

# Selective Random Walk for Transfer Learning in Heterogeneous Label Spaces

**Citation for published version (APA):**

Xiao, Q., Zhang, Y., & Yang, Q. (2024). Selective Random Walk for Transfer Learning in Heterogeneous Label Spaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(6), 4476-4488. Article 10416705. <https://doi.org/10.1109/TPAMI.2024.3360118>

**Document license:**  
TAVERNE

**DOI:**  
[10.1109/TPAMI.2024.3360118](https://doi.org/10.1109/TPAMI.2024.3360118)

**Document status and date:**  
Published: 01/06/2024

**Document Version:**  
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

**Please check the document version of this publication:**

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

**General rights**

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

[www.tue.nl/taverne](http://www.tue.nl/taverne)

**Take down policy**

If you believe that this document breaches copyright please contact us at:

[openaccess@tue.nl](mailto:openaccess@tue.nl)

providing details and we will investigate your claim.

# Selective Random Walk for Transfer Learning in Heterogeneous Label Spaces

Qiao Xiao , Yu Zhang , *Member, IEEE*, and Qiang Yang , *Fellow, IEEE*

**Abstract**—Transfer learning has been widely used in different scenarios, especially in those lacking enough labeled data. However, most of the existing transfer learning methods are based on the assumption that the source and target domains should share the label space entirely or partially, which greatly limits their application scopes. In this article, a Selective Random Walk (SRW) method for transfer learning in heterogeneous label spaces is proposed to make full use of unlabeled auxiliary data, which acts as a bridge for knowledge transfer from the source domain to the target domain. The proposed SRW method can explicitly identify transfer sequences between source and target instances via auxiliary instances based on random walk techniques. Since not all of the transfer sequences generated by random walk are credible for the target task, the SRW method can learn to weight transfer sequences adaptively. Based on the weights of the transfer sequences, the SRW method leverages knowledge by forcing adjacent data points in the transfer sequence to be similar and making the target data point in the sequence represented by other data points in the same sequence. Experiments show that the SRW method outperforms state-of-the-art models in plenty of transfer learning tasks with heterogeneous label spaces constructed within and across several benchmark datasets.

**Index Terms**—Transfer learning, heterogeneous label spaces, selective random walk.

## I. INTRODUCTION

**T**RANSFER learning [1], which has been explored in a wide range of fields, aims to improve the performance of a target domain with few or even no labeled data by exploiting the useful knowledge from a source domain with ample labeled data [2], [3], [4], [5].

Most of the existing transfer learning methods [6], [7], [8] are based on an assumption that the source and target domains should share the same label space and they aim to reduce the distribution discrepancy between domains. However, in real-world

applications, this assumption is hard to be satisfied. To relax the assumption of the identical label space, Partial Domain Adaptation (PDA) methods [9], [10], [11] are proposed for transfer learning under the setting that the source label space is a superset of the target label space. As a more practical scenario, the Open Set Domain Adaptation (OSDA) [12], [13] assumes the target domain contains some classes that do not exist in the source domain. Moreover, under the Universal Domain Adaptation (UDA) setting [14], [15], [16], the source and target label spaces have some overlapping, which however is unknown to the learning model. While all the aforementioned settings suppose that both the well-trained source models and labeled source datasets are available, in some applications, the source datasets may be not accessible, making only the source models and unlabeled target datasets available. Source-free domain adaptation [17], [18], [19] was recently proposed for this challenging scenario, where only a well-trained source model and an unlabeled target dataset are available for adaptation. According to empirical results, those methods only work well when the label spaces between the source and target domains have a large overlap.

The aforementioned transfer learning methods cannot tackle the setting that source and target domains are in heterogeneous label spaces, where the label spaces of the source and target domains have no overlapping, which is a more challenging but more realistic scenario for transfer learning. Moreover, the aforementioned transfer learning methods can hardly achieve good performance, since the distribution discrepancy between two domains is extremely large, resulting in the ‘negative transfer’ phenomenon [1], [20]. To achieve transfer learning in heterogeneous label spaces, several models [21], [22] are proposed. Qi et al. [21] explore the correlations at the instance level between the source and target classes. Luo et al. [22] aim to alleviate the domain shift in heterogeneous label spaces via the semantic transfer by minimizing the entropy of pairwise similarities between the source and target images. Those methods only work when there exists an explicit similarity function between source and target domains, but such similarity function is hard to define precisely in real-world applications, which may limit the use of those methods.

In this article, we propose a transfer learning framework called Selective Random Walk (SRW) under the setting that the source and target domains are in heterogeneous label spaces. Instead of requiring the existence of the explicit similarity function between the source and target domains, an unlabeled auxiliary domain is used in the SRW model to act as a bridge to enable transfer learning between two domains in the heterogeneous

Manuscript received 13 March 2023; revised 6 January 2024; accepted 13 January 2024. Date of publication 30 January 2024; date of current version 7 May 2024. This work was supported in part by the NSFC key under Grant 62136005, in part by the NSFC general under Grant 62076118, and in part by Shenzhen Fundamental Research Program under Grant JCYJ20210324105000003. Recommended for acceptance by Y. Song. (Corresponding author: Yu Zhang.)

Qiao Xiao is with the Department of Mathematics and Computer Science, Eindhoven University of Technology, 5612 AZ Eindhoven, The Netherlands (e-mail: q.xiao@tue.nl).

Yu Zhang is with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China, and also with Peng Cheng Laboratory, Shenzhen 518066, China (e-mail: yu.zhang.ust@gmail.com).

Qiang Yang is with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Kowloon, Hong Kong (e-mail: qyang@cse.ust.hk).

Digital Object Identifier 10.1109/TPAMI.2024.3360118

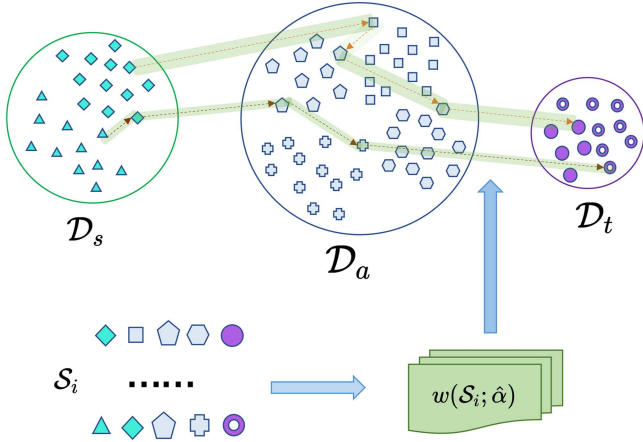


Fig. 1. Transfer learning in heterogeneous label spaces is a challenging setting where the source label space does not intersect with that of the target label space. It can negatively impact the performance if the knowledge is forcefully transferred from the source domain  $\mathcal{D}_s$  to the target domain  $\mathcal{D}_t$ . An auxiliary domain  $\mathcal{D}_a$  is used as a bridge to connect the source and target domains. The proposed SRW model can explicitly generate transfer sequences between the source and target instances through auxiliary instances. By learning the credibility of the transfer sequence  $\mathcal{S}_i$ , a meta weight network  $w$  with parameters  $\hat{\alpha}$  is designed to leverage knowledge from transfer sequences selectively.

label spaces. The unlabeled auxiliary domain is much easier to obtain and we do not impose any assumption on it. To connect the source and target domains via the auxiliary domain, the random walk technique is used in the proposed SRW method to generate transfer sequences. For each transfer sequence generated, adjacent data points in it are enforced to be similar and the target data point in this sequence is assumed to be represented by other data points in the same sequence via a Long Short-Term Memory (LSTM). Since transfer sequences are obtained according to the feature similarity between data, the credibility of each of them is hardly guaranteed. The proposed SRW model selectively leverages knowledge from transfer sequences by learning a meta weight network to weigh them. For illustration, the proposed SRW model is shown in Fig. 1.

In summary, the main contributions of this article are three-fold.

- The SRW model is proposed to handle the setting that the source and target domains are in the heterogeneous label spaces.
- The transfer sequences generated by random walk are selectively used based on weights learned by the proposed meta weight network, which is different from previous studies on the deep random walk that treats different sequences equally.
- Extensive experiments have been conducted on transfer learning tasks in heterogeneous label spaces to validate the effectiveness of the proposed SRW model.

A preliminary version of this work appeared in a conference article [23] published in AAAI 2021. Different from [23] that equally treats sequences generated by random walk, this work designs the SRW method to learn weights for different sequences generated by random walk via meta learning. Moreover, the problem settings of [23] and this work are different. Specifically, [23] assumes that the source and target domains share the

same label space, while this work focuses on the two domains with heterogeneous label spaces. As a result, experiments conducted in this article are completely new and totally different from those in [23].

The rest of this article is organized as follows. Related works are reviewed in Section II. Then we introduce the proposed SRW model in Section III. Section IV analyzes experimental results and Section V concludes the entire article.

## II. RELATED WORKS

In this section, we briefly review related works.

### A. Transfer Learning With Auxiliary Domains

Inspired by the human learning ability that two unrelated concepts could be connected via intermediate concepts that act as a bridge, several methods [23], [24], [25] utilize an auxiliary domain as a bridge to enable transfer learning among two distant domains. Specifically, Tan et al. [24] propose a transitive transfer learning method to transfer knowledge between distant domains with an auxiliary domain that contains co-occurrence data of the source and target domains. To relax such requirements on the auxiliary domain, Tan et al. [25] propose a method to select useful data in the auxiliary domain based on the reconstruction error. Xiao and Zhang [23] propose to use the auxiliary domain as a bridge for transfer learning between two distant domains. Different from the aforementioned methods which have the same label space in the source and target domains, the proposed SRW model works under the more challenging setting that the source and target domains are in heterogeneous label spaces.

### B. Random Walk for Transfer Learning

Random walk could produce sequences of nodes in a graph [26] and it has been adopted in transfer learning methods such as [23], [27], [28] which assume that the label spaces of the source and target domains are identical. Ng et al. [27] use the random walk to propagate ranking scores of labeled data in the source domain to unlabeled data in the target domain based on a joint transition probability graph. Tan et al. [28] use the random walk to transfer information according to a transition probability matrix between the source and target domains. Different from those studies that use the random walk to calculate the probabilities of traversing between instances in two domains, the proposed SRW model uses random walk to generate transfer sequences between the source and target domains via auxiliary domains. Moreover, different from [23] that treats transfer sequences equally, the knowledge of transfer sequences is leveraged selectively in the proposed SRW model according to the weights learned by the proposed meta weight network.

### C. Meta Learning

Inspired by the recent advance of meta learning [29], [30], [31], [32], various adaptive data weighting schemes have been proposed to improve the performance. [33], [34], [35] propose meta-learning approaches based on weight functions such as

Bayesian function approximators and deep neural networks with the attention mechanism. However, most of those methods only operate on the training loss of all the training data. Recently, [36], [37] proposed meta networks on training data and obtain good performance for imbalance or noisy datasets. Different from them, the proposed meta weight network learns to assign weights to each sequence with variable lengths, which expands the use of meta learning to more complex data.

### III. THE SRW MODEL

In this section, we introduce the proposed SRW model.

#### A. Problem Settings

Given a source domain and a target domain, the source domain has a large labeled training dataset  $\mathcal{D}_s = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^{n_s}$ , where  $y_i^s \in \mathcal{Y}^S$  is the class label of the  $i$ th data point  $\mathbf{x}_i^s$  in the source domain and  $n_s$  denotes the number of data points in the source domain, and the target domain has a small labeled training dataset  $\mathcal{D}_t = \{(\mathbf{x}_i^t, y_i^t)\}_{i=1}^{n_t}$ , where  $y_i^t \in \mathcal{Y}^T$  is the class label of the  $i$ th data point  $\mathbf{x}_i^t$  in the target domain and  $n_t$  denotes the number of data points in the target domain. Here we assume  $n_s \gg n_t$ . Since the source and target domains have a large discrepancy due to the heterogeneous label spaces in the two domains, the direct transfer from the source domain to the target domain may have no effect or even a negative effect on improving the performance of the target domain. Therefore, we assume that there is an unlabeled auxiliary dataset  $\mathcal{D}_a = \{\mathbf{x}_1^a, \dots, \mathbf{x}_{n_a}^a\}$ , where  $n_a \gg n_t$  denotes the number of data points in the auxiliary domain. This auxiliary domain contains data points from diverse domains and it will act as a bridge to help transfer the knowledge from the source domain to the target domain.

