KD3A: Unsupervised Multi-Source Decentralized Domain Adaptation via Knowledge Distillation

Hao-zhe Feng [1] **Zhaoyang You** [2] **Minghao Chen** [1] **Tianye Zhang** [1] **Minfeng Zhu** [1]
Fei Wu [2] **Chao Wu** [3] **Wei Chen** [1]

Abstract

Conventional unsupervised multi-source domain adaptation (UMDA) methods assume all source domains can be accessed directly. However, this assumption neglects the privacy-preserving policy, where all the data and computations must be kept decentralized. There exist three challenges in this scenario: (1) Minimizing the domain distance requires the pairwise calculation of the data from source and target domains, while the data on the source domain is not available. (2) The communication cost and privacy security limit the application of existing UMDA methods, such as the domain adversarial training. (3) Since users cannot govern the data quality, the irrelevant or malicious source domains are more likely to appear, which causes negative transfer. To address the above problems, we propose a privacy-preserving UMDA paradigm named **K**nowledge **D**istillation based **D**ecentralized **D**omain **A**daptation (KD3A), which performs domain adaptation through the knowledge distillation on models from different source domains. The extensive experiments show that KD3A significantly outperforms state-of-the-art UMDA approaches. Moreover, the KD3A is robust to the negative transfer and brings a $100\times$ reduction of communication cost compared with other decentralized UMDA methods.

1. Introduction

Most deep learning models are trained with large-scale datasets via supervised learning. Since it is often costly to get sufficient data, we usually use other similar datasets to train the model. However, due to the domain shift, naively combining different datasets often results in unsatisfying performance. **U**nsupervised **M**ulti-source **D**omain **A**daptation (UMDA) (Zhang et al., 2015) addresses such problems by establishing transferable features from multiple source domains to an unlabeled target domain.

Recent advanced UMDA methods (Chang et al., 2019; Zhao et al., 2020) perform the knowledge transfer within two steps: (1) Combining data from source and target domains to construct **S**ource-**T**arget pairs. (2) Establishing transferable features by minimizing the $\mathcal{H}$-divergence. This prevailing paradigm works well when all source domains are available. However, in terms of the privacy-preserving policy, we cannot access the sensitive data such as the patient data from different hospitals and the client profiles from different companies. In these cases, all the data and computations on source domains must be kept decentralized.

Most conventional UMDA methods are not applicable under this privacy-preserving policy due to three problems: (1) Minimizing the $\mathcal{H}$-divergence in UMDA requires the pairwise calculation of the data from source and target domains, while the data on source domain is not available. (2) The communication cost and privacy security limit the application of advanced UMDA methods. For example, the domain adversarial training is able to optimize the $\mathcal{H}$-divergence without accessing data (Peng et al., 2020). However, it requires each source domain to synchronize model with target domain after every single batch, which results in huge communication cost and causes the privacy leakage (Zhu et al., 2019). (3) The negative transfer problem (Pan & Yang, 2010). Since it is difficult to govern the data quality, there can exist some irrelevant source domains that are very different from the target domain or even some malicious source domains which perform the poisoning attack (Bagdasaryan et al., 2020). With these bad domains, the negative transfer occurs.

In this study, we propose a solution to the above problems, **K**nowledge **D**istillation based **D**ecentralized **D**omain **A**daptation (KD3A), which aims to perform decentralized domain adaptation through the knowledge distillation on models from different source domains. Our KD3A approach

---

[1]State Key Lab of CAD&CG, Zhejiang University, Hangzhou, China [2]College of Computer Science and Technology, Zhejiang University, Hangzhou, China [3]School of Public Affairs, Zhejiang University, Hangzhou, China. Correspondence to: Wei Chen <chenvis@zju.edu.cn>.

consists of three components used in tandem. First, we propose a multi-source knowledge distillation method named *Knowledge Vote* to obtain high-quality domain consensus. Based on the consensus quality of different source domains, we devise a dynamic weighting strategy named *Consensus Focus* to identify the malicious and irrelevant source domains. Finally, we derive a decentralized optimization strategy of $\mathcal{H}$-divergence named *BatchNorm MMD*. Moreover, we analyze the decentralized generalization bound for KD3A from a theoretical perspective. The extensive experiments show our KD3A has the following advantages:

- The KD3A brings a $100\times$ reduction of communication cost compared with other decentralized UMDA methods and is robust to the privacy leakage attack.

- The KD3A assigns low weights to those malicious or irrelevant domains. Therefore, it is robust to negative transfer.

- The KD3A significantly outperforms the *state-of-the-art* UMDA approaches with 51.1% accuracy on the large-scale DomainNet dataset.

In addition, our KD3A is easy to implement and we create an open-source framework to conduct KD3A on different benchmarks.

2. Related work

2.1. Unsupervised Multi-Source Domain Adaptation

Unsupervised Multi-source Domain Adaptation (UMDA) establish the transferable features by reducing the $\mathcal{H}$-divergence between source domain $\mathbb{D}_S$ and target domain $\mathbb{D}_T$. There are two prevailing paradigms that provide the optimization strategy of $\mathcal{H}$-divergence, i.e. maximum mean discrepancy (MMD) and the adversarial training. In addition, knowledge distillation is also used to perform model-level knowledge transfer.

