Wang *et al. J Wireless Com Network* (2022) 2022:37
https://doi.org/10.1186/s13638-022-02104-8

EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

A framework for self-supervised federated domain adaptation

Bin Wang[1], Gang Li[2], Chao Wu[2*], WeiShan Zhang[1], Jiehan Zhou[3] and Ye Wei[4]

*Correspondence:
chao.wu@zju.edu.cn
[2] School of Public Affairs, ZheJiang University, HangZhou 310000, China
Full list of author information is available at the end of the article

Abstract
Unsupervised federated domain adaptation uses the knowledge from several distributed unlabelled source domains to complete the learning on the unlabelled target domain. Some of the existing methods have limited effectiveness and involve frequent communication. This paper proposes a framework to solve the distributed multi-source domain adaptation problem, referred as self-supervised federated domain adaptation (SFDA). Specifically, a multi-domain model generalization balance is proposed to aggregate the models from multiple source domains in each round of communication. A weighted strategy based on centroid similarity is also designed for SFDA. SFDA conducts self-supervised training on the target domain to tackle domain shift. Compared with the classical federated adversarial domain adaptation algorithm, SFDA is not only strong in communication cost and privacy protection but also improves in the accuracy of the model.

Keywords: Domain adaptation, Distributed system, Self-supervised, Federated learning

1 Introduction

In recent years, deep learning has been developed, and some technologies [1–6] to improve the performance of deep learning have also emerged. However, deep learning often requires large amounts of labelled data. Labelling data are notoriously time-consuming and laborious. At present, there are various datasets, but when a new task comes along, the model trained on these datasets does not work well for the new task due to distribution differences. How to learn when the probability distributions of the source domain and target domain are inconsistent is the domain adaptation learning problem [7]. Specifically, learning the target model with labelled source domain data and unlabelled target domain data is domain adaptation (generally unsupervised domain adaptation). Domain adaptation is easy to confuse with domain generalization. The main difference between the two is the availability of target domain data. The former should obtain target domain data, and the latter should not. Sometimes more than one source domain is used to improve adaptation performance, i.e. the accuracy of the target model. Unsupervised multi-source domain adaptation (UMDA) improves the model performance by establishing transferable features from multiple labelled source domains to unlabelled target domains. Many UMDA methods combine the data from the target domain with the data from each source domain to form several source–target domain pairs, and then establish transferable characteristics by narrowing the distance between the domains [8]. This paper focuses on UMDA scenario.

In addition to the adaption problems, data privacy protection is also receiving increasing attention. To protect privacy, raw data cannot be shared between domains during the training of the model. Based on this limitation, federated learning [9] is used to solve this problem. Federated learning can be considered a kind of distributed machine learning, but it is very different from traditional distributed machine learning where it has very strict requirements on privacy and efficiency, and the data distribution of each node is non-independent and identically distributed(non-IID), which causes there to be obvious domain shifts between nodes in a specific task. Inspired by federated learning and unsupervised domain adaptation, Peng et al. [10] proposed unsupervised federated domain adaptation (UFDA). In addition, they also mainly introduced a method to solve the problem of UFDA by using federated adversarial domain adaptation (FADA). However, FADA not only has poor accuracy but also has high communication costs and is prone to privacy leakage. This adversarial training method requires each source domain to exchange and update model parameters with the target domain after each mini-batch, which undoubtedly consumes a large amount of communication resources, to some extent, and increases the risk of privacy leakage. In contrast, the multi-domain model generalization balance (MDMGB) in this paper reduces the impact of the two defects and improves accuracy. At the same time, the performance of the model is further enhanced with information maximization and pseudo-label technology. We coin the whole process, including MDMGB, as self-supervised federated domain adaptation (SFDA).

Our main contributions are summarized as follows.

1. Propose an architecture which efficiently and effectively transfers knowledge learned from multiple source domains to the target domain.
2. Develop a weighting strategy based on the centroid similarity between the source and target domains. The proposed strategy does not require the sharing of raw data between domains.
3. Our approach is empirically evaluated over two benchmark datasets, and compared with existing methods, the performance of our method is significantly improved.

Section 2 presents the related work. Section 3 describes the details of multi-domain model generalization balance and self-supervised federated domain adaptation. Section 4 presents the experiments and evaluates the results. The experiments are provided and analysed in Sect. 4. Finally, Sect. 5 concludes this work and discuss future work in Sect. 5.

2 Related works

2.1 Unsupervised multi-source domain adaptation

UMDA aims to transfer knowledge learned from multiple labelled source domains to a single unlabelled target domain. Many unsupervised multi-source domain adaptation methods are based on the theory of learning from different domains proposed by Ben-David et al. [11]. At present, there are two mainstream strategies to learn transferable features. One is the moment matching method represented by the maximum mean discrepancy (MMD) and correlation alignment (CORAL) [12–18], and the other is the adversarial training method [17, 19–22]. In the first kind of method, MMD [12] and CORAL [14] are designed to match first-order (mean) and second-order (covariance) statistics of different distributions, respectively, while HoMM [15] can perform moment tensor matching of any order. The maximum mean discrepancy is often used to measure the distance between two distributions and is a commonly used loss function in transfer learning. MMD is an effective measure that can compare different distributions without initial estimation of the density function. In domain adaptation, the original MMD [12] is defined as

$$MMD(F,p,q) = sup_{f\in F}\left(E_{x\sim p}[f(x)] - E_{y\sim q}\left[f\left(y\right)\right]\right), \tag{1}$$

where p and q represent the probability distributions of the source domain and target domain, respectively. sup stands for supremum and E stands for expectation value. F is a set of functions in RKHS (reproducing kernel Hilbert space) whose norm is less than or equal to 1. However, the above form cannot be calculated directly and requires the use of the kernel trick. The most commonly used kernel function is the Gaussian kernel function,

$$k_G\left(x,y\right) = \exp\left(-\frac{\left\|\mathrm{x}-\mathrm{y}\right\|^2}{2s^2}\right). \tag{2}$$