Unlike previous transfer learning methods which assume partial or fully shared label spaces (i.e.,  $\mathcal{Y}^S = \mathcal{Y}^T$  or  $\mathcal{Y}^S \cap \mathcal{Y}^T \neq \emptyset$ ), we can also tackle transfer learning problems in heterogeneous label spaces, where the source and target label spaces are non-overlapping, i.e.,  $\mathcal{Y}^S \cap \mathcal{Y}^T = \emptyset$ .

#### B. Graph Construction

In the proposed SRW method, we first learn a hidden representation for all the data in the source, auxiliary, and target domains as  $\hat{\mathbf{x}}_i^* = f(\mathbf{x}_i^*; \phi)$ , where  $f(\cdot)$  denotes a feature extraction network with the parameters  $\phi$  and the superscript  $*$  in  $\mathbf{x}_i^*$  can be  $s$ ,  $a$  or  $t$ .

To measure similarities between data points in a mini-batch, we use all the data points to construct a graph  $G$ , in which each node corresponds to a data point, and edge weights in the graph are defined as

$$\begin{aligned} e(\hat{\mathbf{x}}, \hat{\mathbf{x}}) &= 0 \quad \forall \mathbf{x} \in \mathcal{D}_s \cup \mathcal{D}_a \cup \mathcal{D}_t \\ e(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) &= \exp\{\langle \hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2 \rangle\} \quad \forall \mathbf{x}_1, \mathbf{x}_2 \in \mathcal{D}_s \cup \mathcal{D}_a \\ e(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2) &= \exp\{\langle \hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2 \rangle\} \quad \forall \mathbf{x}_1, \mathbf{x}_2 \in \mathcal{D}_t \cup \mathcal{D}_a, \end{aligned} \quad (1)$$

where  $\hat{\mathbf{x}} = f(\mathbf{x}; \phi)$ ,  $\hat{\mathbf{x}}_1 = f(\mathbf{x}_1; \phi)$ ,  $\hat{\mathbf{x}}_2 = f(\mathbf{x}_2; \phi)$ ,  $\cup$  denotes the union of domains, and  $\langle \cdot, \cdot \rangle$  denotes the inner product between two vectors, matrices, or tensors of the same size. According to (1), there is no self-loop in this graph and there is no

edge between the source and target samples because of the large discrepancy between two domains with heterogeneous label spaces, resulting in unreliable similarities. During the training process,  $\phi$  changes over epochs and so do edge weights.

Based on the graph  $G$ , the SRW method conducts a random walk in two directions. The first kind of random walk will randomly select a node corresponding to a data point in the source domain as the starting point, and then visit the next data point with probability proportional to  $e(i, j)$  until it reaches a node in the target domain or the number of nodes visited exceeds a threshold  $T$ . Similarly, the second type of random walk starts with a node in the target domain and stops when it reaches a node in the source domain or the number of nodes visited exceeds  $T$ . A sequence of data points obtained in one execution of a random walk is called a *transfer sequence*, which could define a transition between the source and target domains.

#### C. Loss Functions

We can obtain several transfer sequences by the random walk technique mentioned in the previous section, where the  $i$ th sequence is denoted by  $\mathcal{S}_i = (\hat{\mathbf{x}}_{i,1}, \dots, \hat{\mathbf{x}}_{i,n_{s_i}})$  with  $n_{s_i}$  denoting the length of this sequence or equivalently the number of nodes visited satisfying  $n_{s_i} \leq T$ .

For the sequence  $\mathcal{S}_i$ , we expect two neighboring data points in it to be similar and define the corresponding similarity loss as

$$\begin{aligned} \ell_1(\mathcal{S}_i; \theta) &= \sum_{j=1}^{n_{s_i}-1} (-\ln \sigma(\langle \hat{\mathbf{x}}_{i,j}, \hat{\mathbf{x}}_{i,j+1} \rangle) - \ln(1 - \sigma(\langle \hat{\mathbf{x}}_{i,j}, \hat{\mathbf{z}}_{i,j} \rangle))), \end{aligned} \quad (2)$$

where  $\sigma(x) = \frac{1}{1+\exp\{-x\}}$  denotes the sigmoid function and  $\hat{\mathbf{z}}_{i,j}$  is sampled out of  $\mathcal{S}_i$  to act as a dissimilar data point to  $\hat{\mathbf{x}}_{i,j}$ .

For  $\mathcal{S}_i$ , suppose the last data point is from the target domain, which means that the random walk finds a path from the source domain to the target domain, and we expect that the last data point can be reconstructed from other data points in this sequence. So we can formulate the reconstruction loss as

$$\ell_2(\mathcal{S}_i; \theta) = \|\hat{\mathbf{x}}_{i,n_{s_i}} - f_d(\text{LSTM}(\hat{\mathbf{x}}_{i,1}, \dots, \hat{\mathbf{x}}_{i,n_{s_i}-1}))\|, \quad (3)$$

where  $\text{LSTM}(\cdot)$  denotes an LSTM to output the hidden state of the last position,  $f_d(\cdot)$  is a neural decoder to generate an approximation of  $\hat{\mathbf{x}}_{i,n_{s_i}}$ , and  $\|\cdot\|$  denotes the  $\ell_2$  norm. Here the LSTM is used since  $\hat{\mathbf{x}}_{i,1}, \dots, \hat{\mathbf{x}}_{i,n_{s_i}-1}$  form a sequence.

Moreover, based on the labeled data in  $\mathcal{D}_s$  and  $\mathcal{D}_t$ , we formulate the classification loss as

$$\begin{aligned} L_{cls}(\mathcal{D}_s, \mathcal{D}_t; \theta) &= \frac{1}{|\mathcal{D}_s|} \sum_{(\hat{\mathbf{x}}, y) \in \mathcal{D}_s} \ell(f_{c_s}(\hat{\mathbf{x}}), y) + \frac{1}{|\mathcal{D}_t|} \sum_{(\hat{\mathbf{x}}, y) \in \mathcal{D}_t} \ell(f_{c_t}(\hat{\mathbf{x}}), y), \end{aligned} \quad (4)$$

where  $f_{c_s}(\cdot)$  and  $f_{c_t}(\cdot)$  denote the classification network of the source and target domains, respectively, and  $|\cdot|$  denotes the number of data points in the corresponding dataset, and  $\ell(\cdot, \cdot)$  denotes the cross-entropy loss. Due to the heterogeneous

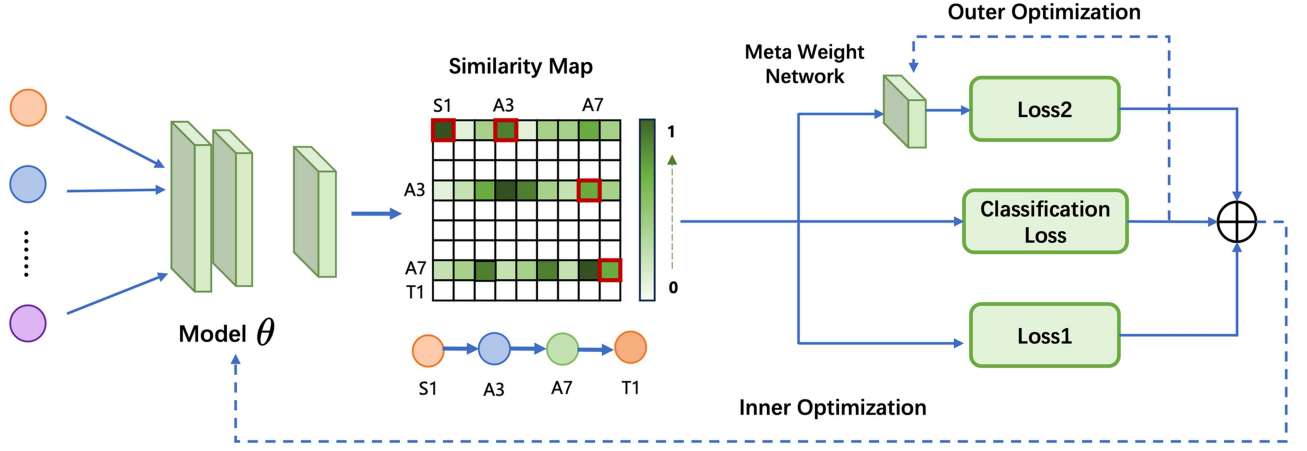


Fig. 2. Architecture of the proposed SRW model. According to the hidden feature representation generated by  $\theta$ , we construct a graph  $G$  on data points in a mini-batch and adopt the random walk to generate transfer sequences based on a similarity map. For example,  $\mathcal{S}_i = S1 \rightarrow A3 \rightarrow A7 \rightarrow T1$  is a transfer sequence from the source domain to the target domain via the auxiliary domain. The dotted lines represent the bi-level optimization process for the meta weight network  $w(\mathcal{S}_i; \alpha)$ . Based on the learned meta weight network, the SRW model can be optimized by Loss2 selectively as well as the other two loss functions.

label spaces in the source and target domains, two different classification networks (i.e.,  $f_{c_s}(\cdot)$  and  $f_{c_t}(\cdot)$ ) should be used.

Since transfer sequences obtained by the random walk are based on the similarity between node features, it is difficult to guarantee the credibility of each transfer sequence. We expect to leverage knowledge from transfer sequences selectively according to the specific target tasks. Hence, we will design a meta weight network  $w(\mathcal{S}_i; \alpha)$  parameterized by  $\alpha$  to learn the weight of each transfer sequence automatically with the hope of assigning large weights to trustworthy transfer sequences. When given the meta weight network  $w(\mathcal{S}_i; \hat{\alpha})$ , where  $\hat{\alpha}$  denotes the learned value for  $\alpha$ , the objective function by combining the above loss functions is formulated as

$$\begin{aligned} \min_{\theta} L_{cls}(\mathcal{D}_s, \mathcal{D}_t; \theta) &+ \frac{\lambda}{l_{st}} \sum_{i=1}^{l_{st}} \ell_2(\mathcal{S}_i; \theta) \\ &+ \frac{\gamma}{l} \sum_{i=1}^l w(\mathcal{S}_i; \hat{\alpha}) \ell_1(\mathcal{S}_i; \theta), \end{aligned} \quad (5)$$

where  $\theta$  denotes the set of all the parameters in the SRW method except those in the meta weight network,  $l$  denotes the number of all the transfer sequences generated by the random walk technique in both directions and  $l_{st}$  denotes the number of transfer sequences which start from a source data point and end with a target data point.  $\lambda$  and  $\gamma$  in problem (5) are the trade-off parameters to balance the three terms. The last term in problem (5) is to weigh similarity losses of different transfer sequences according to weights produced by the meta weight network and it can help utilize transfer sequences selectively. The entire architecture of the proposed SRW model is shown in Fig. 2.