MMD based methods (Tzeng et al., 2014) construct a reproducing kernel Hilbert space (RKHS) $\mathcal{H}_\kappa$ with the kernel κ, and optimize the $\mathcal{H}$-divergence by minimizing the MMD distance $d_{\text{MMD}}^\kappa(\mathbb{D}_S, \mathbb{D}_T)$ on $\mathcal{H}_\kappa$. Using the kernel trick, MMD can be computed as

$$d_{\text{MMD}}^\kappa(\mathbb{D}_S, \mathbb{D}_T) = -2\mathbb{E}_{\mathbf{X}_S, \mathbf{X}_T \sim \mathbb{D}_S, \mathbb{D}_T} \kappa(\mathbf{X}_S, \mathbf{X}_T) + \\ \mathbb{E}_{\mathbf{X}_S, \mathbf{X}'_S \sim \mathbb{D}_S} \kappa(\mathbf{X}_S, \mathbf{X}'_S) + \mathbb{E}_{\mathbf{X}_T, \mathbf{X}'_T \sim \mathbb{D}_T} \kappa(\mathbf{X}_T, \mathbf{X}'_T) \quad (1)$$

Recent works propose the variations of MMD, e.g., multi-kernel MMD (Long et al., 2015), class-weighted MMD(Yan et al., 2017) and domain-crossing MMD (Peng et al., 2019). However, all these methods require the pairwise calculation of the data from source and target domains, which is not allowed under the decentralization constraints.

The adversarial training strategy (Saito et al., 2018; Zhao et al., 2018) apply adversarial training in feature space to optimize $\mathcal{H}$-divergence. It is proved that with the adversarial training strategy, the UMDA model can work under the privacy-preserving policy (Peng et al., 2020). However, the adversarial training requires each source domain to exchange and update model parameters with the target domain after every single batch, which consumes huge communication resources.

Knowledge distillation in domain adaptation. Self-supervised learning has many applications (Zhang et al., 2021; Chen et al., 2021) in the label-lacking scenarios. Knowledge distillation (KD) (Hinton et al., 2015; Chen et al., 2020) is an efficient self-supervised method to transfer knowledge between different models. Recent works (Meng et al., 2018; Zhou et al., 2020) extend the knowledge distillation into domain adaptation with a teacher-student training strategy: training multiple teacher models on source domains and ensembling them on target domain to train a student model. This strategy outperforms other UMDA method in practice. However, due to the irrelevant and malicious source domains, the conventional KD strategies may fail to obtain proper knowledge.

2.2. Federated Learning

Federated learning (Konecný et al., 2016) is a distributed machine learning approach, it can train a global model by aggregating the updates of local models from multiple decentralized datasets. Recent works (McMahan et al., 2017) find a trade-off between model performance and communication efficiency, that is, to make the global model achieve better performance, we need to conduct more communication rounds, which raises the communication costs. Besides, the frequent communication will also cause privacy leakage (Wang et al., 2019), making the training process insecure.

Federated domain adaptation. There are few works discussing the decentralized UMDA methods. FADA (Peng et al., 2020) first raises the concept of federated domain adaptation. It applies the adversarial training to optimize the $\mathcal{H}$-divergence without accessing data. However, FADA consumes high communication costs and is vulnerable to the privacy leakage attack. Model Adaptation (Li et al., 2020) and SHOT (Liang et al., 2020) provide source-free methods to solve the single source decentralized domain adaptation. However, they are vulnerable to the negative transfer in multi-source situations.

3. KD3A: Decentralized Domain Adaptation via Knowledge Distillation

Preliminary. Let $\mathbb{D}_S$ and $\mathbb{D}_T$ denote the source domain and target domain. In UMDA, we have K *source* domains

$\{\mathbb{D}_S^k\}_{k=1}^K$ where each domain contains N_k labeled examples as $\mathbb{D}_S^k := \{(\mathbf{X}_i^k, \mathbf{y}_i^k)\}_{i=1}^{N_k}$ and a *target* domain $\mathbb{D}_T$ with N_T unlabeled examples as $\mathbb{D}_T := \{\mathbf{X}_i^T\}_{i=1}^{N_T}$. The goal of UMDA is to learn a model h which can minimize the task risk $\epsilon_{\mathbb{D}_T}$ in $\mathbb{D}_T$, i.e. $\epsilon_{\mathbb{D}_T}(h) = \Pr_{(\mathbf{X},\mathbf{y})\sim\mathbb{D}_T}[h(\mathbf{X}) \neq \mathbf{y}]$. Without loss of generality, we consider C-way classification task and assume the target domain shares the same tasks with the source domains. In a common UMDA, we combine K source domains with different domain weights as $\boldsymbol{\alpha}$, and perform domain adaptation by minimizing the following generalization bound (Ben-David et al., 2010; Zhao et al., 2018) with the multiple source domains as:

Theorem 1 *Let $\mathcal{H}$ be the model space, $\{\epsilon_{\mathbb{D}_S^k}(h)\}_{k=1}^K$ and $\epsilon_{\mathbb{D}_T}(h)$ be the task risks of source domains $\{\mathbb{D}_S^k\}_{k=1}^K$ and the target domain $\mathbb{D}_T$, and $\boldsymbol{\alpha} \in \mathcal{R}_+^K, \sum_{k=1}^K \boldsymbol{\alpha}_k = 1$ be the domain weights. Then for all $h \in \mathcal{H}$ we have:*

$$\epsilon_{\mathbb{D}_T}(h) \leq \sum_{k=1}^K \boldsymbol{\alpha}_k \left(\epsilon_{\mathbb{D}_S^k}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{D}_S^k, \mathbb{D}_T) \right) + \lambda_0 \tag{2}$$

where λ_0 is a constant according to the task risk of the optimal model on the source domains and target domain.

Problem formulation for decentralized scenarios. In decentralized UMDA, the data from K source domains are stored locally and are not available. The accessible information in each communication round includes: (1) The size of the training sets $\{N_S^k\}_{k=1}^K$ on source domains and the parameters of K models $\{h_S^k\}_{k=1}^K$ trained on these source domains. (2) The target domain data containing N_T unlabeled examples as $\mathbb{D}_T := \{\mathbf{X}_i^T\}_{i=1}^{N_T}$. In KD3A, we apply knowledge distillation to perform domain adaptation without accessing the data.

3.1. Extending Source Domains With Consensus Knowledge

Knowledge distillation can perform knowledge transfer through different models. Suppose we have K fully-trained models from K source domains denoted by $\{h_S^k\}_{k=1}^K$. we use $q_S^k(\mathbf{X})$ to denote the confidence for each class and use the class with the maximum confidence as label, i.e. $h_S^k(\mathbf{X}) = \arg_c \max[q_S^k(\mathbf{X})]_c$. As shown in Figure 1(a), the knowledge distillation in UMDA consists of two steps. First, for each target domain data $\mathbf{X}_i^T$, we obtain the inferences of the source domain models. Then, we use the ensemble method to get the consensus knowledge of the source models, e.g., $\mathbf{p}_i = \frac{1}{K} \sum_{k=1}^K q_S^k(\mathbf{X}_i^T)$. In order to utilize the consensus knowledge for domain adaptation, we define an extended source domain $\mathbb{D}_S^{K+1}$ with the consensus knowledge $\mathbf{p}_i$ for each target domain data $\mathbf{X}_i^T$ as

$$\mathbb{D}_S^{K+1} = \{(\mathbf{X}_i^T, \mathbf{p}_i)\}_{i=1}^{N_T}$$
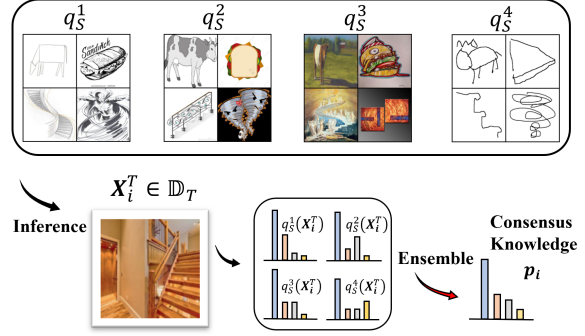
(a) Knowledge distillation process in UMDA.
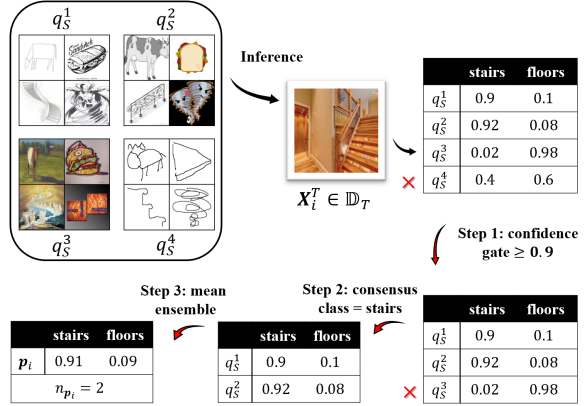
(b) Knowledge vote ensemble.

Figure 1. (a) Knowledge distillation in UMDA consists of two steps: obtaining the inferences from source domain models and performing knowledge ensemble to get the consensus knowledge. (b) Our knowledge vote extracts strong consensus knowledge with 3 steps: confidence gate, consensus class vote and mean ensemble. '×' means the eliminated model in each step.