By replacing the expectation with the mean value of each small batch, the following computable form can be obtained:

$$MMD_e(x_s,x_t) = \left\|\frac{1}{n_s}\sum_{i=1}^{n_s}\emptyset\left(x_s^{(i)}\right) - \frac{1}{n_t}\sum_{j=1}^{n_t}\emptyset\left(x_t^{(j)}\right)\right\|_H, \tag{3}$$

$$MMD_e(x_s,x_t) = \sqrt{\frac{1}{n_s^2}\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}k\left(x_s^{(i)},x_s^{(j)}\right) + \frac{1}{n_t^2}\sum_{i=1}^{n_t}\sum_{j=1}^{n_t}k\left(x_t^{(i)},x_t^{(j)}\right) - \frac{2}{n_sn_t}\sum_{i=1}^{n_s}\sum_{j=1}^{n_t}k\left(x_s^{(i)},x_t^{(j)}\right)}. \tag{4}$$

Among the second kind of methods, the representative algorithm is DANN (domain adversarial neural network), which for the first time introduces the idea of adversarial training [23] into the field of transfer learning and learns the features with category discrimination and domain invariance through the joint optimization of feature extractor, label predictor and domain classifier. The feature extractor and domain classifier are equivalent to the generator and discriminator in the generative adversarial network, respectively, and they form an adversarial relationship to promote the emergence of features with domain invariance.

2.1.1 Domain generalization

Unlike domain adaptation, domain generalization [8, 24, 25] cannot use any sample of the target domain, but it still has to capture transferable information across domains. To complete the classification task without the target domain available at the time of training, labelled data from several related classification tasks can be used. Meta-learning involves generalization to a new task [13], but because meta-learning is more concerned with how quickly the model converges when the labelled data for the new task are acquired (in small amounts), it assumes that the labelled data for the new task are accessible. When a new task appears, meta-learning needs to be retrained, but the domain generalization model does not need to be retrained.

2.1.2 Federated learning

Federated learning (FL) is a machine learning setup that aims to train a high-quality centralized model while training data remain distributed over a large number of clients with unreliable and relatively slow network connections for each client. For the optimization of communication problems, many effective methods [26] have been proposed, so this paper focuses on how to learn a centralized model with high performance. The federated optimization problem has four key properties: non-IID: the data on each client are specific to the user; unbalanced: the amount of data generated by the client varies greatly; massively distributed: the number of clients participating in an optimization is much larger than the average number of examples per client; and limited communication: network connections are unstable, insecure or expensive. The third feature has received less attention because most of the data are typically concentrated in a very small number of nodes, with the remaining nodes contributing almost nothing to the results. The federated averaging (FedAvg) algorithm [9] is a basic algorithm that aggregates model updates from different clients in each round of communication. Its key point is that each client model must have the same random initialization except that the server distributes the aggregated model to the client in each round of communication. However, FedAvg's weighting of the client model is based on the amount of data, and the amount of data does not correlate significantly with differences in the domain distribution. Therefore, it cannot be simply applied to unsupervised multi-source domain adaptation.

2.1.3 Federated domain adaptation

There has been very little discussion of distributed UMDA, but there is no lack of excellent articles, the most advanced and representative of which is the recently proposed KD3A (knowledge distillation-based decentralized domain adaptation) [27]. However, its computational efficiency is relatively low because each iteration is trained in the source domain in the early stage and the target domain in the late stage. When either party is training, the other party has to idle for a long time and wait for the return. The concept of unsupervised federated domain adaptation (UFDA) was first proposed in FADA [10]. UFDA studies how to transfer knowledge learned from decentralized nodes to a new node with a different data distribution. UFDA presents three challenges: first, the data are stored locally on the client and cannot be shared, which makes mainstream domain adaptation approaches unfeasible because they require both labelled source domain data and unlabelled target domain data. For each node, the model parameters are trained separately and converge at different rates, and their contribution to the target model depends on how close each source domain is to the target domain. Finally, the knowledge learned from each source domain is highly entangled, which is likely to result in a negative transfer. SHOT [28] proposed a self-supervised method to solve the domain adaptation problem of the separation of a single source domain and target domain. (The two do not share data.) It produces the same number of result models as the number of source domains, which increases the overhead of storage space. In addition, each model is involved in the calculation of the results.

To solve the problem of multi-source model aggregation in federated domain adaptation, this paper proposes a multi-domain model generalization balance algorithm (MDMGB). The weighting strategy of this method abandons the traditional weighting method which depends on the quantity of data, and measures the tightness of the source domain and target domain by the similarity of the centroid. MDMGB can calculate weights without sharing data between domains. In addition, to improve the performance of domain adaptation under federated learning constraints, we propose a self-supervised federated domain adaptation framework (SFDA). MDMGB is used for the aggregation of each source model in each communication. SFDA completely separates the training of the source domain model from the training of the target domain model and simultaneously trains the target model in a self-supervised way.

3 Methods

Let D_S and D_T represent the source and target domains, respectively. In the unsupervised multi-source domain adaptation, there are a total of K source domains $\left\{D_S^k\right\}_{k=1}^{K}$, each of which has N_k labelled samples, i.e. $D_S^k = \left\{\left(X_i^k, y_i^k\right)\right\}_{i=1}^{N_k}$, and a target domain with N_T unlabelled samples, i.e. $D_T = \left\{X_i^T\right\}_{i=1}^{N_T}$. The goal of UMDA is to learn a model

h that minimizes the task-specific loss ε_{D_T} on the target domain D_T, where $\varepsilon_{D_T}(h) = Pr_{(X,y)\sim D_T}[h(X) \neq y]$. The general UMDA model learns transferable features by minimizing $\mathcal{H}$-divergence. In the context of distributed UMDA, this paper does not need to access the data of each source domain, but studies the domain adaptation problem under the federated learning framework. Each domain that contains the target domain and the source domains is treated as a client in federated learning. If conditions permit, the client of the target domain can also act as the server.