#### D. Meta Weight Network

To obtain  $w(\mathcal{S}_i; \hat{\alpha})$  in an adaptive and reliable way, we will learn  $\alpha$  in a meta learning manner. To achieve that, we split the target dataset into two subsets as  $\mathcal{D}_t = \mathcal{D}_t^{tr} \cup \mathcal{D}_t^{val}$  for the

training and validation, and the meta weight network can be defined as

$$w(\mathcal{S}_i^{tr}; \alpha) = f_m(\text{LSTM}(\mathcal{S}_i^{tr})), \quad (6)$$

where  $\mathcal{S}_i^{tr}$  denotes the  $i$ th transfer sequence generated from the graph constructed by  $\mathcal{D}_s$ ,  $\mathcal{D}_a$ , and  $\mathcal{D}_t^{tr}$ , and  $f_m(\cdot)$  is a neural decoder to generate the weight for each  $\mathcal{S}_i^{tr}$ . Here the LSTM is used since the length of  $\mathcal{S}_i^{tr}$  is varying and it is different from the LSTM used in (3). The architecture of the meta weight network is shown in Fig. 3.

We expect the meta weight network to perform well on  $\mathcal{D}_t^{val}$  by minimizing the loss as

$$\frac{1}{|\mathcal{D}_t^{val}|} \sum_{(\hat{\mathbf{x}}^{val}, y^{val}) \in \mathcal{D}_t^{val}} \ell(f_{c_t}(\hat{\mathbf{x}}^{val}), y^{val}). \quad (7)$$

Taking problem (5) and (7) into consideration, the objective function of the SRW model can be formulated as

$$\begin{aligned} \min_{\alpha} \mathcal{L}^{outer}(\theta, \alpha) &= \frac{1}{|\mathcal{D}_t^{val}|} \sum_{(\hat{\mathbf{x}}^{val}, y^{val}) \in \mathcal{D}_t^{val}} \ell(f_{c_t}(\hat{\mathbf{x}}^{val}), y^{val}) \\ \text{s.t. } \hat{\theta} &= \arg \min_{\theta} \mathcal{L}^{inner}(\theta, \alpha) \\ \mathcal{L}^{inner}(\theta, \alpha) &\equiv L_{cls}(\mathcal{D}_s, \mathcal{D}_t^{tr}; \theta) + \frac{\lambda}{l_{st}^{tr}} \sum_{i=1}^{l_{st}^{tr}} \ell_2(\mathcal{S}_i^{tr}; \theta) \\ &+ \frac{\gamma}{l_{tr}} \sum_{i=1}^{l_{tr}} w(\mathcal{S}_i^{tr}; \alpha) \ell_1(\mathcal{S}_i^{tr}; \theta) \end{aligned} \quad (8)$$

where  $l_{tr}$  denotes the number of transfer sequences generated from the graph  $G^{tr}$  constructed by  $\mathcal{D}_s$ ,  $\mathcal{D}_a$ , and  $\mathcal{D}_t^{tr}$ , and  $l_{st}^{tr}$  denotes the number of transfer sequences, which start from a source data point and end with a target data point, based on  $G^{tr}$ . Problem (8) is a bi-level optimization problem [38], which requires solving two nested problems. The inner-level problem (i.e.,  $\mathcal{L}^{inner}(\theta, \alpha)$ ) is to find optimal  $\theta$  by minimizing

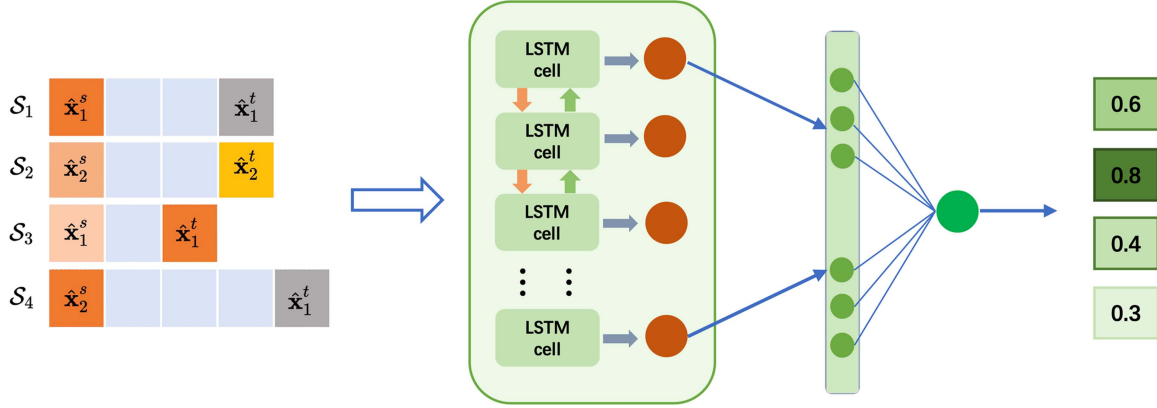


Fig. 3. Architecture of the meta weight network. Here  $S_i$  denotes the  $i$ th transfer sequence generated from the graph constructed by  $\mathcal{D}_s$ ,  $\mathcal{D}_a$ , and  $\mathcal{D}_t^{tr}$ . The variable-length transfer sequences are processed in the meta weight network which is constructed by LSTM and fully connected layer with the sigmoid function to produce the weight for each sequence. For example, we can get four weights, each of which is within (0, 1), for transfer sequences  $S_1, S_2, S_3, S_4$ , respectively.

---

**Algorithm 1: SRW Model.**


---

**INPUT:**  $\mathcal{D}_t^{tr}$ ,  $\mathcal{D}_t^{val}$ ,  $\mathcal{D}_s$  and  $\mathcal{D}_a$ , max iterations  $I$ .

**OUTPUT:** optimal parameters  $\hat{\alpha}$  and  $\hat{\theta}$ .

Initialize parameter  $\alpha_0$  and  $\theta_0$ ;

**while**  $i < I - 1$  **do**

$\{x, y\} \leftarrow \text{SampleMiniBatch}(\mathcal{D}_t^{tr}, \mathcal{D}_s, \mathcal{D}_a)$ ;

$\{x^{(val)}, y^{(val)}\} \leftarrow \text{SampleMiniBatch}(\mathcal{D}_t^{val})$ ;

    Compute  $\mathcal{L}^{inner}(\theta_t, \alpha_t)$ ;

    Update  $\theta_{t+1}$  by (9);

    Compute  $\mathcal{L}^{outer}(\theta_{t+1}, \alpha_t)$ ;

    Update  $\alpha_{t+1}$  by (10);

**end while**

---

the weighted empirical risk, while the outer-level problem (i.e.,  $\mathcal{L}^{outer}(\theta, \alpha)$ ) is to minimize the classification loss w.r.t  $\alpha$ .

To meet the efficiency requirement, we adopt an online approximation method proposed in [37], [39] for the bi-level optimization and employ the Stochastic Gradient Descent (SGD) technique for the optimization.

*Updating  $\theta$ .* Given the meta weight network  $w$  parameterized by  $\alpha_t$ ,  $\theta_{t+1}$  can be obtained by minimizing the inner-level problem via SGD as

$$\theta_{t+1} = \theta_t - \eta_\theta \nabla_\theta \mathcal{L}^{inner}(\theta_t, \alpha_t), \quad (9)$$

where  $\eta_\theta$  denotes the learning rate.

*Updating  $\alpha$ .* After obtaining  $\theta_{t+1}$ , the outer-level problem can be solved by SGD as

$$\alpha_{t+1} = \alpha_t - \eta_\alpha \nabla_\alpha \mathcal{L}^{outer}(\theta_{t+1}, \alpha_t), \quad (10)$$

where  $\eta_\alpha$  denotes the learning rate.

The entire algorithm for the SRW model to solve the problem (8) is summarized in Algorithm 1.

After solving the problem (8), we fix the meta weight network and further update  $\theta$  by solving the problem (5) based on  $\mathcal{D}_s$  and  $\mathcal{D}_t$ .

#### IV. EXPERIMENTS

In this section, we empirically evaluate the proposed SRW method.

##### A. Datasets

Experiments are conducted on three benchmark datasets, including the Animals with Attributes (AwA) dataset [40], the Caltech-101 dataset [41], and the MiniImagenet dataset [30]. The RGB channels of all images are centered and scaled, and all the images are cropped centrally to the size of  $224 \times 224$ .

The source domain, auxiliary domain and target domain are constructed based on the above three datasets. To better test the performance of the proposed SRW method, we consider three scenarios: transfer within the same dataset, transfer across datasets, and transfer from various styles to real-world images, which correspond to three ways to construct different domains as discussed in the following.

For transfer tasks within a dataset, we construct the source, auxiliary and target domains on each dataset, respectively. For the AwA dataset, we randomly select 8 categories as the source domain and another 5 categories as the target domain. The remaining categories are used as the auxiliary domain. In this way, we construct 5 transfer tasks (denoted by *task01* to *task05*), where the categories contained in the source and target domains are shown in Table I. For the Caltech-101 dataset, similarly, 8 categories are randomly selected as the source domain, another 5 categories are randomly selected as the target domain, and the remaining categories are used as the auxiliary domain. Specifically, we construct 5 transfer tasks (denoted by *task06* to *task10*), where the categories used in the source and target domains are shown in Table II. For the MiniImagenet dataset, all the 60,000 images are used to construct the three domains. In each experimental configuration, 7 categories are selected randomly as the source domain and 5 categories for the target domain without overlapping. Data in the remaining categories are used as the auxiliary domain. Based on such construction, we construct 14 transfer tasks (denoted by *task11* to *task24*),

TABLE I  
DETAILS FOR TASKS WITHIN THE AWA DATASET

| taskid        | source domain                                                                 | target domain                                    |
|---------------|-------------------------------------------------------------------------------|--------------------------------------------------|
| <i>task01</i> | dolphin, hamster, cow, dalmatian, persian+cat, chihuahua, leopard, raccoon    | wolf, siamese+cat, giant+panda, chimpanzee, deer |
| <i>task02</i> | rat, mole, ox, pig, buffalo, moose, giraffe, humpback+whale                   | skunk, tiger, hamster, deer, persian+cat         |
| <i>task03</i> | grizzly+bear, pig, gorilla, hippopotamus, buffalo, lion, giant+panda, giraffe | skunk, tiger, hamster, deer, persian+cat         |
| <i>task04</i> | walrus, deer, rat, polar+bear, squirrel, mouse, buffalo, giraffe              | raccoon, tiger, rabbit, antelope, fox            |
| <i>task05</i> | elephant, bat, dalmatian, buffalo, tiger, horse, weasel, persian+cat          | mouse, giant+panda, skunk, polar+bear, chihuahua |

TABLE II  
DETAILS FOR TASKS WITHIN THE CALTECH-101 DATASET

| taskid        | source domain                                                                                 | target domain                                                      |
|---------------|-----------------------------------------------------------------------------------------------|--------------------------------------------------------------------|
| <i>task06</i> | cormorant, hot-air-balloon, touring-bike, vcr, dog, cake, paper-shredder, birdbath            | jesus-christ, spaghetti, roulette-wheel, rainbow, fern             |
| <i>task07</i> | kayak, giraffe, coin, tower-pisa, american-flag, football-helmet, blimp, segway               | tomato, license-plate, guitar-pick, porcupine, tennis-racket       |
| <i>task08</i> | wine-bottle, giraffe, knife, chandelier-101, coffin, dog, washing-machine, vcr                | tomato, license-plate, guitar-pick, porcupine, tennis-racket       |
| <i>task09</i> | microscope, backpack, computer-monitor, toad, yarmulke, fern, hot-air-balloon, bonsai-101     | tower-pisa, smokestack, diamond-ring, license-plate, american-flag |
| <i>task10</i> | galaxy, hummingbird, ibis-101, joy-stick, umbrella-101, screwdriver, speed-boat, picnic-table | tower-pisa, smokestack, diamond-ring, license-plate, american-flag |

