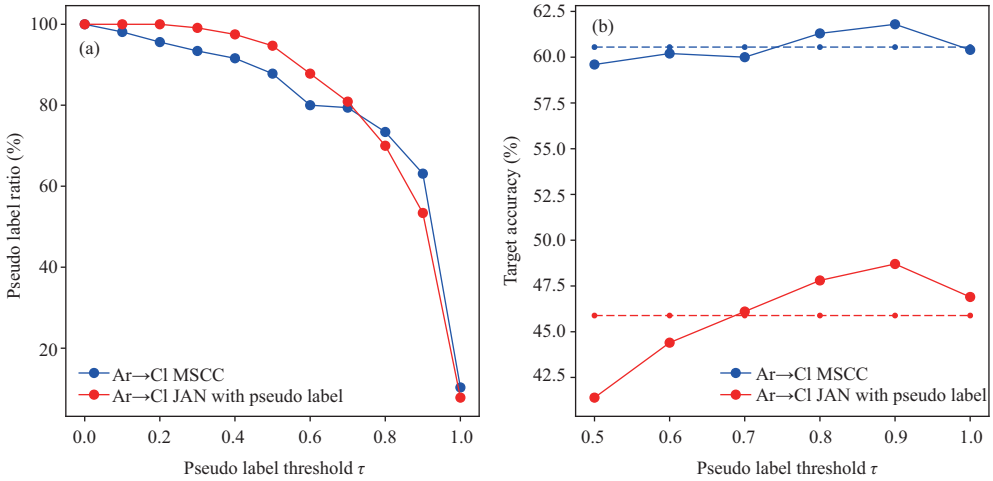

Figure 3 Hyper parameter sensitivity. (a) Pseudo label ratio; (b) target accuracy.

(3) MSCC based on a better pseudo labeler achieves better performance, which illustrates that adopting a better pseudo labeler could improve the performance of MSCC further in the future.

We investigate the effects of the hyper-parameter τ in the pseudo label select function (Eq. 9). Figure 3(a) shows the ratio of pseudo labeled samples in all target samples. According to Figure 3(a), the elbow point on the curve $\tau = 0.9$, should be good choices. To validate this selection, we run JAN with pseudo labels and MSCC, and then illustrate the variation of transfer classification performance as $\tau \in [0.5, 1.0]$. The results are shown in Figure 3(b). We can observe that MSCC is more stable than JAN with various pseudo label threshold τ and $\tau = 0.9$ is a good choice.

In summary, all experimental results on MSCC indicate that every component in MSCC is vital, effective, and stable. Significantly, the strategy of exploiting the conceptual consistency of multiple self-supervised tasks could enhance UDA further, which is a meaningful discovery for UDA.

5 Conclusion

This paper shows that in complex UDA tasks, the source-dominant feature extraction methods perform poorly for extracting discriminative target features. Moreover, we propose a novel method named MSCC to overcome the dominance of source during feature extraction and capture more discriminative intrinsic patterns in target data themselves. We apply multiple transform-based self-supervised auxiliary tasks to capture intrinsic patterns without the target main task annotation. In addition, we exploit the conceptual consistency of different self-supervised tasks from multiple perspectives. Hence, MSCC is a novel method to train auxiliary tasks. Based on the results of our explanation and abundant experiments, our method can correctly extract the target's intrinsic patterns and imply the target's hidden actual ground truth through conceptual consistency. This work has a certain significance for further capturing the target-intrinsic patterns in UDA. In future work, we could use a better base pseudo labeler to improve the ultimate performance. Meanwhile, we can study more valuable intrinsic features from unlabeled target data themselves not dominated by source supervision.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 62076121, 61921006).

References

1 Pan S J, Yang Q. A survey on transfer learning. IEEE Trans Knowl Data Eng, 2009, 22: 1345–1359
2 Dai W Y, Yang Q. Xue G R, et al. Boosting for transfer learning. In: Proceedings of the 24th International Conference on Machine Learning (ICML), 2007. 193–200
3 Raina R, Battle A, Lee H, et al. Self-taught learning: transfer learning from unlabeled data. In: Proceedings of the 24th International Conference on Machine Learning (ICML), 2007. 759–766
4 Lawrence N D, Platt J C. Learning to learn with the informative vector machine. In: Proceedings of the 21st International Conference on Machine Learning (ICML), 2004. 65
5 Mihalkova L, Huynh T, Mooney R J. Mapping and revising Markov logic networks for transfer learning. In: Proceedings of the 22nd AAAI Conference on Artificial Intelligence (AAAI), 2007. 7: 608–614
6 Tzeng E, Hoffman J, Zhang N, et al. Deep domain confusion: maximizing for domain invariance. 2014. ArXiv:1412.3474
7 Long M S, Cao Y, Wang J M, et al. Learning transferable features with deep adaptation networks. In: Proceedings of the 32nd International Conference on Machine Learning (ICML), 2015. 97–105