3.1 Centroid alignment strategy

The centroid of each class in the domain is obtained by the following formula [29]:

$$c_k^{(0)} = \frac{\sum_{x\in X} \delta_k\left(\hat{f}(x)\right) g(x)}{\sum_{x\in X} \delta_k\left(\hat{f}(x)\right)}. \tag{5}$$

$f(x) = h\big(g(x)\big)$ is the k-dimensional output of each sample, g represents the feature extractor and h represents the classifier. $\delta_k(a) = \frac{\exp(a_k)}{\sum_i \exp(a_i)}$ refers to the kth element in the softmax output of the k-dimensional vector a. These centroids can stably and more reliably represent the distribution of different classes in the domain. Then, the closest centroid classifier is used to obtain the pseudo-label of each sample:

$$\hat{y} = \arg\min_k D_f\left(g(x), c_k^{(0)}\right), \tag{6}$$

$D_f(a, b)$ measures the cosine distance between a and b. The specific form is as follows:

$$D_f(a,b) = 1 - \cos(a,b) = \frac{\|a\|_2\left\|b\right\|_2 - a\cdot b}{\|a\|_2\left\|b\right\|_2}. \tag{7}$$

The centroids are constantly updated in an iterative manner through the following formula:

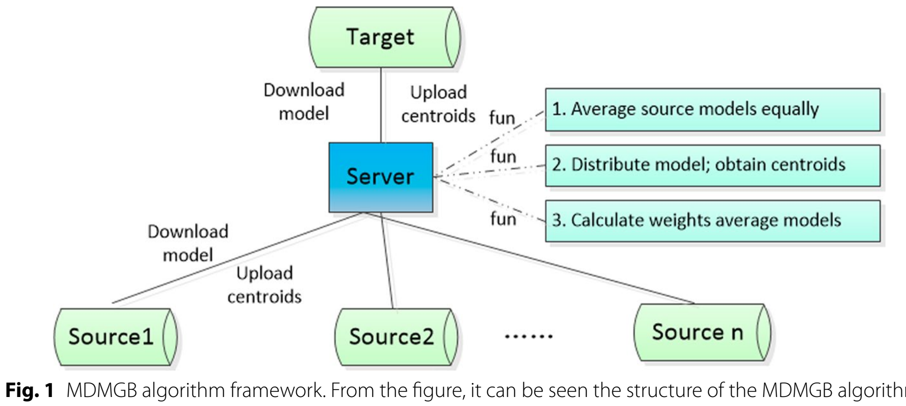

Fig. 1 MDMGB algorithm framework. From the figure, it can be seen the structure of the MDMGB algorithm and processes among sections.

$$c_k^{(1)} = \frac{\sum_{x \in X} 1_{[\hat{y}=k]} g(x)}{\sum_{x \in X} 1_{[\hat{y}=k]}}, \tag{8}$$

$$\hat{y} = \arg\min_k D_f\left(g(x), c_k^{(1)}\right). \tag{9}$$

However, one update is usually good enough. It is worth mentioning that in practical calculations, a one-dimensional nonzero number (such as 1) is appended to each eigenvector to avoid division by zero.

3.2 The algorithm of MDMGB

Figure 1 shows the structure of the MDMGB algorithm. The weight of each source domain is determined according to its distance (tightness) from the target domain. Since the weight calculation requires the use of the target domain, MDMGB omits steps 2 and 3 when the target domain is not yet available. For UMDA, there are many methods to calculate the weight of the source domain based on the distance between the source domain and the target domain, but they do not work in the UFDA condition due to the need to obtain both source domain and target domain data. In contrast to existing distributed UMDA approaches, MDMGB aggregates the model rather than the original gradient. In addition, MDMGB does not use the amount of data on each source domain in the process of weighting the source domain; for specific reasons, please refer to Appendix A. It should be emphasized that MDMGB is built on the premise that each source domain has a sufficiently large amount of data, corresponding to the critical few nodes in federated learning, and that the data volume of each node cannot differ by orders of magnitude.

The following is the weighting strategy of the MDMGB algorithm. Assume that we now have the uploaded source models $\left\{w^1, w^2, \ldots, w^K\right\}$ from K source domains. A preliminary model was obtained by averaging them directly:

$$w^0 = \frac{1}{K}\sum_{k=1}^{K} w^k. \tag{10}$$

Next, w^0 is distributed to the target domain and each source domain to compute and return the centroid for each category as described below. At this point, assuming that the number of categories is L, the centroids of each source domain are $\left\{c_0^k, c_1^k, \ldots, c_{L-1}^k\right\}_{k=1}^{K}$ and the centroids of the target domain are $\left\{c_0^T, c_1^T, \ldots, c_{L-1}^T\right\}$. The size of a set of centroids uploaded by each domain is only related to the number of categories and the size of the eigenvector, which are small relative to the model. This process has almost no impact on communication efficiency. Then, the cosine similarity of each centroid of each source domain and the corresponding centroid of the target domain is calculated and summed:

$$sum^k = \sum_{i=0}^{L-1} cos\left(c_i^T, c_i^k\right). \tag{11}$$

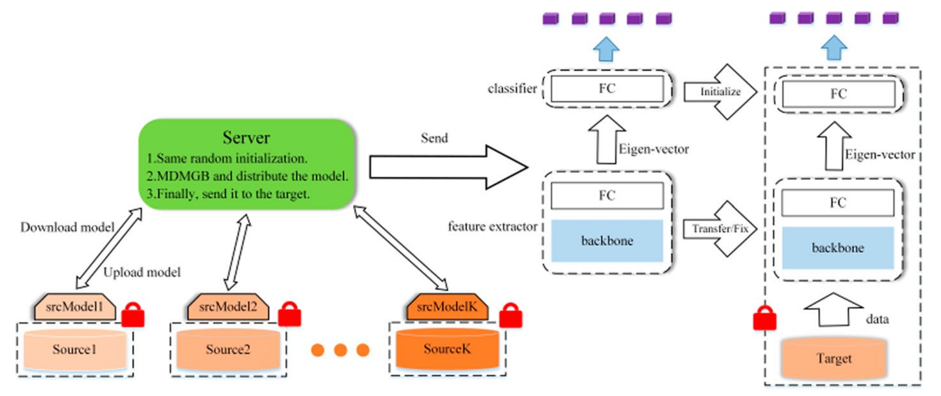

Fig. 2 SFDA algorithm framework. The figure shows function of parts and process between sections.