We also define the related task risk for $\mathbb{D}_S^{K+1}$ as

$$\epsilon_{\mathbb{D}_S^{K+1}}(h) = \Pr_{(\mathbf{X},\mathbf{p})\sim\mathbb{D}_S^{K+1}}[h(\mathbf{X}) \neq \arg_c \max \mathbf{p}_c].$$

With this new source domain, we can train the source model h_S^{K+1} through the knowledge distillation loss as

$$L^{\mathrm{kd}}(\mathbf{X}_i^T, q_S^{K+1}) = D_{\mathrm{KL}}(\mathbf{p}_i \| q_S^{K+1}(\mathbf{X}_i^T)). \tag{3}$$

In decentralized UMDA, we get the target model as the aggregation of the source models, i.e. $h_T := \sum_{k=1}^{K+1} \boldsymbol{\alpha}_k h_S^k$. A common question is, how does the new model h_S^{K+1} improve the UMDA performance? It is easy to find that minimizing KD loss (3) leads to the optimization of $\epsilon_{\mathbb{D}_S^{K+1}}(h)$ (proof in Appendix A). With this insight, we can derive the generalization bound for knowledge distillation as follows (proof in Appendix B):

Proposition 1 (*The generalization bound for knowledge distillation*). *Let $\mathcal{H}$ be the model space and $\epsilon_{\mathbb{D}_S^{K+1}}(h)$ be*

the task risk of the new source domain $\mathbb{D}_S^{K+1}$ based on knowledge distillation. Then for all $h_T \in \mathcal{H}$, we have:

$$\epsilon_{\mathbb{D}_T}(h_T) \leq \epsilon_{\mathbb{D}_S^{K+1}}(h_T) + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{D}_S^{K+1}, \mathbb{D}_T)$$
$$+ \min\{\lambda_1, \sup_{h \in \mathcal{H}} |\epsilon_{\mathbb{D}_S^{K+1}}(h) - \epsilon_{\mathbb{D}_T}(h)|\} \quad (4)$$

where λ_1 is a constant for the task risk of the optimal model.

3.2. Knowledge Vote: Producing Good Consensus

Proposition 1 shows the new source domain $\mathbb{D}_S^{K+1}$ will improve the generalization bound if the consensus knowledge is good enough to represent the ground-truth label, i.e. $\sup_{h \in \mathcal{H}} |\epsilon_{\mathbb{D}_T}(h) - \epsilon_{\mathbb{D}_S^{K+1}}(h)| \leq \lambda_1$. However, due to the irrelevant and malicious source domains, the conventional ensemble strategies (e.g., maximum and mean ensemble) may fail to obtain proper consensus. Therefore, we propose the *Knowledge Vote* to provide high-quality consensus.

The main idea of knowledge vote is that if a certain consensus knowledge is supported by more source domains with high confidence (e.g., > 0.9), then it will be more likely to be the true label. As shown in Figure 1(b), it takes three steps to perform *Knowledge Vote*:

1. **Confidence gate**. For each $\mathbf{X}_i^T \in \mathbb{D}_T$, we firstly use a high-level confidence gate to filter the predictions $\{q_S^k(\mathbf{X}_i^T)\}_{k=1}^K$ of teacher models and eliminate the unconfident models.

2. **Consensus class vote**. For the models remained, the predictions are added up to find the consensus class which has the maximum value. Then we drop the models that are inconsistent with the consensus class.

3. **Mean ensemble**. After the class vote, we obtain a set of models that all support the consensus class. Finally, we get the consensus knowledge $\mathbf{p}_i$ by conducting the mean ensemble on these supporting models. We also record the number of domains that support $\mathbf{p}_i$, denoted by $n_{\mathbf{p}_i}$. For those $\mathbf{X}^T$ with all teacher models eliminated by the confidence gate, we simply use the mean ensemble to get $\mathbf{p}$ and assign a relatively low weight to them as $n_{\mathbf{p}} = 0.001$.

After *Knowledge Vote*, we obtain the new source domain $\mathbb{D}_S^{K+1} = \{(\mathbf{X}_i^T, \mathbf{p}_i, n_{\mathbf{p}_i})\}_{i=1}^{N_T}$. We use the $n_{\mathbf{p}_i}$ to re-weight the knowledge distillation loss as

$$L^{\mathrm{kv}}(\mathbf{X}_i^T, q) = n_{\mathbf{p}_i} \cdot D_{\mathrm{KL}}(\mathbf{p}_i \| q(\mathbf{X}_i^T)) \quad (5)$$

Compared with other ensemble strategies, our *Knowledge Vote* makes model learn high-quality consensus knowledge since we assign high weights to those items with high confidence and many support domains.