TABLE III  
DETAILS FOR TASKS WITHIN THE MINIIMAGENET DATASET

| taskid        | source domain                                                                                  | target domain                                                      |
|---------------|------------------------------------------------------------------------------------------------|--------------------------------------------------------------------|
| <i>task11</i> | frying-pan, stage, french-bulldog, miniskirt, garbage-truck, iPod, vase                        | komondor, school-bus, pencil-box, organ, ant                       |
| <i>task12</i> | goose, carousel, king-crab, ant, golden-retriever, tile-roof, black-footed-ferret              | tibetan-mastiff, harvestman, horizontal-bar, rock-beauty, trifle   |
| <i>task13</i> | green-mamba, carton, toucan, solar-dish, chime, carousel, snorkel                              | tibetan-mastiff, harvestman, horizontal-bar, rock-beauty, trifle   |
| <i>task14</i> | bookshop, garbage-truck, hotdog, ibizan-hound, ashcan, arctic-fox, robin                       | ear, snorkel, tile-roof, theater-curtain, holster                  |
| <i>task15</i> | hair-slide, mixing-bowl, pencil-box, missile, vase, upright, tibetan-mastiff                   | ear, snorkel, tile-roof, theater-curtain, holster                  |
| <i>task16</i> | yawl, gordon-setter, horizontal-bar, scoreboard, saluki, bookshop, nematode                    | ear, snorkel, tile-roof, theater-curtain, holster                  |
| <i>task17</i> | electric-guitar, missile, african-hunting-dog, boxer, theater-curtain, arctic-fox, mixing-bowl | spider-web, ibizan-hound, rock-beauty, house-finch, komondor       |
| <i>task18</i> | trifle, wok, unicycle, nematode, street-sign, barrel, solar-dish                               | spider-web, ibizan-hound, rock-beauty, house-finch, komondor       |
| <i>task19</i> | combination-lock, file, slot, bookshop, toucan, dugong, coral-reef                             | carousel, rock-beauty, consomme, house-finch, ant                  |
| <i>task20</i> | tank, slot, green-mamba, cliff, saluki, vase, orange                                           | cannon, arctic-fox, miniature-poodle, tibetan-mastiff, photocopier |
| <i>task21</i> | organ, garbage-truck, combination-lock, school-bus, coral-reef, cannon, fire-screen            | tile-roof, frying-pan, worm-fence, pencil-box, dome                |
| <i>task22</i> | combination-lock, file, slot, bookshop, toucan, dugong, coral-reef                             | carousel, rock-beauty, consomme, house-finch, ant                  |
| <i>task23</i> | newfoundland, robin, white-wolf, organ, aircraft-carrier, ladybug, cuirass                     | carousel, rock-beauty, consomme, house-finch, ant                  |
| <i>task24</i> | yawl, miniskirt, tobacco-shop, newfoundland, orange, bolete, missile                           | spider-web, ibizan-hound, rock-beauty, house-finch, komondor       |

where the categories used in the source and target domains are shown in Table III.

For transfer tasks across datasets, two of these datasets are used as the source and target domains, respectively, while the rest one is acting as the auxiliary domain. In detail, the MiniImagenet dataset (denoted by **M**) consists of 100 categories with 600 instances for each category. The AWA dataset (denoted

by **A**) contains 30,475 pictures in 51 categories, where the number of instances per class varies from 92 to 1,168. The Caltech-101 dataset (denoted by **C**) contains 8,677 images with 101 categories and we remove the background category. To make a thorough evaluation, we evaluate all methods on all six transfer tasks:  $C \rightarrow A$ ,  $A \rightarrow C$ ,  $M \rightarrow A$ ,  $A \rightarrow M$ ,  $M \rightarrow C$ , and  $C \rightarrow M$ .

For transfer tasks involving various styles transitioning to real-world images, we assess the proposed model using the PACS and DomainNet datasets. The PACS dataset comprises four domains: Photo, Art Painting, Cartoon, and Sketch. For the PACS dataset, the Photo domain is treated as the target domain. We randomly select one of the remaining domains as the source domain and use the others as auxiliary domains. This results in three transfer tasks:  $C \rightarrow P$ ,  $S \rightarrow P$ , and  $A \rightarrow P$ . The DomainNet dataset encompasses images from six distinct domains: Real images, Paintings, Clipart, QuickDraw images, Infographics, and Sketches. The dataset holds between 48 K to 172 K images per domain and has 600 K images in total. In our experiments, the Real image is selected as the target domain. We then randomly choose another domain as the source and designate the rest as auxiliary domains. Consequently, we establish five transfer tasks:  $P \rightarrow R$ ,  $Q \rightarrow R$ ,  $C \rightarrow R$ ,  $S \rightarrow R$ , and  $I \rightarrow R$ .

### B. Experimental Settings

For a fair comparison, the same architectures for the feature extractor and classifier are used for all deep methods. For efficiency, we use the AlexNet model [42] pre-trained on the ImageNet-1 K dataset [43] as the backbone network for transfer tasks across datasets and styles transfer. For transfer tasks within a dataset, without using information from other large-scale datasets (e.g., ImageNet-21K [44]), we adopt the RotNet [45] to pre-train an AlexNet model on each dataset as the backbone network, as RotNet is a self-supervised method and can be integrated into any pipeline easily. The RotNet is trained by the SGD method for 160 epochs with a learning rate as 0.001. Note that during the pre-training of the RotNet, the samples from target categories are not used.

For each task, the backbone network is followed by a feature extraction network, which has a Fully-Connected (FC) layer with 256 hidden units. The same network architecture is used for all baseline methods. In the proposed SRW model, a one-layer bi-directional LSTM with 128 hidden units and a neural decoder  $f_d(\cdot)$  that has an FC layer with 256 outputs are used to compute the reconstruction loss defined in (3). The classification network has two branches for the source and target domains, where each branch has an FC layer with various outputs depending on the number of classes in the corresponding domain. The meta weight network is constructed by a one-layer bi-directional LSTM with 128 hidden units followed by an FC layer  $f_m(\cdot)$ , which has one output with the sigmoid function. So the weight of each sequence learned by the meta weight network is within [0, 1].

The mini-batch SGD with a Nestorov momentum of 0.9 is used to solve problem (8). The batch size is set to 100, including 25, 25, and 50 images for the source, target and auxiliary domains, respectively. The initial learning rates for the inner and outer-level problems are set to 0.01. The threshold  $T$  in the random walk technique is set to 6. The hyper-parameters  $\lambda$  and  $\gamma$  in the inner-level problem of problem (8) are set to 1.0 and 0.3, respectively.

We make a comprehensive comparison with different transfer learning methods. The baseline methods in the comparison include

TABLE IV  
PERFORMANCE OF DIFFERENT MODELS ON DIFFERENT TRANSFER TASKS  
ACROSS DATASETS

| Method                  | C→A                         | A→C                         | M→A                         | A→M                         | M→C                         | C→M                         | Avg         |
|-------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-------------|
| DNN $\mathcal{D}_t$     | 55.9 <sub>(0.1)</sub>       | 70.8 <sub>(0.2)</sub>       | 55.9 <sub>(0.1)</sub>       | 50.7 <sub>(0.2)</sub>       | 70.8 <sub>(0.2)</sub>       | 50.7 <sub>(0.2)</sub>       | 59.1        |
| Fine-tune               | 50.8 <sub>(0.3)</sub>       | 66.1 <sub>(0.3)</sub>       | 55.4 <sub>(0.4)</sub>       | 47.6 <sub>(0.5)</sub>       | 69.2 <sub>(0.3)</sub>       | 46.6 <sub>(0.4)</sub>       | 55.9        |
| DNN $\mathcal{D}_{s,t}$ | 54.5 <sub>(0.3)</sub>       | 70.9 <sub>(0.5)</sub>       | 57.4 <sub>(0.4)</sub>       | 51.3 <sub>(0.4)</sub>       | 71.2 <sub>(0.3)</sub>       | 50.2 <sub>(0.2)</sub>       | 59.2        |
| DAN                     | 54.6 <sub>(0.2)</sub>       | 69.7 <sub>(0.2)</sub>       | 55.8 <sub>(0.4)</sub>       | 50.8 <sub>(0.4)</sub>       | 70.0 <sub>(0.5)</sub>       | 49.5 <sub>(0.3)</sub>       | 58.4        |
| DANN                    | 54.5 <sub>(0.2)</sub>       | 67.5 <sub>(0.5)</sub>       | 55.2 <sub>(0.4)</sub>       | 49.1 <sub>(0.3)</sub>       | 68.3 <sub>(0.3)</sub>       | 48.7 <sub>(0.5)</sub>       | 57.2        |
| ETN                     | 54.7 <sub>(0.4)</sub>       | 69.2 <sub>(0.3)</sub>       | 57.2 <sub>(0.3)</sub>       | 51.2 <sub>(0.4)</sub>       | 70.3 <sub>(0.5)</sub>       | 49.6 <sub>(0.3)</sub>       | 58.7        |
| LEL                     | 54.6 <sub>(0.3)</sub>       | 69.6 <sub>(0.3)</sub>       | 56.5 <sub>(0.4)</sub>       | 51.6 <sub>(0.5)</sub>       | 69.8 <sub>(0.4)</sub>       | 49.5 <sub>(0.4)</sub>       | 58.6        |
| DFF                     | 53.3 <sub>(0.5)</sub>       | 69.3 <sub>(0.4)</sub>       | 56.2 <sub>(0.3)</sub>       | 49.8 <sub>(0.4)</sub>       | 69.7 <sub>(0.3)</sub>       | 58.2 <sub>(0.4)</sub>       | 59.4        |
| <b>SRW</b>              | <b>58.1<sub>(0.3)</sub></b> | <b>72.4<sub>(0.4)</sub></b> | <b>59.5<sub>(0.4)</sub></b> | <b>54.0<sub>(0.5)</sub></b> | <b>73.1<sub>(0.3)</sub></b> | <b>54.5<sub>(0.4)</sub></b> | <b>61.9</b> |

The best results are in bold.

*DNN $\mathcal{D}_t$* : The DNN $\mathcal{D}_t$  method adopts the same backbone network as the SRW model and is trained on  $\mathcal{D}_t$  only.

*Fine-tune*: The architecture of this method is the same as the DNN $\mathcal{D}_t$  method. It is pre-trained on  $\mathcal{D}_s$  and then fine-tuned on  $\mathcal{D}_t$ .

*DNN $\mathcal{D}_{s,t}$* : Different from the DNN $\mathcal{D}_t$  method, the DNN $\mathcal{D}_{s,t}$  method is trained on both  $\mathcal{D}_s$  and  $\mathcal{D}_t$ , and has separate classification network branches for  $\mathcal{D}_s$  and  $\mathcal{D}_t$ .

*DAN* [46]: The DAN method uses maximum mean discrepancy for the distribution alignment with separate classification network branches for  $\mathcal{D}_s$  and  $\mathcal{D}_t$ .

*DANN* [47]: The DANN method is an adversarial deep method, which trains a discriminator through the adversarial learning strategy.

*ETN* [11]: The ETN method is designed for PDA.

*LEL* [22]: The LEL method is a transfer learning method under the setting that the source and target domains possess heterogeneous label spaces.

*DFF* [48]: The DFF method is a feature-based distant transfer learning approach comprised of feature fusion, and it can be extended to multi-class image classification tasks.

For the evaluation of transfer tasks across datasets, we randomly select 10 labeled instances of each class in the target domain as the training dataset  $\mathcal{D}_t$  and the rest is for testing. For the evaluation of transfer tasks within a dataset, we randomly select 30 labeled instances of each class in the target domain as the training dataset  $\mathcal{D}_t$  and the rest is for testing. For training the meta weight network  $w$ , the entire training dataset in the target domain is randomly divided into two parts (i.e.,  $\mathcal{D}_t^{tr}$  and  $\mathcal{D}_t^{val}$ ) with the ratio of 3:2 for tasks within the dataset. For tasks across the dataset, we do not perform this split, as the target domain is small. After learning the meta weight network, the entire training dataset is used for training the SRW model further.