Since the cosine similarity ranges from $[-1, 1]$, for sum^k to be non-negative, the final sum value is:

$$sum^k = sum^k + L. \tag{12}$$

Finally, we use this sum value to calculate the weights:

$$\alpha_k = \frac{sum^k}{\sum_i^K sum^i}. \tag{13}$$

At this point, the weights obtained can be used to re-aggregate the source models.

3.3 The algorithm of SFDA

Figure 2 shows the complete algorithm framework. The algorithm flow in this paper is mainly divided into two steps: training the source model and training the target model classifier. The source model is composed of a feature learning module and classifier module (hypothesis). SFDA leaves the feature extractor fixed and uses the classifier module as the initializer of the target domain learning.

3.3.1 Training the source model

First, the model (the same model distributed by the server) is trained separately on each client (source domain) in a federated learning manner. During the training process, the model parameters are communicated once for several iterations. Label smoothing (LS) is used to improve the model's generalization ability as each source domain client trains the model locally. The standard cross-entropy loss function is

$$L_{src}\left(f_s; X_s, Y_s\right) = -E_{(x_s,y_s)\in X_s\times Y_s} \sum_{k=1}^{K} q_k \log \delta_k\left(f_s(x_s)\right). \tag{14}$$

f_s is the output of the source domain sample x_s on the source model, and q is a k-dimensional vector, all of which are "0" except that it is "1" at the position corresponding to the correct label. To improve the discriminating ability of the model, LS is applied to source model learning. With the label smoothing technique, the loss function becomes:

$$L_{src}^{ls}(f_s; X_s, Y_s) = -E_{(x_s,y_s)\in X_s\times Y_s}\sum_{k=1}^{K} q_k^{ls}\log\delta_k(f_s(x_s)), \tag{15}$$

$$q_k^{ls} = (1-\alpha)q_k + \alpha/K. \tag{16}$$

K represents the number of categories and α is the smoothing factor, which is usually set to 0.1.

During each round of communication, the client sends its own model to the server for aggregation (apply MDMGB). Finally, the server distributes the updated model to each client. This process is performed several times until the model converges. If the target domain is still unknown at this time, the unified model obtained can be considered the result of domain generalization.

The domain can be understood as a kind of interference information [30, 31]. An image in a digit recognition dataset is made up of numbers and interfering information (background colour, light intensity, line thickness, etc.). The difference between the images in different domains is caused by the interference information, while the basic characteristics of the numbers themselves remain unchanged. In addition, the learned model represents the knowledge expressed in each source domain for a specific task, and this knowledge contains interference information and corresponding basic characteristics.

In distributed training, it is very important to set the number of communication rounds r. Since the models in different source domains have different convergence rates, the models need to be aggregated r times in each iteration. Based on this, each iteration can be equally divided into r stages, and the model can be aggregated after each stage. $r=0.2$ means aggregation once every 5 iterations; $r=1$ means that it aggregates once per iteration. It has been shown that for better performance, more communication is needed during training (in general, more rounds are better). However, an unrestricted increase in the number of communication rounds not only makes the improvement of accuracy less and less obvious but it also increases the communication cost proportionally [9, 16, 19, 29].

3.3.2 Training the classifier of the target model

First, the target model is initialized with the trained source model, then the feature extractor is frozen (parameters are not updated during training), and the classifier of the target model is trained. It is worth noting that SHOT fixes the classifier when training the target model and only trains the feature extractor. We fix the feature extractor and train the target model classifier for the target domain.

The loss function of target model training is composed of two parts: information maximization (IM) loss and pseudo-labelling (PL) cross-entropy loss.

(1) Information maximization

The ideal probability vector output should be similar to the one-hot encoding, but the probability vectors should be different from each other for different classes of samples. Therefore, in this paper, information maximization loss is applied to maximize the probability of labels with maximum probability (individual certainty of predicted results), while the number of predicted labels of each category is basically the same (global diversity of predicted results). IM loss is composed of L_{ent} and L_{div}:

$$L_{ent}\left(f_t;X_t\right) = -E_{x_t\in X_t}\sum_{k=1}^{K}\delta_k\left(f_t(x_t)\right)\log\delta_k\left(f_t(x_t)\right), \tag{17}$$

$$L_{div}\left(f_t;X_t\right) = \sum_{k=1}^{K}\hat{p}_k\log\hat{p}_k, \tag{18}$$

where $\hat{p} = E_{x_t\in X_t}\left[\delta\left(f_t(x_t)\right)\right]$ is the mean value of the probability vector of the whole target domain by element operation. $L_{ent}\left(f_t;X_t\right)$ makes the prediction result of each sample more certain; $L_{div}\left(f_t;X_t\right)$ promotes the diversity of the probability vector outputs.

(2) Pseudo-labelling

Using IM losses alone can lead the model to go in the wrong direction. This negative effect comes from inaccurate network output. For example, a target domain sample from the first class that passes through the network with a probability vector of [0.34, 0.36, 0.1, 0.1, 0.1] might be forced to have an expected output [0.0, 1.0, 0.0, 0.0 0.0]. To mitigate this effect, pseudo-labels should be applied to each unlabelled sample to better supervise the training of the classifier. In this paper, a self-supervised pseudo-labelling strategy is applied to reduce the impact of such adverse factor.

Firstly, calculate the centroid of each category on the target domain as described in Subsection 3.1, and then, pseudo-labels are generated on the basis of the centroids c^T:

$$\hat{y}_t = \arg\min_k D_f\left(g_t(x_t), c_k^T\right). \tag{19}$$

Because $\hat{y}_t$ is generated by centroids generated in an unsupervised manner, $\hat{y}_t$ is called a self-supervised pseudo-label.