3.3. Consensus Focus: Against Negative Transfer

Domain weights $\boldsymbol{\alpha}$ determine the contribution of each source domain. Ben-David et al. (2010) proves the optimal $\boldsymbol{\alpha}$ should be proportional to the amount of data when all source domains are equally important. However, this condition is hard to satisfy in KD3A since some source domains are usually very different from the target domain, or even malicious domains with corrupted labels. These bad domains lead to negative transfer. One common solution (Zhao et al., 2020) is to re-weight each source domain with the $\mathcal{H}$-divergence as

$$\boldsymbol{\alpha}_k = N_k e^{-d_{\mathcal{H}}(\mathbb{D}_S^k, \mathbb{D}_T)} / \sum_k N_k e^{-d_{\mathcal{H}}(\mathbb{D}_S^k, \mathbb{D}_T)}. \quad (6)$$

However, calculating $\mathcal{H}$-divergence requires to access the source domain data. Besides, $\mathcal{H}$-divergence only measures the domain similarity on the input space, which does not utilize the label information and fails to identify the malicious domain. Reasonably, we propose *Consensus Focus* to identify those irrelevant and malicious domains. As mentioned in *Knowledge Vote*, the UMDA performance is related to the quality of consensus knowledge. With this motivation, the main idea of *Consensus Focus* is to assign high weights to those domains which provide high-quality consensus and penalize those domains which provide bad consensus. To perform *Consensus Focus*, we first derive the definition of consensus quality and then calculate the contribution to the consensus quality for each source domain.

The definition of consensus quality. Suppose we have a set of source domains denoted by $\mathcal{S} = \{\mathbb{D}_S^k\}_{k=1}^K$. For each coalition of source domains $\mathcal{S}', \mathcal{S}' \subseteq \mathcal{S}$, we want to estimate the quality of the knowledge consensus obtained from $\mathcal{S}'$. Generally speaking, if one consensus class is supported by more source domains with higher confidence, then it will be more likely to represent the true label, which means the consensus quality gets better. Therefore, for each $\mathbf{X}_i^T \in \mathbb{D}_T$ with the consensus knowledge $(\mathbf{p}_i(\mathcal{S}'), n_{\mathbf{p}_i}(\mathcal{S}'))$ obtained from $\mathcal{S}'$, We define the related consensus quality as $n_{\mathbf{p}_i}(\mathcal{S}') \cdot \max \mathbf{p}_i(\mathcal{S}')$ and the total consensus quality Q is

$$Q(\mathcal{S}') = \sum_{\mathbf{X}_i^T \in \mathbb{D}_T} n_{\mathbf{p}_i}(\mathcal{S}') \cdot \max \mathbf{p}_i(\mathcal{S}') \quad (7)$$

With the consensus quality defined in (7), we derive the consensus focus (CF) value to quantify the contribution of each source domain as

$$\mathrm{CF}(\mathbb{D}_S^k) = Q(\mathcal{S}) - Q(\mathcal{S} \setminus \{\mathbb{D}_S^k\}) \quad (8)$$

$\mathrm{CF}(\mathbb{D}_S^k)$ describes the marginal contribution of the single source domain $\mathbb{D}_S^k$ to the consensus quality of all source domains $\mathcal{S}$. If one source domain is a bad domain, then removing it will not decrease the total quality Q, which leads to a low consensus focus value. With the CF value, we can

assign proper weights to different source domains. Since we introduce a new source domain $\mathbb{D}_S^{K+1}$ in *Knowledge Vote*, we compute the domain weights with two steps. First, we obtain $\boldsymbol{\alpha}_{K+1} = N_T/(\sum_{k=1}^{K} N_k + N_T)$ for $\mathbb{D}_S^{K+1}$ based on the amount of data. Then we use the CF value to re-weight each original source domain as

$$\boldsymbol{\alpha}_k^{\text{CF}} = (1 - \boldsymbol{\alpha}_{K+1}) \cdot \frac{N_k \cdot \text{CF}(\mathbb{D}_S^k)}{\sum_{k=1}^{K} N_k \cdot \text{CF}(\mathbb{D}_S^k)} \qquad (9)$$

Compared with the re-weighting strategy in (6), our *Consensus Focus* has two advantages. First, the calculation of $\boldsymbol{\alpha}^{\text{CF}}$ does not need to access the original data. Second, $\boldsymbol{\alpha}^{\text{CF}}$ obtained through *Consensus Focus* is based on the quality of consensus, which utilize both data and label information and can identify malicious domains.

3.4. BatchNorm MMD: Decentralized Optimization Strategy of $\mathcal{H}-$divergence

To get a better UMDA performance, we need to minimize the $\mathcal{H}$-divergence between source domains and target domain, where the kernel-based MMD distance is widely used. Existing works (Long et al., 2015; Peng et al., 2019) use the feature $\boldsymbol{\pi}$ extracted by the fully-connected (fc) layers to build kernel as $\kappa(\mathbf{X}^S, \mathbf{X}^T) = \langle \boldsymbol{\pi}^S, \boldsymbol{\pi}^T \rangle$ and the related optimization target is

$$\min_{h \in \mathcal{H}} \sum_{k=1}^{K+1} \boldsymbol{\alpha}_k d_{\text{MMD}}^{\kappa}(\mathbb{D}_S^k, \mathbb{D}_T) \qquad (10)$$

However, these methods is not applicable in decentralized UMDA since the source domain data is unavailable. Besides, only using the high-level features from fc-layers may lose the detailed 2-D information. Therefore, we propose the *BatchNorm MMD*, which utilizes the mean and variance parameters in each BatchNorm layer to optimize the $\mathcal{H}-$divergence without accessing data.