### C. Results

Each setting was repeated for three times, and the results are reported in Tables IV, VI, VII, VIII, and IX in the format of ‘mean (standard deviation)’ and Fig. 4 that shows the the results on *task21* to *task24* for better presentation. The results show that the DNN $\mathcal{D}_t$  method slightly outperforms the Fine-tune and DNN $\mathcal{D}_{s,t}$  methods in most cases, which implies that the information learned from the source domain may have a negative impact on the performance of the target domain due to the domain shift caused by heterogeneous label spaces. Additionally, the DNN $\mathcal{D}_t$

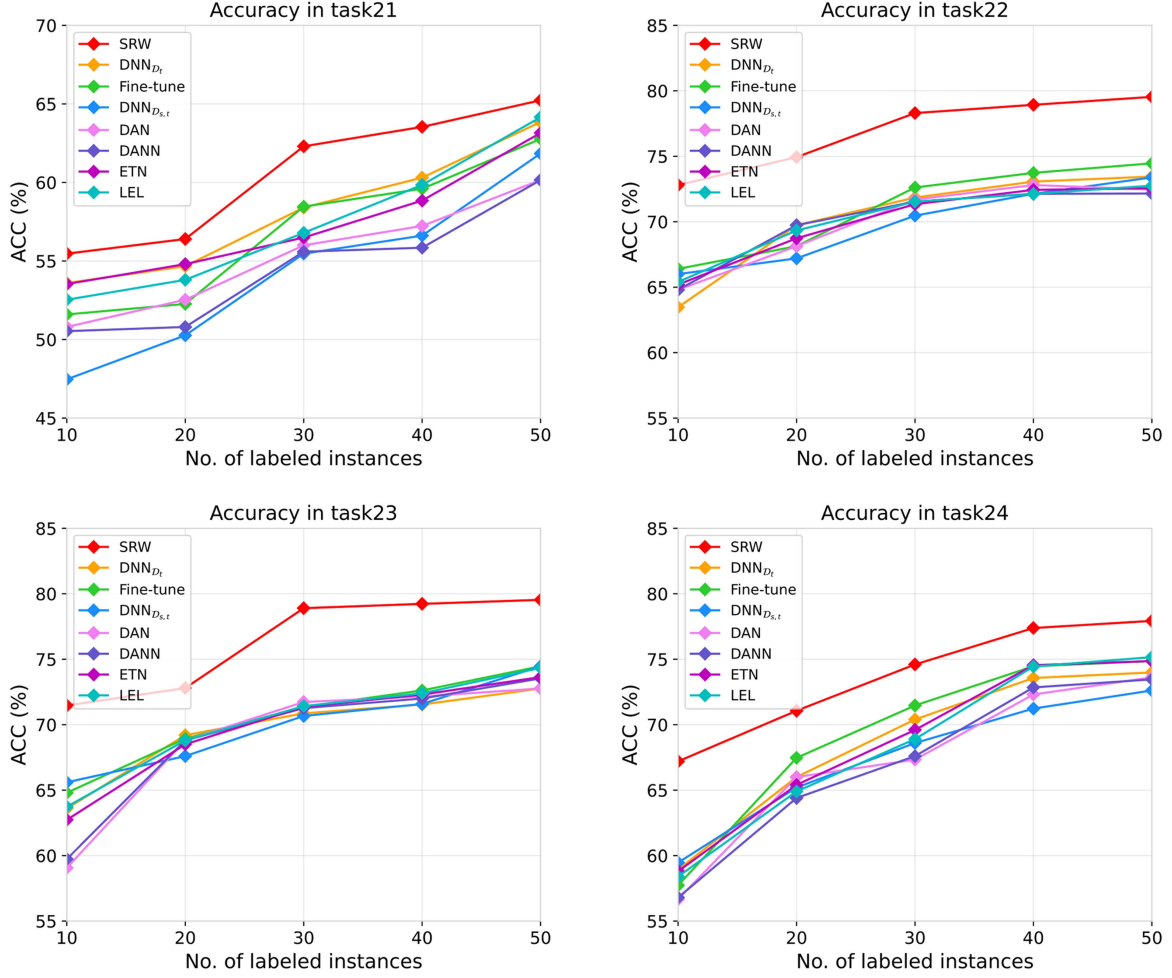


Fig. 4. Performance of different methods with varying numbers of labeled target instances on different transfer tasks within the MiniImagenet dataset.

method performs slightly better than DAN and DANN methods, which demonstrates that forcibly aligning the two domains in the heterogeneous label spaces will negatively impact the performance of the target domain. The DAN method is better than DANN, as the outstanding capability of the discriminator in DANN for matching source and target domains leads to a more severe negative transfer under the setting of heterogeneous label spaces. In most transfer tasks, the accuracy of the ETN method is lower than that of the  $DNN_{D_t}$  method, meaning that this PDA approach results in negative transfer. The LEL and DFF methods perform well in some transfer tasks but fall short in others when compared to other baseline methods. This suggests that without leveraging knowledge in the auxiliary domain, these methods may not be effective in all transfer tasks, particularly those with heterogeneous label spaces. In contrast, the proposed SRW model performs the best in all the transfer tasks, which demonstrates its effectiveness.

In the above experiments, the AlexNet model, initially trained on the ImageNet-1 K dataset, is applied for transfer tasks across datasets. Note that there is category overlap between the AWA and ImageNet-1 K datasets. To demonstrate that the good performance of the proposed SRW model is not attributable to previously seen data from the ImageNet-1 K dataset, the

TABLE V  
PERFORMANCE COMPARISON OF VARIOUS MODELS ON FOUR TRANSFER TASKS ACROSS DATASETS, WHERE  $A_{seen}$  DENOTES A SUBSET OF THE AWA DATASET CONTAINING OVERLAPPING CATEGORIES WITH THE IMAGE-1 K DATASET AND  $A_{unseen}$  DENOTES THE REMAINING SUBSET OF THE AWA DATASET CONTAINING UNSEEN CATEGORIES FROM THE IMAGE-1 K DATASET

| Tasks                      | $DNN_{D_t}$           | $DNN_{D_{s,t}}$       | DAN                   | DANN                  | LEL                   | DFF                   | SRW                         |
|----------------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------------|
| $C \rightarrow A_{unseen}$ | 70.3 <sub>(0.2)</sub> | 71.0 <sub>(0.2)</sub> | 71.2 <sub>(0.3)</sub> | 69.7 <sub>(0.4)</sub> | 71.3 <sub>(0.3)</sub> | 71.2 <sub>(0.4)</sub> | <b>73.9<sub>(0.2)</sub></b> |
| $C \rightarrow A_{seen}$   | 59.6 <sub>(0.2)</sub> | 58.2 <sub>(0.3)</sub> | 58.5 <sub>(0.2)</sub> | 57.6 <sub>(0.3)</sub> | 58.3 <sub>(0.4)</sub> | 58.7 <sub>(0.4)</sub> | <b>62.3<sub>(0.1)</sub></b> |
| $M \rightarrow A_{unseen}$ | 70.3 <sub>(0.2)</sub> | 72.6 <sub>(0.4)</sub> | 72.9 <sub>(0.4)</sub> | 72.3 <sub>(0.3)</sub> | 73.2 <sub>(0.5)</sub> | 72.8 <sub>(0.3)</sub> | <b>73.8<sub>(0.4)</sub></b> |
| $M \rightarrow A_{seen}$   | 59.6 <sub>(0.2)</sub> | 61.8 <sub>(0.1)</sub> | 61.6 <sub>(0.2)</sub> | 61.7 <sub>(0.2)</sub> | 61.7 <sub>(0.2)</sub> | 61.8 <sub>(0.4)</sub> | <b>64.7<sub>(0.2)</sub></b> |

AWA dataset is divided into two subsets: the ‘unseen’ subset containing 12 categories that do not appear in the ImageNet-1 K dataset, and the ‘seen’ subset comprising the remaining overlapping categories. Consequently, this allows for four transfer tasks across datasets, i.e.,  $C \rightarrow A_{unseen}$ ,  $C \rightarrow A_{seen}$ ,  $M \rightarrow A_{unseen}$ , and  $M \rightarrow A_{seen}$ , where  $A_{seen}$  denotes a subset of the AWA dataset containing overlapping categories with the ImageNet-1 K dataset and  $A_{unseen}$  denotes the remaining subset of the AWA dataset containing unseen categories from the ImageNet-1 K dataset. According to results shown in Table V, the proposed SRW method performs better than all the baseline methods for both subsets, showing the effectiveness of the proposed SRW

TABLE VI  
PERFORMANCE OF DIFFERENT MODELS ON STYLE TRANSFER TASKS ON THE PACS DATASETS

| Method                  | C→P                          | S→P                          | A→P                          | Avg         |
|-------------------------|------------------------------|------------------------------|------------------------------|-------------|
| DNN $\mathcal{D}_t$     | 94.4 <sub>(0.2)</sub>        | 94.4 <sub>(0.2)</sub>        | 94.4 <sub>(0.2)</sub>        | 94.4        |
| Fine-tune               | 92.0 <sub>(0.4)</sub>        | 93.7 <sub>(0.4)</sub>        | 91.5 <sub>(0.3)</sub>        | 92.4        |
| DNN $\mathcal{D}_{s,t}$ | 94.3 <sub>(0.2)</sub>        | 94.4 <sub>(0.2)</sub>        | 94.2 <sub>(0.3)</sub>        | 94.3        |
| DAN                     | 94.3 <sub>(0.2)</sub>        | 94.2 <sub>(0.1)</sub>        | 94.4 <sub>(0.1)</sub>        | 94.3        |
| DANN                    | 94.2 <sub>(0.1)</sub>        | <b>94.6</b> <sub>(0.2)</sub> | 94.3 <sub>(0.1)</sub>        | 94.4        |
| ETN                     | 94.0 <sub>(0.2)</sub>        | 94.2 <sub>(0.1)</sub>        | 94.0 <sub>(0.2)</sub>        | 94.1        |
| LEL                     | 94.6 <sub>(0.1)</sub>        | 93.9 <sub>(0.3)</sub>        | 94.9 <sub>(0.1)</sub>        | 94.5        |
| DFF                     | 94.4 <sub>(0.1)</sub>        | 93.5 <sub>(0.3)</sub>        | 95.0 <sub>(0.2)</sub>        | 94.3        |
| <b>SRW</b>              | <b>95.7</b> <sub>(0.2)</sub> | 94.5 <sub>(0.1)</sub>        | <b>95.4</b> <sub>(0.3)</sub> | <b>95.2</b> |

The best results are in bold.