In summary, given the source model $f_s(x) = h_s\left(g_s(x)\right)$ and the pseudo-labels above, this paper fixes the feature extractor $g_t = g_s$ to learn the classifier h_t, and the total loss function on the target domain is as follows:

$$L\left(g_t\right) = L_{ent}\left(f_t;X_t\right) + L_{div}\left(f_t;X_t\right) - \beta E_{(x_t,\hat{y}_t)\in X_t\times\hat{Y}_t}\sum_{k=1}^{K}1_{[k=\hat{y}_t]}\log\delta_k\left(f_t(x_t)\right), \tag{20}$$

where β is greater than or equal to 0, which is used to control the weight of the pseudo-label cross-entropy loss.

In the end, only one result model is generated, and the training of the source model and target model is completely separated. After the decoupling of the two-step operation, the efficiency is substantially improved.

From the perspective of the federated optimization problem, the source domains involved here correspond to a small number of nodes that play a key role in federated learning. Moreover, the distribution differences among several source domains also fully reflect the property of non-IID in federated optimization. Both MDMGB and full SFDA can be extended to other network architectures very directly. Algorithm 1 provides the complete pseudo-code of the SFDA training process.

Algorithm 1 the complete training process of SFDA

```
Input:
    Source domains $S = \{D_S^k\}_{k=1}^K$. Target domain $T = D_T$;
    Global model $f_0$ with parameters $\theta_0$;
    The number of epochs of training source model $E$;
    The number of communication rounds per epoch $r$.
Output: The resulting model $f_T$ with the parameters $\theta_T$.
 1: //Training source model
 2: for $D_S^k$ in $S$ do
 3:     Download $(f_0, \theta_0)$ to $D_S^k$
 4:     Model initialize: $(f_S^k, \theta_S^k) \leftarrow (f_0, \theta_0)$
 5: end for
 6: for epoch in range(E) do
 7:     for $D_S^k$ in $S$ do
 8:         Train $f_S^k$ with classification(+LS) loss on $D_S^k$
 9:     end for
10:     if (epoch+1) % (1/r)==0 do
11:         Upload $\{(f_S^k, \theta_S^k)\}_{k=1}^K$ to the server
12:         $(f_S, \theta_s) \leftarrow$ MDMGB($\{(f_S^k, \theta_S^k)\}_{k=1}^K$)
13:         if epoch+1==E do
14:             Download $(f_S, \theta_s)$ to $D_T$
15:                 Go to step 22
16:         end if
17:         for $D_S^k$ in $S$ do
18:             Download $(f_S, \theta_s)$ to $D_S^k$
19:             Model update: $(f_S^k, \theta_S^k) \leftarrow (f_S, \theta_s)$
20:         end for
21:     end if
22: end for
23: //Training target model (only training the classifier)
24: Train $(f_S, \theta_s)$ with IM+PL loss on $D_T$ to get $(f_T, \theta_T)$
25: Return $(f_T, \theta_T)$
```

4 Results and discussion

This section first introduces datasets and settings for experiments. Then, we compare the SFDA with some baselines in terms of accuracy. Subsection 4.4 analyses the effects of label smoothing and pseudo-labelling. To some extent, Subsection 4.5 reflects the feasibility of SFDA in actual scenarios. Subsection 4.6 illustrates the advantages of SFDA in communication efficiency.

4.1 Datasets and settings

In this paper, SFDA was evaluated on the DigitFive and DomainNet datasets. DigitFive, as a benchmark dataset, is widely used. It contains five digit recognition datasets, namely

MNIST, MNIST-M, SYN, SVHN and USPS. DomainNet is by far the largest multi-source domain adaptation dataset, with a total of 6 domains (Clipart, Infograph, Painting, Quickdraw, Real and Sketch), 345 categories and approximately 600,000 images. Figure 3 shows some sample data (images) from DigitFive and DomainNet. Following the previous settings [13], this paper uses a three-layer CNN as the backbone network