BatchNorm (BN) (Ioffe & Szegedy, 2015) is a widely-used normalization technique. For the feature $\boldsymbol{\pi}$, BatchNorm is expressed as $\text{BN}(\boldsymbol{\pi}) = \gamma \cdot \frac{\boldsymbol{\pi} - \mathbb{E}(\boldsymbol{\pi})}{\sqrt{\text{Var}(\boldsymbol{\pi})}} + \beta$, where $(\mathbb{E}(\boldsymbol{\pi}), \text{Var}(\boldsymbol{\pi}))$ are estimated in training process[1]. Supposing the model contains L BatchNorm layers, we consider the quadratic kernel for the feature $\boldsymbol{\pi}_l$ of the l-th BN-layer, i.e. $\kappa(\mathbf{X}^S, \mathbf{X}^T) = \left(\langle \boldsymbol{\pi}_l^S, \boldsymbol{\pi}_l^T \rangle + \frac{1}{2} \right)^2$. The MMD distance based on this kernel is

$$\begin{aligned} d_{\text{MMD}}^{\kappa}(\mathbb{D}_S^k, \mathbb{D}_T) = & \|\mathbb{E}(\boldsymbol{\pi}_l^k) - \mathbb{E}(\boldsymbol{\pi}_l^T)\|_2^2 \\ & + \|\mathbb{E}[\boldsymbol{\pi}_l^k]^2 - \mathbb{E}[\boldsymbol{\pi}_l^T]^2\|_2^2 \end{aligned} \qquad (11)$$

Compared with other works using the quadratic kernel (Peng et al., 2019), we can obtain all required parameters in (11)

---

[1] Implemented with *running-mean* and *running-var* in Pytorch.

Algorithm 1 KD3A training process with epoch t.

Input:
    Source domains $\mathcal{S} = \{\mathbb{D}_S^k\}_{k=1}^K$. Target domain $\mathbb{D}_T$;
    Target model $h_T^{(t-1)}$ with parameters $\boldsymbol{\Theta}^{(t-1)}$;
    Confidence gate $g^{(t)}$;

Output:
    Target model $h_T^{(t)}$ with parameters $\boldsymbol{\Theta}^{(t)}$.

1: // Locally training on source domains:
2: **for** $\mathbb{D}_S^k$ in $\mathcal{S}$ **do**
3:     Model initialize: $(h_S^k, \boldsymbol{\Theta}_S^k) \leftarrow (h^{(t-1)}, \boldsymbol{\Theta}^{(t-1)})$.
4:     Train h_S^k with classification loss on $\mathbb{D}_S^k$.
5: **end for**
6: Upload $\{(h_S^k, \boldsymbol{\Theta}_S^k)\}_{k=1}^K$ to the target domain.
7: // *Knowledge Vote:*
8: $\mathbb{D}_S^{K+1} \leftarrow \text{KnowledgeVote}(\mathbb{D}_T, g^{(t)}, \{h_S^k\}_{k=1}^K)$.
9: Train h_S^{K+1} with L^{kv} loss (5) on $\mathbb{D}_S^{K+1}$.
10: // *Consensus Focus:*
11: $\boldsymbol{\alpha}^{\text{CF}} \leftarrow \text{ConsensusFocus}(\mathbb{D}_T, \{h_S^k\}_{k=1}^K, \{N_k\}_{k=1}^K)$.
12: // *Model Aggregation:*
13: $\boldsymbol{\Theta}^{(t)} \leftarrow \sum_{k=1}^{K+1} \boldsymbol{\alpha}_k^{\text{CF}} \cdot \boldsymbol{\Theta}_S^k$.
14: // *BatchNorm MMD:*
15: Obtain $\{\mathbb{E}[\boldsymbol{\pi}_l^k]^i, i = 1, 2\}_{l,k=1}^{L,K+1}$ from $\{(h_S^k, \boldsymbol{\Theta}_S^k)\}_{k=1}^{K+1}$
16: Train $h_T^{(t)}$ with BatchNorm MMD on $\mathbb{D}_T$.
17: Return $(h_T^{(t)}, \boldsymbol{\Theta}^{(t)})$.

---

through the parameters $(\mathbb{E}(\boldsymbol{\pi}_l), \text{Var}(\boldsymbol{\pi}_l))$ of BN-layers in source domain models without accessing data[2]. Based on this advantage, BatchNorm MMD can perform the decentralized optimization strategy of $\mathcal{H}-$divergence with two steps. First, we obtain $\{(\mathbb{E}(\boldsymbol{\pi}_l^k), \text{Var}(\boldsymbol{\pi}_l^k))\}_{l=1}^l$ from the models on different source domains. Then, for every mini-batch $\mathbf{X}^T \in \mathbb{D}_T$, we train the model h_T to optimize the domain adaptation target (10) with the following loss

$$\sum_{l=1}^{L} \sum_{k=1}^{K+1} \boldsymbol{\alpha}_k \left(\|\mu(\boldsymbol{\pi}_l^T) - \mathbb{E}(\boldsymbol{\pi}_l^k)\|_2^2 + \|\mu[\boldsymbol{\pi}_l^T]^2 - \mathbb{E}[\boldsymbol{\pi}_l^k]^2\|_2^2 \right) \tag{12}$$

where $(\boldsymbol{\pi}_1^T, \ldots, \boldsymbol{\pi}_L^T)$ are the features of target model h_T from BatchNorm layers corresponding to the input $\mathbf{X}^T$. In training process, We use the mean value μ of every mini-batch to estimate the expectation $\mathbb{E}$. In addition, optimizing the loss (12) requires traversing all Batchnorm layers, which is time-consuming. Therefore, we propose a computation-efficient optimization strategy in Appendix E.