TABLE VII  
PERFORMANCE OF DIFFERENT MODELS ON STYLE TRANSFER TASKS ON THE DOMAINNET DATASETS

| Method                  | P→R                          | Q→R                          | C→R                          | S→R                          | I→R                          | Avg         |
|-------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|-------------|
| DNN $\mathcal{D}_t$     | 43.1 <sub>(0.2)</sub>        | 43.1 <sub>(0.2)</sub>        | 43.1 <sub>(0.2)</sub>        | 43.1 <sub>(0.2)</sub>        | 43.1 <sub>(0.2)</sub>        | 43.1        |
| Fine-tune               | 42.8 <sub>(0.2)</sub>        | 41.7 <sub>(0.2)</sub>        | 42.6 <sub>(0.3)</sub>        | 42.6 <sub>(0.1)</sub>        | 41.9 <sub>(0.3)</sub>        | 42.3        |
| DNN $\mathcal{D}_{s,t}$ | 43.8 <sub>(0.2)</sub>        | 42.7 <sub>(0.3)</sub>        | 43.5 <sub>(0.2)</sub>        | 43.5 <sub>(0.2)</sub>        | 43.7 <sub>(0.1)</sub>        | 43.4        |
| DAN                     | 44.0 <sub>(0.1)</sub>        | 42.2 <sub>(0.2)</sub>        | 43.8 <sub>(0.2)</sub>        | 43.8 <sub>(0.3)</sub>        | 43.5 <sub>(0.3)</sub>        | 43.5        |
| DANN                    | 43.7 <sub>(0.2)</sub>        | 42.0 <sub>(0.2)</sub>        | 43.4 <sub>(0.3)</sub>        | 43.5 <sub>(0.2)</sub>        | 43.6 <sub>(0.3)</sub>        | 43.2        |
| ETN                     | 43.9 <sub>(0.1)</sub>        | 42.3 <sub>(0.2)</sub>        | 43.6 <sub>(0.2)</sub>        | 43.7 <sub>(0.1)</sub>        | 43.6 <sub>(0.3)</sub>        | 43.4        |
| LEL                     | 43.8 <sub>(0.1)</sub>        | 42.1 <sub>(0.1)</sub>        | 43.7 <sub>(0.2)</sub>        | 43.9 <sub>(0.2)</sub>        | 43.2 <sub>(0.3)</sub>        | 43.3        |
| DFF                     | 42.4 <sub>(0.2)</sub>        | 41.2 <sub>(0.2)</sub>        | 42.3 <sub>(0.2)</sub>        | 42.6 <sub>(0.2)</sub>        | 42.3 <sub>(0.1)</sub>        | 42.2        |
| <b>SRW</b>              | <b>44.7</b> <sub>(0.1)</sub> | <b>43.7</b> <sub>(0.3)</sub> | <b>43.9</b> <sub>(0.2)</sub> | <b>44.4</b> <sub>(0.3)</sub> | <b>44.2</b> <sub>(0.2)</sub> | <b>44.2</b> |

The best results are in bold.

method. Moreover, the performance on the ‘unseen’ subset is better than that on the ‘seen’ subset. One possible reason is that the numbers of categories and samples on the ‘unseen’ subset are much smaller than those on the ‘seen’ subset, making the former easier to be learned.

#### D. Sensitivity Analysis

First, we study the impact of the amount of labeled data on the performance. According to Fig. 4, the performance of the proposed SRW method improves with the increasing number of labeled data in the target domain. Additionally, Table X showcases the results for transfer tasks across different datasets. Notably, our results suggest that when there are fewer labeled instances in the target domain for transfer tasks, the advantages of employing our approach are more significant.

Second, we analyze the impact of different similarity measures for the construction of the graph defined in (1), including the cosine similarity (denoted by C similarity), euclidean distance (denoted by E distance), and inner product used in (1), to the performance. According to results in Table XIV, we observe that the SRW method with the inner product outperforms other variants, making it the first choice to construct the graph  $G$ .

Third, we vary the value of hyper-parameter  $\gamma$  to testify its sensitivity to the performance of the proposed SRW method on different transfer tasks by fixing other hyperparameters. As shown in Table XI where the value of  $\gamma$  varies between 0.1 and 1.0, we can find that the average performance on six transfer tasks is the best when  $\gamma$  is set to 0.3, which is the setting for  $\gamma$  in all the transfer tasks.

Finally, in Table XII, we conducted analysis to study the impact of  $\eta_\theta$ , the learning rate of the inner-level problem, to the performance, when fixing  $\eta_\alpha$ , the learning rate of the outer-level problem, to be 0.01. According to the results, we can see that the performance is not so sensitive to  $\eta_\theta$ . Consequently, we set the two learning rates to be the same (i.e., 0.01).

#### E. Ablation Study

First, we analyze the necessity of different losses defined in Section III-C. We compare the SRW model with its variants, including  $\text{SRW}_{w/o\ell_1}$  that does not use  $\ell_1$  defined in (2) and  $\text{SRW}_{w/o\ell_2}$  that does not use  $\ell_2$  defined in (3). According to results in Table XIV, we observe that the  $\text{SRW}_{w/o\ell_1}$  and  $\text{SRW}_{w/o\ell_2}$  models perform worse than the SRW model, indicating that both losses are important to the performance of the SRW model. To further study the effect of the proposed  $\ell_1$  and  $\ell_2$  losses in the SRW model, we conduct ablation studies to compare the variants of the SRW method on six transfer tasks across datasets. According to the results shown in Table XIII, we can find that the  $\ell_1$  and  $\ell_2$  losses are beneficial for the proposed SRW model to improve the classification performance. Overall, the proposed SRW method with both  $\ell_1$  and  $\ell_2$  losses performs the best, which demonstrates the usefulness of the two losses in SRW.

Second, we analyze the effectiveness of the meta weight network  $w$ , which aims to weigh each transfer sequence. We compare the SRW model with its variants, including  $\text{SRW}_{random}$  that weighs each transfer sequence with a random weight uniformly generated within  $[0, 1]$  and  $\text{SRW}_{equal}$  that assigns an equal weight 1.0 to each transfer sequence. According to results shown in Table XIV, SRW outperforms  $\text{SRW}_{random}$  and  $\text{SRW}_{equal}$ , indicating that not all the transfer sequences generated by random walk are equally credible and that the proposed meta weight network can reliably learn their credibility.

#### F. Visualization of Transfer Sequences

To understand how the proposed SRW method selectively leverages knowledge from transfer sequences via weights learned by the meta weight network, we plot in Fig. 5 some transfer sequences sampled by random walk and their corresponding weights learned by the meta weight network. According to Fig. 5, we can see that in each transfer sequence corresponding to one row where the source image is completely different from the target image, the SRW method can learn to weigh each transfer sequence adaptively according to the target domain. To see that, for example, in a transfer sequence ‘coral-to-flower’ that is located in the sixth row from the bottom of Fig. 5, the SRW method first relates the source image to other coral images that are similar to the source images, and then gradually visits images that are more similar to the target domain until reaching some target image. In this transfer sequence, although the images in the source and target domains belong to different categories, they have related properties such as similar shapes, resulting in a higher weight learned by the proposed meta-weight network. In contrast, in another transfer sequence ‘orange-to-bird’ that

TABLE VIII  
ACCURACY (%) OF DIFFERENT MODELS ON DIFFERENT TRANSFER TASKS WITHIN THE AWA AND CALTECH-101 DATASETS, RESPECTIVELY

| dataset                 | AwA                          |                              |                              |                              |                              | Caltech101                   |                              |                              |                              |                              |             |
|-------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|-------------|
| Methods                 | task01                       | task02                       | task03                       | task04                       | task05                       | task06                       | task07                       | task08                       | task09                       | task10                       | Avg         |
| DNN $\mathcal{D}_t$     | 68.4 <sub>(0.3)</sub>        | 68.4 <sub>(0.4)</sub>        | 69.5 <sub>(0.3)</sub>        | 69.2 <sub>(0.5)</sub>        | 70.3 <sub>(0.5)</sub>        | 76.8 <sub>(0.3)</sub>        | 81.5 <sub>(0.2)</sub>        | 81.2 <sub>(0.4)</sub>        | 81.1 <sub>(0.3)</sub>        | 82.9 <sub>(0.3)</sub>        | 74.9        |
| Fine-tune               | 68.0 <sub>(0.3)</sub>        | 68.8 <sub>(0.2)</sub>        | 70.1 <sub>(0.2)</sub>        | 69.5 <sub>(0.4)</sub>        | 70.7 <sub>(0.5)</sub>        | 77.4 <sub>(0.3)</sub>        | 79.7 <sub>(0.4)</sub>        | 79.7 <sub>(0.4)</sub>        | 82.3 <sub>(0.3)</sub>        | 79.3 <sub>(0.2)</sub>        | 74.5        |
| DNN $\mathcal{D}_{s,t}$ | 67.9 <sub>(0.2)</sub>        | 69.3 <sub>(0.3)</sub>        | 69.2 <sub>(0.4)</sub>        | 68.5 <sub>(0.4)</sub>        | 68.7 <sub>(0.3)</sub>        | 76.2 <sub>(0.2)</sub>        | 78.5 <sub>(0.2)</sub>        | 77.3 <sub>(0.5)</sub>        | 81.1 <sub>(0.4)</sub>        | 79.3 <sub>(0.4)</sub>        | 73.6        |
| DAN                     | 67.1 <sub>(0.2)</sub>        | 67.5 <sub>(0.3)</sub>        | 69.1 <sub>(0.3)</sub>        | 64.8 <sub>(0.4)</sub>        | 69.7 <sub>(0.4)</sub>        | 76.2 <sub>(0.5)</sub>        | 80.9 <sub>(0.3)</sub>        | 80.9 <sub>(0.4)</sub>        | 80.8 <sub>(0.2)</sub>        | 79.0 <sub>(0.3)</sub>        | 73.6        |
| DANN                    | 68.5 <sub>(0.3)</sub>        | 68.3 <sub>(0.3)</sub>        | 66.5 <sub>(0.2)</sub>        | 63.6 <sub>(0.4)</sub>        | 69.3 <sub>(0.5)</sub>        | 76.8 <sub>(0.4)</sub>        | 80.0 <sub>(0.4)</sub>        | 80.0 <sub>(0.5)</sub>        | 80.2 <sub>(0.3)</sub>        | 80.2 <sub>(0.2)</sub>        | 73.3        |
| ETN                     | 68.3 <sub>(0.3)</sub>        | 67.6 <sub>(0.2)</sub>        | 68.8 <sub>(0.2)</sub>        | 66.8 <sub>(0.4)</sub>        | 70.7 <sub>(0.4)</sub>        | 77.3 <sub>(0.5)</sub>        | 80.2 <sub>(0.5)</sub>        | 81.7 <sub>(0.4)</sub>        | 81.3 <sub>(0.4)</sub>        | 81.4 <sub>(0.5)</sub>        | 74.4        |
| LEL                     | 71.1 <sub>(0.2)</sub>        | 71.2 <sub>(0.3)</sub>        | 70.8 <sub>(0.2)</sub>        | 71.7 <sub>(0.5)</sub>        | 71.1 <sub>(0.5)</sub>        | 79.2 <sub>(0.4)</sub>        | 83.0 <sub>(0.3)</sub>        | 83.0 <sub>(0.2)</sub>        | 80.8 <sub>(0.5)</sub>        | 82.6 <sub>(0.4)</sub>        | 76.5        |
| SRW                     | <b>72.5</b> <sub>(0.5)</sub> | <b>73.3</b> <sub>(0.4)</sub> | <b>73.1</b> <sub>(0.4)</sub> | <b>72.4</b> <sub>(0.2)</sub> | <b>73.7</b> <sub>(0.3)</sub> | <b>82.7</b> <sub>(0.4)</sub> | <b>86.7</b> <sub>(0.3)</sub> | <b>85.2</b> <sub>(0.3)</sub> | <b>85.3</b> <sub>(0.2)</sub> | <b>84.4</b> <sub>(0.4)</sub> | <b>78.9</b> |