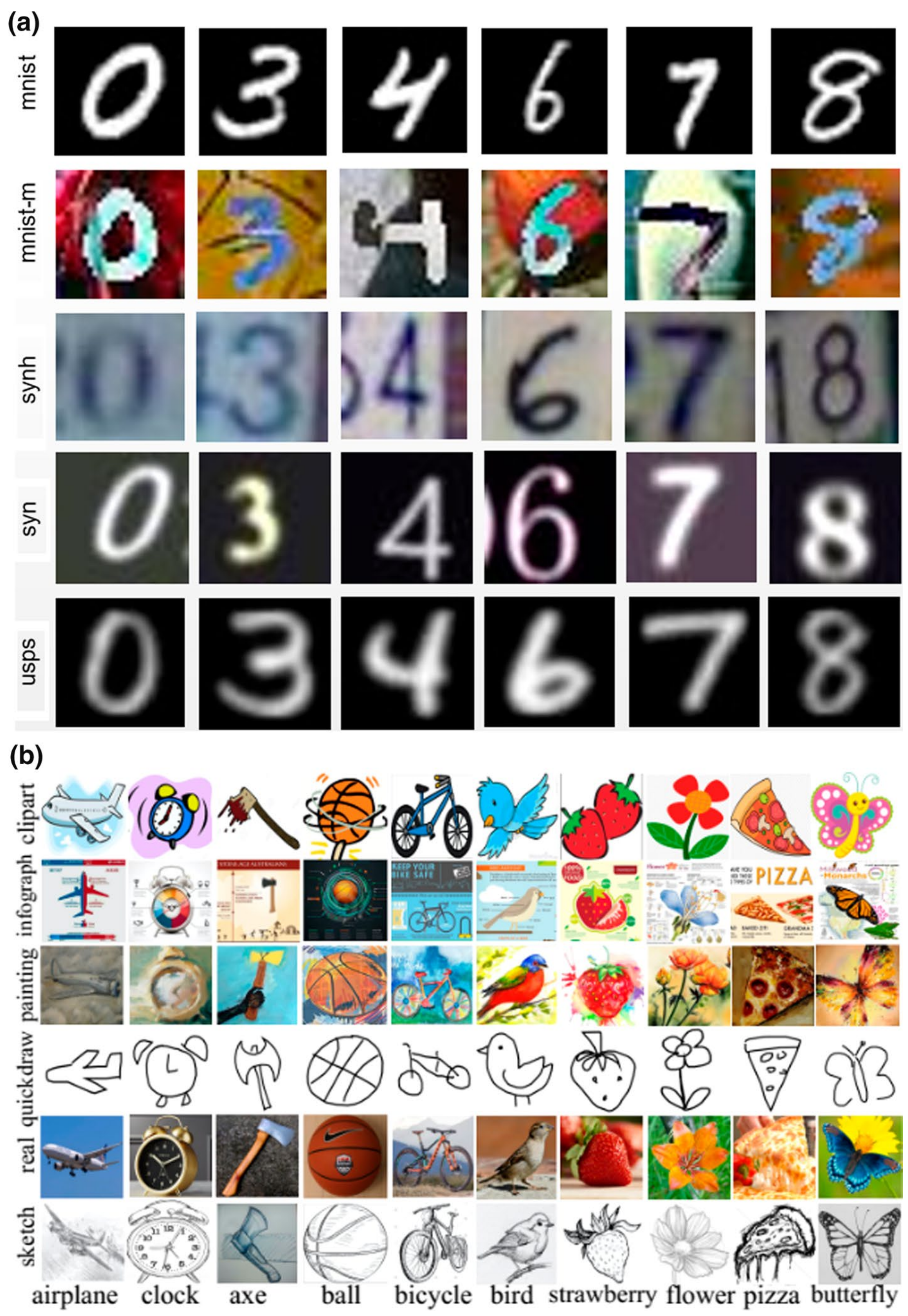

Fig. 3 Examples of images in **a** DigitFive and **b** DomainNet. DigitFive contains five digit recognition datasets. And DomainNet is by far the largest multi-source domain adaptation dataset, with a total of 6 domains (Clipart, Infograph, Painting, Quickdraw, Real, Sketch)

on DigitFive and pre-trained ResNet101 on DomainNet. In the experiments, we set each domain (dataset) as the target domain in turn and all the remaining domains as the source domains, and then calculate the single accuracy and the average accuracy.

4.2 Baselines

In this paper, the advantages of SFDA are highlighted by comparison with the following methods: DAN [18], multilayer adaptation with multi-kernel MMD; M^{3}SDA [32], dynamically aligning moments of feature distributions; DANN, based on adversarial training; FADA, advanced distributed UMDA method. This paper designed two baseline experiments without domain adaptation, namely oracle and source-only. Oracle conducts supervised learning directly on the target domain. Source-only simply combines source domains to form a hybrid domain to train a single model in a supervised learning manner.

4.3 Performance on DigitFive

Table 1 shows that SFDA exceeds FADA in both single and average accuracy. The improvements in accuracy ranged from approximately 6 percentage points to approximately 23 percentage points. In addition to the results when MNIST-M was used as the target domain, SFDA also showed great improvement in single accuracy and average accuracy compared with M^{3}SDA. For the accuracy of MNIST-M as the target domain, the difference between the two is less than 0.5%. In general, the performance of distributed UMDA is not as good as that of the common multi-source domain adaptation method due to the inability to obtain both source domain and target domain data at the same time and the limitations of communication. However, the accuracy of SFDA not only exceeds the distributed unsupervised multi-source domain adaptation method FADA to a large extent but also exceeds the common multi-source domain adaptation method M^{3}SDA in general.

Table 1 Accuracy (%) of unsupervised multi-source domain adaptation on DigitFive

Methods	**Target**					
	Mnist	**Mnist-m**	**Svhn**	**Syn**	**Usps**	**Avg**
Upper and lower bounds						
Oracle	99.5	95.4	92.3	98.7	99.2	97.0
Source-only	92.3	63.7	71.5	83.4	90.71	80.3
Common multi-source domain adaptation						
DAN	96.3	63.8	72.5	85.4	94.2	82.4
M^{3}SDA	98.4	72.8	81.3	89.6	96.2	87.7
DANN	97.6	71.3	63.5	85.4	92.33	82.1
Decentralized multi-source domain adaptation						
FADA	92.5	64.5	63.5	82.8	91.7	79.0
SFDA	**99.10**	**72.31**	**86.02**	**90.37**	**98.06**	**89.17**

Bold value represents the current highest accuracy

Table 2 Influence of label smoothing and pseudo-labelling on the accuracy (%) in DigitFive

Methods	Target					
	Mnist	**Mnist-m**	**Svhn**	**Syn**	**Usps**	**Avg**
No label smoothing; No pseudo-labelling. (learning rate is 0.05)						
SFDA-LS-PL	99.19	74.53	83.13	89.42	97.74	88.80
No label smoothing. (learning rate is 0.01)						
SFDA-LS	99.26	74.09	84.80	89.29	97.69	89.03
Standard settings. (learning rate is 0.01)						
SFDA	99.10	72.31	86.02	90.37	98.06	89.17

Bold value represents the current highest accuracy

Table 3 Accuracy (%) of unsupervised multi-source domain adaptation on DomainNet

Methods	Target						
	Clipart	**Infograph**	**Painting**	**Quickdraw**	**Real**	**Sketch**	**Avg**
Upper and lower bounds							
Oracle	69.3	34.5	66.3	66.8	80.1	60.7	63.0
Source-only	47.6	13.0	38.1	13.3	51.9	33.7	32.9
Common multi-source domain adaptation							
DAN	48.4	14.8	40.2	15.3	53.9	34.0	34.5
M^{3}SDA	58.6	26.0	52.3	6.3	62.7	49.5	42.6
DANN	52.5	11.1	42.0	14.7	52.9	38.1	35.2
Decentralized multi-source domain adaptation							
FADA	52.3	16.3	41.9	**13.9**	**52.7**	36.8	35.7
SFDA	**57.67**	**17.57**	**45.69**	13.08	51.65	**48.68**	**39.06**