8 Long M S, Zhu H, Wang J M, et al. Unsupervised domain adaptation with residual transfer networks. In: Proceedings of the Advances in Neural Information Processing Systems 29 (NeurIPS), 2016. 136–144
9 Yaroslav G, Evgeniya U, Hana A, et al. Domain-adversarial training of neural networks. J Mach Learning Res, 2016, 17: 1–35
10 Yaroslav G, Victor L. Unsupervised domain adaptation by backpropagation. In: Proceedings of the 32nd International Conference on Machine Learning (ICML), 2015. 1180–1189
11 Tzeng E, Hoffman J, Saenko K, et al. Adversarial discriminative domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017. 7167–7176
12 Chen C Q, Xie W P, Wen Y, et al. Multiple-source domain adaptation with generative adversarial nets. Knowledge-Based Syst, 2020, 199: 105962
13 Judy H, Eric T, Taesung P, et al. CYCADA: cycle-consistent adversarial domain adaptation. In: Proceedings of the 35th International Conference on Machine Learning (ICML), 2018. 1994–2003
14 Wang Z M, Du B, Tu W P, et al. Incorporating distribution matching into uncertainty for multiple kernel active learning. IEEE Trans Knowl Data Eng, 2019, 33: 128–142
15 Xie S A, Zheng Z B, Chen L, et al. Learning semantic representations for unsupervised domain adaptation. In: Proceedings of the 35th International Conference on Machine Learning (ICML), 2018. 5423–5432
16 Long M S, Cao Z J, Wang J M, et al. Conditional adversarial domain adaptation. In: Proceedings of the Advances in Neural Information Processing Systems 31 (NeurIPS), 2018. 1647–1657
17 Pei Z Y, Cao Z J, Long M S, et al. Multi-adversarial domain adaptation. In: Proceedings of the 32nd AAAI Conference on Artificial Intelligence, 2018. 3934–3941
18 Wang J D, Feng W J, Chen Y Q, et al. Visual domain adaptation with manifold embedded distribution alignment. In: Proceedings of the 26th ACM International Conference on Multimedia (MM), 2018. 402–410
19 Wang Z M, Du B, Guo Y H. Domain adaptation with neural embedding matching. IEEE Trans Neural Netw Learn Syst, 2020, 31: 2387–2397
20 Saito K, Ushiku Y, Harada T, et al. Adversarial dropout regularization. In: Proceedings of the 6th International Conference on Learning Representations (ICLR), 2018
21 Saito K, Watanabe K, Ushiku Y, et al. Maximum classifier discrepancy for unsupervised domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018. 3723–3732
22 Ghifary M, Kleijn W B, Zhang M, et al. Deep reconstruction-classification networks for unsupervised domain adaptation. In: Proceedings of the 14th European Conference on Computer Vision (ECCV), 2016. 597–613
23 Sun Y, Tzeng E, Darrell T, et al. Unsupervised domain adaptation through self-supervision. 2019. ArXiv:1909.11825
24 Zhou Z-H, Zhan D-C, Yang Q. Semi-supervised learning with very few labeled training examples. In: Proceedings of the 22nd AAAI Conference on Artificial Intelligence, 2007. 675–680
25 Ben-David S, Blitzer J, Crammer K, et al. A theory of learning from different domains. Mach Learn, 2010, 79: 151–175
26 Goodfellow I J, Pouget-Abadie J, Mirza M, et al. Generative adversarial networks. 2014. ArXiv:1406.2661
27 Gidaris S, Bursuc A, Komodakis N, et al. Boosting few-shot visual learning with self-supervision. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2019. 8059–8068
28 Lee H, Hwang S J, Shin J. Self-supervised label augmentation via input transformations. In: Proceedings of the 37th International Conference on Machine Learning (ICML), 2020. 5714–5724
29 Kolesnikov A, Zhai X, Beyer L. Revisiting self-supervised visual representation learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019. 1920–1929
30 Jing L L, Tian Y L. Self-supervised visual feature learning with deep neural networks: a survey. IEEE Trans Pattern Anal Mach Intell, 2020, 43: 4037–4058
31 Doersch C, Zisserman A. Multi-task self-supervised visual learning. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017. 2070–2079
32 Kalogeiton V, Weinzaepfel P, Ferrari V, et al. Joint learning of object and action detectors. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017. 4163–4172
33 Tarvainen A, Valpola H. Mean teachers are better role models: weight-averaged consistency targets improve semi-supervised deep learning results. In: Proceedings of the Advances in Neural Information Processing Systems 30 (NeurIPS), 2017. 1195–1204
34 Zou Y, Yu Z D, Liu X F, et al. Confidence regularized self-training. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2019. 5982–5991
35 Wang Q, Breckon T P. Unsupervised domain adaptation via structured prediction based selective pseudo-labeling. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), 2020. 34: 6243–6250
36 Long M S, Zhu H, Wang J M, et al. Deep transfer learning with joint adaptation networks. In: Proceedings of the 34th International Conference on Machine Learning (ICML), 2017. 2208–2217
37 Zhang H Y, Cisse M, Dauphin Y N, et al. MixUp: beyond empirical risk minimization. In: Proceedings of the 6th International Conference on Learning Representations (ICLR), 2018
38 Berthelot D, Carlini N, Goodfellow I, et al. MixMatch: a holistic approach to semi-supervised learning. In: Proceedings of the Advances in Neural Information Processing Systems 32 (NeurIPS), 2019. 5049–5059
39 Jiang J G, Fu B, Long M S. Transfer-learning-library. 2020. https://github.com/thuml/Transfer-Learning-Library
40 He K M, Zhang X Y, Ren S Q, et al. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. 770–778
41 Na J, Jung H, Chang H J, et al. FixBi: bridging domain spaces for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021. 1094–1103
42 Laine S, Aila T. Temporal ensembling for semi-supervised learning. In: Proceedings of the 5th International Conference on Learning Representations (ICLR), 2015. 1–13