3.5. The Algorithm of KD3A

In the above sections, we have proposed three essential components that work well in KD3A, and the complete algorithm of KD3A can be obtained by using these components

---

[2] Notice $\mathbb{E}[\boldsymbol{\pi}]^2 = \text{Var}(\boldsymbol{\pi}) + [\mathbb{E}(\boldsymbol{\pi})]^2$

in tandem: First, we obtain an extra source domain $\mathbb{D}_S^{K+1}$ and train the source model h_S^{K+1} through *Knowledge Vote*. Then, we get the target model by aggregating $K+1$ source models through *Consensus Focus*, i.e. $h_T := \sum_{k=1}^{K+1} \boldsymbol{\alpha}_k h_S^k$. Finally, we minimize the $\mathcal{H}-$divergence of the target model through *Batchnorm MMD*. The decentralized training process of KD3A is shown in Algorithm 1. Confidence gate is the only hyper-parameter in KD3A, and should be treated carefully. If the confidence gate is too large, almost all data in target domain would be eliminated and the knowledge vote loss would not work. If too small, then the consensus quality would be reduced. Therefore, we gradually increase it from low (e.g., 0.8) to high (e.g., 0.95) in training.

4. Generalization Bound For KD3A

We further derive the generalization bound for KD3A by combining the original bound (2) and the knowledge distillation bound (4). The related generalization bound is:

Theorem 2 (*The decentralized generalization bound for KD3A*). *Let h_T be the target model of KD3A, $\{\mathbb{D}_S^k\}_{k=1}^{K+1}$ be the extended source domains through Knowledge Vote and $\boldsymbol{\alpha}^{CF} \in \mathcal{R}_+^{K+1}, \sum_{k=1}^{K+1} \boldsymbol{\alpha}_k^{CF} = 1$ be the domain weights through Consensus Focus. Then we have:*

$$\epsilon_{\mathbb{D}_T}(h_T) \leq \sum_{k=1}^{K+1} \boldsymbol{\alpha}_k^{\text{CF}} \left(\epsilon_{\mathbb{D}_S^k}(h_T) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{D}_S^k, \mathbb{D}_T) \right) + \lambda_2 \tag{13}$$

The generalization performance of KD3A bound (13) depends on the quality of the consensus knowledge, as the following proposition shows (see Appendix C for proof):

Proposition 2 *The KD3A bound (13) is a tighter bound than the original bound (2), if the task risk gap between the knowledge distillation domain $\mathbb{D}_S^{K+1}$ and the target domain $\mathbb{D}_T$ is smaller than the following upper-bound for all source domain $k \in \{1, \cdots, K\}$, that is, $\epsilon_{\mathbb{D}_S^{K+1}}(h)$ should satisfy:*

$$\sup_{h \in \mathcal{H}} |\epsilon_{\mathbb{D}_S^{K+1}}(h) - \epsilon_{\mathbb{D}_T}(h)| \leq \inf_{h \in \mathcal{H}} |\epsilon_{\mathbb{D}_S^{K+1}}(h) - \epsilon_{\mathbb{D}_S^k}(h)|$$
$$+ \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{D}_S^k, \mathbb{D}_T) + \lambda_S^k$$

Proposition 2 points out two tighter bound conditions: (1) For those good source domains with small $\mathcal{H}-$divergence and low optimal task risk λ_S^k, the model should take their advantages to provide better consensus knowledge, i.e. the task risk $\epsilon_{\mathbb{D}_S^{K+1}}$ gets close enough to $\epsilon_{\mathbb{D}_T}$. (2) For those irrelevant and malicious source domains with high $\mathcal{H}-$divergence and λ, the model should filter out their knowledge, i.e. the task risk $\epsilon_{\mathbb{D}_S^{K+1}}$ stays away from that for bad domains.

The KD3A has heuristically achieved the above two conditions through the *Knowledge Vote* and *Consensus Focus*: (1)
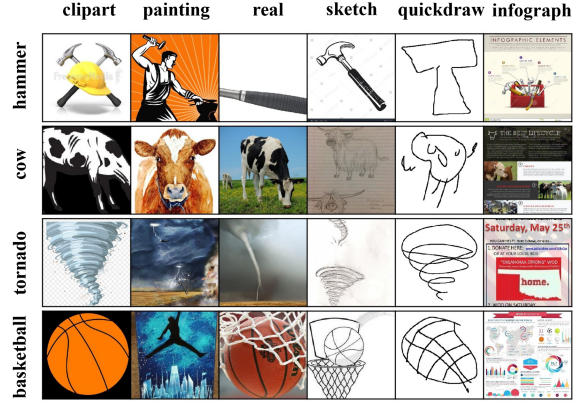

Figure 2. The large-scale dataset DomainNet. *Real* is a domain of high quality containing real-world images, while *Quickdraw* is an irrelevant source domain and may cause the negative transfer.