Bold value represents the current highest accuracy

4.4 Influence of label smoothing and pseudo-labelling

To verify the effectiveness of the label smoothing technique and pseudo-label technique in this paper, we designed a comparative experiment, and the results are shown in Table 2. According to the results in Table 2, the average accuracy increased from 88.80% to 89.03% after the application of pseudo-labelling technology. Without the use of pseudo-labelling technology, the accuracy on SVHN is significantly reduced. As mentioned earlier, models can go in the wrong direction. The accuracy of the model without LS was 89.03%, while the accuracy of the model with label smoothing was 89.17% because the standard cross-entropy loss function would make the source model overfit the source domain, which would lead to the decline of the generalization ability of the model. Following the practice of SHOT [29], that is, only the feature extractor is trained on the target domain, the performance of the target model degrades very seriously. The learning rate in the experiment is the best value obtained by multiple experiments; that is, under the current setting, the value of the learning rate brings the highest accuracy. However, this article does not exhaust all possible values.

Table 4 Comparison of time consumption between FADA and SFDA

Total_epoch	45	90	135	180
FADA	15,660	31,320	46,980	62,640
SFDA	18	36	54	72

4.5 Performance on DomainNet

To fully verify the generalization performance of the model, we selected a large DomainNet dataset to conduct experiments and simulated six scenes: Clipart, Infograph, Painting, Quickdraw, Real and Sketch. The experimental results are shown in Table 3. The models are compared vertically and the results horizontally when each dataset takes turns as the target domain. Table 3 shows that, similarly, SFDA significantly exceeds FADA in terms of average accuracy. The SFDA performs better on 4 of 6 tasks. For the other two tasks, Quickdraw as the target domain and Real as the target domain, the SFDA in this paper is only about 1% behind FADA in accuracy. The experimental results on DomainNet provide a reference for the practical application of the algorithm in this paper because its data volume and the number of categories are very large and the image itself is complex enough.

4.6 Comparison of communication cost

Table 4 shows the change in communication times with the total number of epochs when batch_size = 64 and MNIST-M is the target domain. It is easy to see from this that the number of communications required by SFDA is significantly smaller and that the number of communications required by FADA is around three orders of magnitude larger than that of SFDA.

5 Conclusions and future work

This paper proposes a distributed UMDA approach under a federated learning framework. A unified source model is obtained by aggregating the learned models in multiple source domains (a label smoothing technique is used for the training of the models in each source domain), and the target model is initialized by using the model. Then, the classifier of the target model is trained in a self-supervised way by using the information maximization and pseudo-labelling technique. SFDA takes into account both data privacy protection and communication efficiency while achieving ideal accuracy. Through the comparison with the existing algorithms, both the communication efficiency and the accuracy are greatly improved. In future work, we hope to design a more effective weighting strategy to solve the effect of self-supervised learning on the target domain largely depends on the quality of the source model.

Appendix A

If each source domain model can be adequately trained, they can represent almost perfectly their own source domain, which makes the size of the data volume meaningless in the weighting process. However, when the source models are aggregated in the early stages of training, they may not be fully trained due to the small number of iterations. Therefore, this part explains the defects of using data volume from another aspect. If the closeness of each source domain to the target domain has been quantified as $\{c_1, c_2, \ldots, c_K\}$ and the larger the value, the closer the relationship between the domains is, and the data volume of each source domain happens to be $\left\{\frac{\prod_{i=1}^K c_i}{c_1}, \frac{\prod_{i=1}^K c_i}{c_2}, \ldots, \frac{\prod_{i=1}^K c_i}{c_K}\right\}$. In this case, if the weight of a source domain is calculated in the following way, the size of the data volume just offsets the effect of the tightness:

$$\alpha_k = \frac{c_k \times \frac{\prod_{i=1}^K c_i}{c_k}}{\sum_{j=1}^K c_j \times \frac{\prod_{i=1}^K c_i}{c_j}} = \frac{\prod_{i=1}^K c_i}{\sum_{j=1}^K \prod_{i=1}^K c_i} = \frac{1}{K}.$$

Abbreviations

SFDA: Self-supervised federated domain adaptation; MDMGB: Multi-domain model generalization balance; UMDA: Unsupervised multi-source domain adaptation; FADA: Federated adversarial domain adaptation; UFDA: Unsupervised federated domain adaptation; MMD: Maximum mean discrepancy; CORAL: Correlation alignment; HoMM: Higher-order moment matching for unsupervised domain adaptation; DANN: Domain adversarial neural network; RKHS: Reproducing kernel Hilbert space; FL: Federated learning; FedAvg: Federated average; KD3A: Knowledge distillation-based decentralized domain adaptation; SHOT: Source hypothesis transfer for unsupervised domain adaptation; IM: Information maximization; PL: Pseudo-labelling.

Acknowledgements

The authors thank the reviewers and editors for their efforts in valuable comments and suggestions.

Authors' contributions

All authors have contributed equally. All authors have read and approved the final manuscript.

Funding

This work was supported in part by the National Natural Science Foundation of China (62072469), National Key R&D Program (2018YFE0116700), Natural Science Foundation of Shandong Province (ZR2019MF049) and Fundamental Research Funds for the Central Universities (2015020031).

Availability of data and materials

The datasets used are public. Digit-Five and DomainNet can be downloaded from https://drive.google.com/open?id=1A4RJOFj4BJkmliiEL7g9WzNIDUHLxfmm and http://ai.bu.edu/M3SDA/, respectively.

Declarations

Competing interests

The authors declare that they have no competing interests.

Author details

[1]College of Computer Science and Technology, China University of Petroleum (East), QingDao 266580, China. [2]School of Public Affairs, ZheJiang University, HangZhou 310000, China. [3]University of Oulu, Oulu, Finland. [4]Suzhou Tongji Blockchain Research Institute, SuZhou 215000, China.