TABLE IX  
ACCURACY (%) OF DIFFERENT MODELS ON DIFFERENT TRANSFER TASKS WITHIN THE MINIIMAGENET DATASET

| dataset                 | MiniImagenet                 |                              |                              |                              |                              |                              |                              |                              |                              |                              |             |
|-------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|------------------------------|-------------|
| Method                  | task11                       | task12                       | task13                       | task14                       | task15                       | task16                       | task17                       | task18                       | task19                       | task20                       | Avg         |
| DNN $\mathcal{D}_t$     | 75.6 <sub>(0.2)</sub>        | 77.3 <sub>(0.4)</sub>        | 76.7 <sub>(0.3)</sub>        | 70.7 <sub>(0.3)</sub>        | 70.7 <sub>(0.2)</sub>        | 70.8 <sub>(0.2)</sub>        | 70.8 <sub>(0.1)</sub>        | 71.2 <sub>(0.4)</sub>        | 73.1 <sub>(0.4)</sub>        | 63.7 <sub>(0.3)</sub>        | 72.1        |
| Fine-tune               | 73.9 <sub>(0.1)</sub>        | 77.1 <sub>(0.3)</sub>        | 76.0 <sub>(0.2)</sub>        | 68.5 <sub>(0.4)</sub>        | 70.3 <sub>(0.2)</sub>        | 69.2 <sub>(0.3)</sub>        | 71.9 <sub>(0.3)</sub>        | 69.1 <sub>(0.2)</sub>        | 73.7 <sub>(0.2)</sub>        | 65.1 <sub>(0.4)</sub>        | 71.5        |
| DNN $\mathcal{D}_{s,t}$ | 72.4 <sub>(0.2)</sub>        | 73.6 <sub>(0.2)</sub>        | 74.0 <sub>(0.3)</sub>        | 65.3 <sub>(0.3)</sub>        | 68.3 <sub>(0.4)</sub>        | 68.1 <sub>(0.5)</sub>        | 71.2 <sub>(0.4)</sub>        | 66.5 <sub>(0.2)</sub>        | 72.1 <sub>(0.1)</sub>        | 63.2 <sub>(0.2)</sub>        | 69.5        |
| DAN                     | 73.2 <sub>(0.2)</sub>        | 77.3 <sub>(0.1)</sub>        | 76.6 <sub>(0.2)</sub>        | 68.3 <sub>(0.3)</sub>        | 69.1 <sub>(0.4)</sub>        | 69.5 <sub>(0.3)</sub>        | 69.3 <sub>(0.3)</sub>        | 68.9 <sub>(0.2)</sub>        | 72.8 <sub>(0.2)</sub>        | 62.7 <sub>(0.4)</sub>        | 70.8        |
| DANN                    | 73.3 <sub>(0.2)</sub>        | 76.7 <sub>(0.2)</sub>        | 76.4 <sub>(0.1)</sub>        | 66.8 <sub>(0.3)</sub>        | 70.9 <sub>(0.3)</sub>        | 69.2 <sub>(0.4)</sub>        | 69.9 <sub>(0.3)</sub>        | 67.6 <sub>(0.2)</sub>        | 72.1 <sub>(0.2)</sub>        | 63.5 <sub>(0.4)</sub>        | 70.6        |
| ETN                     | 75.2 <sub>(0.1)</sub>        | 77.4 <sub>(0.2)</sub>        | 76.1 <sub>(0.2)</sub>        | 70.9 <sub>(0.3)</sub>        | 71.4 <sub>(0.3)</sub>        | 70.3 <sub>(0.4)</sub>        | 70.7 <sub>(0.3)</sub>        | 70.4 <sub>(0.2)</sub>        | 73.2 <sub>(0.4)</sub>        | 64.1 <sub>(0.2)</sub>        | 72.0        |
| LEL                     | 77.2 <sub>(0.2)</sub>        | 80.4 <sub>(0.2)</sub>        | 78.7 <sub>(0.1)</sub>        | 71.7 <sub>(0.2)</sub>        | 70.1 <sub>(0.2)</sub>        | 71.3 <sub>(0.4)</sub>        | 73.6 <sub>(0.3)</sub>        | 69.7 <sub>(0.3)</sub>        | 74.0 <sub>(0.2)</sub>        | 65.6 <sub>(0.4)</sub>        | 73.2        |
| SRW                     | <b>78.7</b> <sub>(0.2)</sub> | <b>82.8</b> <sub>(0.2)</sub> | <b>80.8</b> <sub>(0.1)</sub> | <b>74.8</b> <sub>(0.2)</sub> | <b>74.4</b> <sub>(0.4)</sub> | <b>73.5</b> <sub>(0.3)</sub> | <b>75.3</b> <sub>(0.3)</sub> | <b>74.3</b> <sub>(0.3)</sub> | <b>76.9</b> <sub>(0.4)</sub> | <b>67.3</b> <sub>(0.3)</sub> | <b>75.9</b> |

TABLE X  
SENSITIVITY TO THE NUMBER OF LABELED DATA POINTS IN THE TARGET DOMAIN, VARYING FROM 5 TO 30 ON DIFFERENT TRANSFER TASKS ACROSS DATASETS

|     | $N_o$ | C→A  | A→C  | M→A  | A→M  | M→C  | C→M  | Avg  |
|-----|-------|------|------|------|------|------|------|------|
| SRW | 5     | 50.9 | 64.4 | 53.0 | 45.8 | 63.0 | 45.7 | 53.8 |
|     | 10    | 58.1 | 72.4 | 59.5 | 54.0 | 73.1 | 54.5 | 61.9 |
|     | 20    | 64.1 | 77.4 | 65.2 | 59.1 | 77.2 | 59.1 | 67.0 |
|     | 30    | 66.9 | 79.9 | 67.7 | 61.5 | 78.7 | 61.9 | 69.4 |

TABLE XI  
SENSITIVITY OF HYPER-PARAMETER  $\gamma$  WITH THE VALUES VARYING FROM 0.1 TO 1.0 ON DIFFERENT TRANSFER TASKS ACROSS DATASETS

|     | $\gamma$ | C→A  | A→C  | M→A  | A→M  | M→C  | C→M  | Avg         |
|-----|----------|------|------|------|------|------|------|-------------|
| SRW | 0.1      | 57.9 | 72.0 | 59.9 | 54.0 | 72.9 | 54.8 | <b>61.9</b> |
|     | 0.3      | 58.1 | 72.4 | 59.5 | 54.0 | 73.1 | 54.5 | <b>61.9</b> |
|     | 0.5      | 58.2 | 72.1 | 59.5 | 54.0 | 72.6 | 54.3 | 61.8        |
|     | 0.7      | 57.9 | 72.7 | 59.2 | 54.1 | 72.5 | 54.5 | 61.8        |
|     | 1.0      | 57.8 | 72.8 | 59.7 | 54.0 | 72.5 | 54.3 | 61.8        |

TABLE XII  
SENSITIVITY ANALYSIS WITH RESPECT TO THE OUTER LEARNING RATE VARYING FROM 0.005 TO 0.1 ON DIFFERENT TRANSFER TASKS ACROSS DATASETS

|     | $\eta_\theta$ | C→A  | A→C  | M→A  | A→M  | M→C  | C→M  | Avg  |
|-----|---------------|------|------|------|------|------|------|------|
| SRW | 0.005         | 57.9 | 72.7 | 59.7 | 53.9 | 73.2 | 54.7 | 62.0 |
|     | 0.01          | 58.1 | 72.4 | 59.5 | 54.0 | 73.1 | 54.5 | 61.9 |
|     | 0.05          | 58.0 | 72.7 | 59.7 | 54.1 | 72.9 | 54.7 | 62.0 |
|     | 0.1           | 58.1 | 72.7 | 59.8 | 54.1 | 73.2 | 54.7 | 62.1 |

is located in the fifth row from the bottom of Fig. 5, although images in this transfer sequence are similar in terms of the color, the difference between them is quite obvious, which makes the meta weight network learn a low weight, meaning that this transfer sequence is not so reliable for the target domain.

TABLE XIII  
ACCURACY (%) OF THE PROPOSED SRW METHOD AND THEIR VARIANTS IN SIX TRANSFER TASKS ACROSS DATASETS.  $\ell_1$  DENOTES THE SIMILARITY LOSS DEFINED IN (2), AND  $\ell_2$  DENOTES THE RECONSTRUCTION LOSS DEFINED IN (3)

|     | $\ell_1$ | $\ell_2$ | C→A  | A→C  | M→A  | A→M  | M→C  | C→M  | Avg         |
|-----|----------|----------|------|------|------|------|------|------|-------------|
| SRW |          |          | 54.5 | 70.9 | 57.4 | 51.3 | 71.2 | 50.2 | 59.2        |
|     | ✓        |          | 55.8 | 72.2 | 57.3 | 52.4 | 72.1 | 51.7 | 60.2        |
|     |          | ✓        | 58.4 | 71.7 | 59.6 | 53.6 | 72.7 | 54.6 | 61.7        |
|     | ✓        | ✓        | 58.1 | 72.4 | 59.5 | 54.0 | 73.1 | 54.5 | <b>61.9</b> |

TABLE XIV  
ACCURACY (%) OF DIFFERENT MODELS INFLUENCED BY MEASUREMENTS FOR THE FEATURE SIMILARITY AND VARIANTS OF THE SRW MODEL ON TRANSFER TASKS WITHIN THE MINIIMAGENET DATASET

| Method                                 | task21      | task22      | task23      | task24      | Avg         |
|----------------------------------------|-------------|-------------|-------------|-------------|-------------|
| <i>C similarity</i>                    | 59.2        | 75.8        | 76.7        | 72.3        | 71.0        |
| <i>E distance</i>                      | 59.2        | 75.3        | 76.8        | 72.3        | 70.9        |
| SRW <sub>w/o <math>\ell_1</math></sub> | 58.3        | 75.5        | 75.6        | 74.1        | 70.9        |
| SRW <sub>w/o <math>\ell_2</math></sub> | 58.9        | 77.4        | 76.9        | 73.5        | 71.7        |
| SRW <sub>random</sub>                  | 60.1        | 76.7        | 78.2        | 73.2        | 72.1        |
| SRW <sub>equal</sub>                   | 61.6        | 77.1        | 78.5        | 73.7        | 72.7        |
| SRW                                    | <b>62.3</b> | <b>78.3</b> | <b>78.9</b> | <b>74.6</b> | <b>73.5</b> |

Hence, based on transfer sequences, we can understand how the SRW model works and this can help improve the model interpretability.

### G. Visualization of Learned Graph

To better analyze the proposed SRW method, in Fig. 6, we visualize the final learned graph  $G$  using a batch of data for the  $A \rightarrow C$  and  $C \rightarrow A$  transfer tasks. To better visualize  $G$  that is fully connected, we randomly select 1% of the edges in  $G$  in Fig. 6. We can see that the model adeptly assigns distinct weights

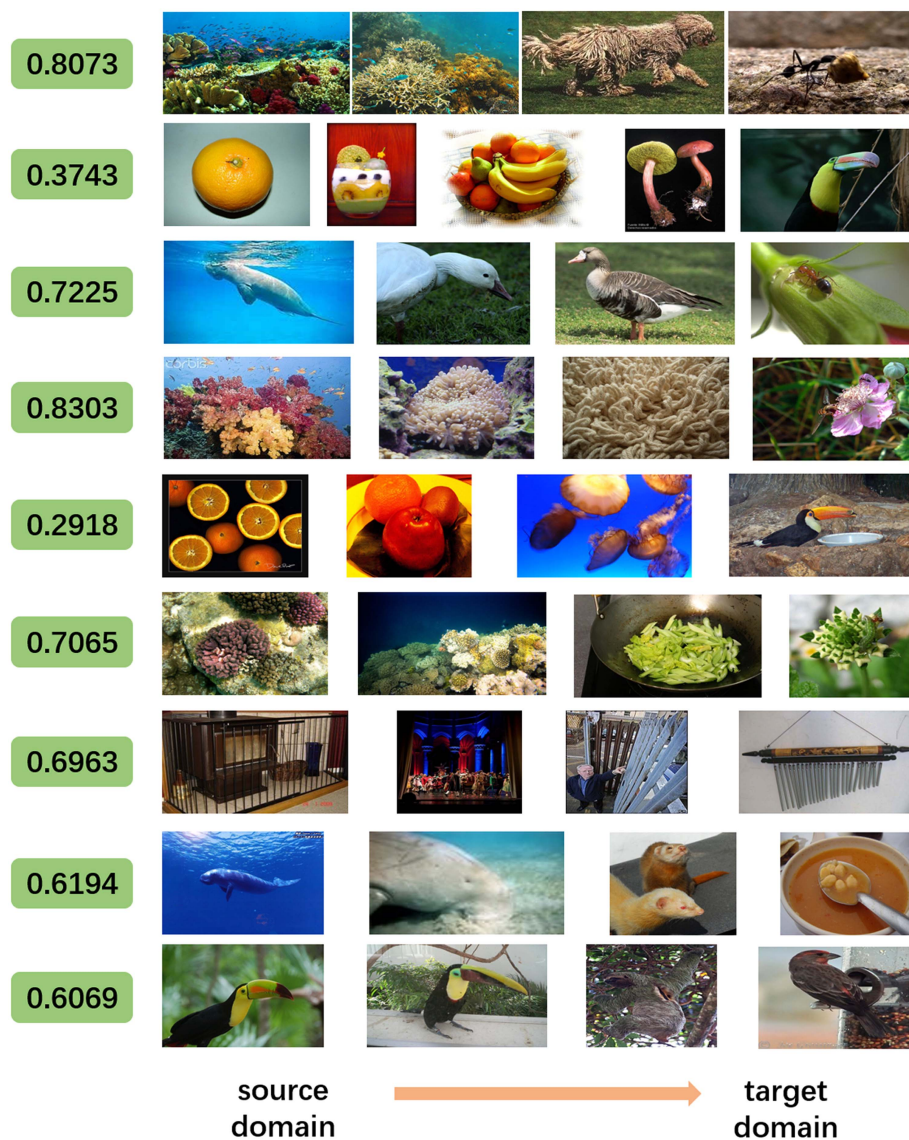


Fig. 5. Examples of transfer sequences and their weights. Each row represents a transfer sequence with its weight on the leftmost column.