Received: 7 September 2021 Accepted: 10 March 2022
Published online: 15 April 2022

References

1. T. Song, Z. Wang, P. Xie et al., A novel dual path gated recurrent unit model for sea surface salinity prediction. J. Atmos. Oceanic Tech. **37**(2), 317–325 (2020)
2. F. Meng, T. Song, D. Xu et al., Forecasting tropical cyclones wave height using bidirectional gated recurrent unit. Ocean Eng. **234**, 108795 (2021)
3. S. Pang, P. Xie, D. Xu et al., NDFTC: a new detection framework of tropical cyclones from meteorological satellite images with deep transfer learning. Remote Sens. **13**, 1860 (2021)
4. T. Song, J. Jiang, W. Li et al., A deep learning method with merged LSTM neural networks for SSHA prediction. IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens. **13**, 2853–2860 (2020)
5. T. Song, N. Han, Y. Zhu et al., Application of deep learning technique to the sea surface height prediction in the South China Sea. Acta Oceanol. Sin. **40**, 1–8 (2021)
6. G. Hinton, O. Vinyals, J. Dean, Distilling the knowledge in a neural network. Comput. Sci. **14**(7), 38–39 (2015)

7. W.M. Kouw, M. Loog, A review of domain adaptation without target labels. IEEE Trans. Pattern Anal. Mach. Intell. **43**(3), 766–785 (2021)
8. V. Mothukuri et al., A survey on security and privacy of federated learning. Futur. Gener. Comput. Syst. **115**, 619–640 (2021)
9. H.B. Mcmahan, E. Moore, D. Ramage, et al. Communication-efficient learning of deep networks from decentralized data. in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. 54:1273–1282 (2017)
10. X. Peng, Z. Huang, Y. Zhu, et al. Federated adversarial domain adaptation. arXiv:abs/1911.02054 (2019)
11. S. Ben-David, J. Blitzer, K. Crammer et al., A theory of learning from different domains. Mach. Learn. **79**(1–2), 151–175 (2020)
12. M. Ghifary, W. Kleijn, M. Zhang. Domain adaptive neural networks for object recognition. arXiv:abs/1409.6041 (2014)
13. E. Tzeng, J. Hoffman, N. Zhang, et al. Deep domain confusion: maximizing for domain invariance. arXiv:abs/1412.3474 (2014)
14. B. Sun, K. Saenko. Deep CORAL: correlation alignment for deep domain adaptation. arXiv:abs/1607.01719 (2016)
15. C. Chen, Z. Fu, Z. Chen et al., HoMM: higher-order moment matching for unsupervised domain adaptation. Proc. AAAI Conf. Artif. Intell. **34**(4), 3422–3429 (2020)
16. T. Song, X. Zeng, P. Zheng, M. Jiang, A. Rodriguez-Paton, A parallel workflow pattern modeling using spiking neural P systems with colored spikes. IEEE Trans Nanobioscience. **17**(4), 474–484 (2018)
17. X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko and B. Wang, Moment Matching for Multi-Source Domain Adaptation, 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 1406–1415 (2019)
18. M. Long, Y. Cao, Z. Cao, J. Wang, M.I. Jordan, Transferable representation learning with deep adaptation networks. IEEE Trans Pattern Anal Mach Intell. **41**(12), 3071–3085 (2019)
19. T. Song, L. Pan, T. Wu, P. Zheng, M.L.D. Wong, A. Rodríguez-Patón, Spiking neural P systems with learning functions. IEEE Trans. Nanobiosci. **18**(2), 176–190 (2019)
20. Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, V. Lempitsky, Domain-adversarial training of neural networks. J. Mach. Learn. Res. **17**(1), 2096–2030 (2016)
21. K. Saito, K. Watanabe, Y. Ushiku, T. Harada. Maximum classifier discrepancy for unsupervised domain adaptation. in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3723–3732 (2018)
22. H. Zhao, S. Zhang, G. Wu, J.P. Costeira, J.M.F. Moura, G.J. Gordon. Multiple source domain adaptation with adversarial training of neural networks. arXiv:abs/1705.09684 (2017)
23. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio. Generative adversarial networks. Advances in Neural Information Processing Systems. 3. https://doi.org/10.1145/3422622 (2014)
24. S. Seo, Y. Suh, D. Kim, et al. Learning to optimize domain specific normalization for domain generalization. ECCV, 68–83 (2020)
25. D Mahajan, S. Tople, A. Sharma. Domain generalization using causal matching. In *International Conference on Machine Learning (ICML)* (2020)
26. V. Mothukuri, P. Khare, R. Parizi, S. Pouriyeh, A. Dehghantanha, G. Srivastava. Federated learning-based anomaly detection for IoT security attacks. IEEE Internet Things J. 2327–4662 (2021)
27. H. Feng, Z. You, M. Chen, T.-Y. Zhang, M. Zhu, F. Wu, C. Wu, W. Chen. KD3A: unsupervised multi-source decentralized domain adaptation via knowledge distillation. ICML (2021)
28. J. Liang, D. Hu, J. Feng. Do we really need to access the source data? Source hypothesis transfer for unsupervised domain adaptation. arXiv:abs/2002.08546 (2021)
29. T. Song, S. Pang, S. Hao et al., A parallel image skeletonizing method using spiking neural P systems with weights. Neural Process Lett. **50**, 1485–1502 (2019)
30. T. Song, P. Zheng, M.L.D. Wong, M. Jiang, X. Zeng, On the computational power of asynchronous axon membrane systems. IEEE Trans. Emerg. Top. Comput. Intell. **4**(5), 696–704 (2020)
31. T. Song, A. Rodríguez-Patón, P. Zheng, X. Zeng, Spiking neural P systems with colored spikes. IEEE Trans. Cognit. Dev. Syst. **10**(4), 1106–1115 (2018)
32. K. Zhou, Y. Yang, Y. Qiao, T. Xiang, Domain adaptive ensemble learning. IEEE Trans. Image Process. **30**, 8008–8018 (2021)

Publisher's Note