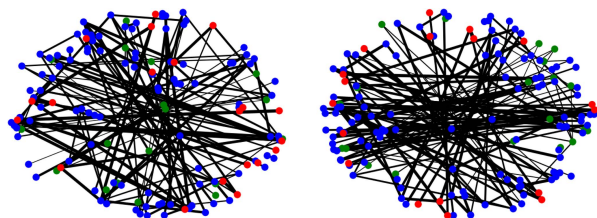


Fig. 6. Visualization of learned graph for the transfer tasks:  $A \rightarrow C$  and  $C \rightarrow A$ . Source data points are in green, target data points are in red, and auxiliary data points are in blue. The weights of edges are normalized for clear illustration, where thinner edges correspond to smaller weights and thicker edges represent larger weights.

to each edge, which demonstrates the capability of the model to discern the significance of each node in  $G$ . Transfer sequences are subsequently generated based on those weights through random walk. The meta weight network thus becomes pivotal

TABLE XV  
COMPUTATIONAL COMPARISON OF TRAINING AND INFERENCE TIME (SECONDS)  
FOR EACH ITERATION ON THE  $C \rightarrow A$  AND  $A \rightarrow C$  TRANSFER TASKS

| Method                  | C $\rightarrow$ A |           | A $\rightarrow$ C |           |
|-------------------------|-------------------|-----------|-------------------|-----------|
|                         | Training          | Inference | Training          | Inference |
| DNN $\mathcal{D}_{s,t}$ | 0.88              | 0.02      | 0.95              | 0.02      |
| Fine-tune               | 0.85              | 0.02      | 0.88              | 0.02      |
| SRW                     | 3.61              | 0.02      | 3.12              | 0.02      |

to further guarantee the credibility of each transfer sequence. As Table XIV substantiates, the weights of transfer sequences allocated by the proposed meta weight network consistently surpass those that are assigned equally or randomly, highlighting that transfer sequences generated through random walk are not equally reliable.

## H. Complexity Analysis

As detailed in Table XV, we present a comparative analysis of the training and inference time per iteration for the  $A \rightarrow C$  and  $C \rightarrow A$  transfer tasks. The proposed SRW method possesses a bi-level optimization process, leading to longer training time compared to other transfer learning approaches. However, it is crucial to highlight that the inference time is comparable to those of conventional methods.

## V. CONCLUSION

In this article, the SRW method is proposed to improve the performance of transfer learning in heterogeneous label spaces. The SRW model can transfer the knowledge with transfer sequences between the source domain and the target domain via the unlabeled auxiliary data. The information of transfer sequences can be leveraged selectively by the proposed meta weight network. Extensive experiments have been conducted to verify the effectiveness of the SRW method. In our future study, we are interested in extending the SRW method to other related learning tasks.

## REFERENCES

- [1] Q. Yang, Y. Zhang, W. Dai, and S. J. Pan, *Transfer Learning*. Cambridge, U.K.: Cambridge Univ. Press, 2020.
- [2] S. J. Pan, D. Shen, Q. Yang, and J. T. Kwok, "Transferring localization models across space," in *Proc. 23rd AAAI Conf. Artif. Intell.*, 2008, pp. 1383–1388.
- [3] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko, "Simultaneous deep transfer across domains and tasks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 4068–4076.
- [4] M. Long, J. Wang, Y. Cao, J. Sun, and P. S. Yu, "Deep learning of transferable representation for scalable domain adaptation," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 8, pp. 2027–2040, Aug. 2016.
- [5] M. N. A. Khan and D. R. Heisterkamp, "Adapting instance weights for unsupervised domain adaptation using quadratic mutual information and subspace learning," in *Proc. 23rd Int. Conf. Pattern Recognit.*, 2016, pp. 1560–1565.
- [6] W. Dai, Q. Yang, G. Xue, and Y. Yu, "Boosting for transfer learning," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 193–200.
- [7] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," in *Proc. 21st Int. Joint Conf. Artif. Intell.*, 2009, pp. 1187–1192.
- [8] D. H. Hu and Q. Yang, "Transfer learning for activity recognition via sensor mapping," in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, 2011, pp. 1962–1967.
- [9] Z. Cao, M. Long, J. Wang, and M. I. Jordan, "Partial transfer learning with selective adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2724–2732.
- [10] Z. Cao, L. Ma, M. Long, and J. Wang, "Partial adversarial domain adaptation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 139–155.
- [11] Z. Cao, K. You, M. Long, J. Wang, and Q. Yang, "Learning to transfer examples for partial domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2980–2989.
- [12] K. Saito, S. Yamamoto, Y. Ushiku, and T. Harada, "Open set domain adaptation by backpropagation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 156–171.
- [13] P. P. Busto and J. Gall, "Open set domain adaptation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 754–763.
- [14] K. You, M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Universal domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2720–2729.
- [15] K. Saito, D. Kim, S. Sclaroff, and K. Saenko, "Universal domain adaptation through self supervision," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 16 282–16 292.
- [16] W. Zhang, X. Li, H. Ma, Z. Luo, and X. Li, "Universal domain adaptation in fault diagnostics with hybrid weighted deep adversarial learning," *IEEE Trans. Ind. Inform.*, vol. 17, no. 12, pp. 7957–7967, Dec. 2021.
- [17] J. N. Kundu, N. Venkat, M. V. Rahul, and R. V. Babu, "Universal source-free domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4544–4553.
- [18] S. Yang, Y. Wang, J. van de Weijer, L. Herranz, and S. Jui, "Unsupervised domain adaptation without source data by casting a BAIT," 2020, *arXiv: 2010.12427*.
- [19] J. Liang, D. Hu, and J. Feng, "Do we really need to access the source data? Source hypothesis transfer for unsupervised domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 6028–6039.
- [20] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
- [21] G.-J. Qi, C. Aggarwal, Y. Rui, Q. Tian, S. Chang, and T. Huang, "Towards cross-category knowledge propagation for learning visual concepts," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2011, pp. 897–904.
- [22] Z. Luo, Y. Zou, J. Hoffman, and L. Fei-Fei, "Label efficient learning of transferable representations across domains and tasks," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 164–176.
- [23] Q. Xiao and Y. Zhang, "Distant transfer learning via deep random walk," in *Proc. 35th AAAI Conf. Artif. Intell.*, 2021, pp. 10422–10429.
- [24] B. Tan, Y. Song, E. Zhong, and Q. Yang, "Transitive transfer learning," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 1155–1164.
- [25] B. Tan, Y. Zhang, S. J. Pan, and Q. Yang, "Distant domain transfer learning," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 2604–2610.
- [26] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2014, pp. 701–710.
- [27] M. K. Ng, Q. Wu, and Y. Ye, "Co-transfer learning via joint transition probability graph based method," in *Proc. 1st Int. Workshop Cross Domain Knowl. Discov. Web Social Netw. Mining*, 2012, pp. 1–9.
- [28] B. Tan, E. Zhong, M. K. Ng, and Q. Yang, "Mixed-transfer: Transfer learning over mixed graphs," in *Proc. SIAM Int. Conf. Data Mining*, 2014, pp. 208–216.
- [29] J. Shu, Z. Xu, and D. Meng, "Small sample learning in Big Data era," 2018, *arXiv: 1808.04572*.
- [30] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *Proc. 5th Int. Conf. Learn. Representations*, 2017, pp. 1–11.
- [31] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 4077–4087.
- [32] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.
- [33] M. Dehghani, A. Mehrjou, S. Gouws, J. Kamps, and B. Schölkopf, "Fidelity-weighted learning," in *Proc. 6th Int. Conf. Learn. Representations*, 2018, pp. 1–20.
- [34] Y. Fan, F. Tian, T. Qin, X. Li, and T. Liu, "Learning to teach," in *Proc. 6th Int. Conf. Learn. Representations*, 2018, pp. 1–16.
- [35] L. Wu et al., "Learning to teach with dynamic loss functions," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 6467–6478.
- [36] J. Shu et al., "Meta-weight-net: Learning an explicit mapping for sample weighting," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 1917–1928.
- [37] L.-Z. Guo, Z.-Y. Zhang, Y. Jiang, Y.-F. Li, and Z.-H. Zhou, "Safe deep semi-supervised learning for unseen-class unlabeled data," in *Proc. 37th Int. Conf. Mach. Learn.*, 2020, pp. 3897–3906.
- [38] J. F. Bard, *Practical Bilevel Optimization: Algorithms and Applications*, vol. 30. Berlin, Germany: Springer, 2013.
- [39] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 4331–4340.
- [40] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 951–958.
- [41] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," California Inst. Technol., Pasadena, CA, USA, Tech. Rep. CNS-TR-2007-001, 2007.
- [42] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
- [43] O. Russakovsky et al., "Imagenet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.
- [44] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

- [45] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *Proc. 6th Int. Conf. Learn. Representations*, 2018, pp. 1–16.
- [46] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," in *Proc. 32nd Int. Conf. Mach. Learn.*, 2015, pp. 97–105.
- [47] Y. Ganin et al., "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, pp. 59:1–59:35, 2016.
- [48] S. Niu, Y. Hu, J. Wang, Y. Liu, and H. Song, "Feature-based distant domain transfer learning," in *Proc. IEEE Int. Conf. Big Data*, 2020, pp. 5164–5171.



**Qiao Xiao** received the BS degree from Shenzhen University, China, in 2014, and the MS degree from the Harbin Institute of Technology, in 2018, respectively. He is currently working toward the PhD degree with the Department of Mathematics and Computer Science, Eindhoven University of Technology. His research interests mainly include transfer learning, sparse training, and federated learning.



**Yu Zhang** (Member, IEEE) is an associate professor with the Department of Computer Science and Engineering, Southern University of Science and Technology. His research interests mainly include artificial intelligence and machine learning, especially in multi-task learning, transfer learning, dimensionality reduction, metric learning, and semi-supervised learning. He has published a book *Transfer Learning* and about 80 papers on top-tier conferences and journals. He serves as a reviewer for various journals and area chairs/(senior) program committee members for several top-tier conferences. He has won the best article awards in UAI 2010 and PAKDD 2019, and the best student article award in WI 2013.



**Qiang Yang** (Fellow, IEEE) is the chief artificial intelligence officer of WeBank and chair professor of CSE Department of Hong Kong University of Science and Technology. He is the conference chair of AAAI-21, president of Hong Kong Society of Artificial Intelligence and Robotics (HKSAR), the president of Investment Technology League (ITL), and a former president of IJCAI (2017–2019). He is a fellow of AAAI, ACM, and AAAS. His research interests include transfer learning and federated learning. He is the founding EiC of two journals *IEEE Transactions on Big Data* and *ACM Transactions on Intelligent Systems and Technology*. His latest books are *Transfer Learning* and *Federated Learning*.