For good source domains, KD3A provides better consensus knowledge with *Knowledge Vote*, making $\epsilon_{\mathbb{D}_S^{K+1}}$ closer to $\epsilon_{\mathbb{D}_T}$. (2) For bad domains, KD3A filters out their knowledge with *Consensus Focus*, making $\epsilon_{\mathbb{D}_S^{K+1}}$ stay away from that for bad domains. We also conduct sufficient experiments to show our KD3A achieves tighter bound with better performance than other UMDA approaches.

5. Experiments

5.1. Domain Adaptation Performance

We perform experiments on four benchmark datasets: (1) **Amazon Review** (Ben-David et al., 2006), which is a sentimental analysis dataset including four domains. (2) **Digit-5** (Zhao et al., 2020), which is a digit classification dataset including five domains. (3) **Office-Caltech10** (Gong et al., 2012), which contains ten object categories from four domains. (4) **DomainNet** (Peng et al., 2019), which is a recently introduced benchmark for large-scale multi-source domain adaptation with 345 classes and six domains, i.e. *Clipart (clp)*, *Infograph (inf)*, *Painting (pnt)*, *Quickdraw (qdr)*, *Real (rel)* and *Sketch (skt)*, as shown in Figure 2. We follow the protocol used in prevailing works, selecting each domain in turn as the target domain and using the rest domains as source domains. Due to space limitations, we mainly present results on DomainNet; more results on Amazon Review, Digit-5 and Office-Caltech10 are provided in Appendix.

Baselines. We conduct extensive comparison experiments with the current best UMDA approaches from 4 categories: (1) $\mathcal{H}$-divergence based methods, i.e. the multi-domain adversarial network (MDAN) (Zhao et al., 2018) and moment matching (M³SDA) (Peng et al., 2019). (2) Knowledge ensemble based methods, i.e. the domain adaptive ensemble

Standards	Methods	Clipart	Infograph	Painting	Quickdraw	Real	Sketch	Avg
W/o DA	Oracle	$69.3_{\pm 0.37}$	$34.5_{\pm 0.42}$	$66.3_{\pm 0.67}$	$66.8_{\pm 0.51}$	$80.1_{\pm 0.59}$	$60.7_{\pm 0.48}$	63.0
	Source-only	$52.1_{\pm 0.51}$	$23.1_{\pm 0.28}$	$47.7_{\pm 0.96}$	$13.3_{\pm 0.72}$	$60.7_{\pm 0.32}$	$46.5_{\pm 0.56}$	40.6
$\mathcal{H}-$div.	MDAN	$60.3_{\pm 0.41}$	$25.0_{\pm 0.43}$	$50.3_{\pm 0.36}$	$8.2_{\pm 1.92}$	$61.5_{\pm 0.46}$	$51.3_{\pm 0.58}$	42.8
	M^{3}SDA	$58.6_{\pm 0.53}$	$\mathbf{26.0_{\pm 0.89}}$	$52.3_{\pm 0.55}$	$6.3_{\pm 0.58}$	$62.7_{\pm 0.51}$	$49.5_{\pm 0.76}$	42.6
Knowledge Ensemble	DAEL	$70.8_{\pm 0.14}$	$26.5_{\pm 0.13}$	$57.4_{\pm 0.28}$	$12.2_{\pm 0.7}$	$65.0_{\pm 0.23}$	$60.6_{\pm 0.25}$	48.7
Source Selection	CMSS	$64.2_{\pm 0.18}$	$28.0_{\pm 0.2}$	$53.6_{\pm 0.39}$	$16.0_{\pm 0.12}$	$63.4_{\pm 0.21}$	$53.8_{\pm 0.35}$	46.5
Others	DSBN*	60.3	22.6	52.3	9.1	62.7	47.6	42.4
Decentralized UMDA	SHOT*	61.7	22.2	52.6	12.2	67.7	48.6	44.2
	FADA*	59.1	21.7	47.9	8.8	60.8	50.4	41.5
	FADA	$45.3_{\pm 0.7}$	$16.3_{\pm 0.8}$	$38.9_{\pm 0.7}$	$7.9_{\pm 0.4}$	$46.7_{\pm 0.4}$	$26.8_{\pm 0.4}$	30.3
	KD3A	$\mathbf{72.5_{\pm 0.62}}$	$23.4_{\pm 0.43}$	$\mathbf{60.9_{\pm 0.71}}$	$16.4_{\pm 0.28}$	$\mathbf{72.7_{\pm 0.55}}$	$\mathbf{60.6_{\pm 0.32}}$	$\mathbf{51.1}$

Table 1. UMDA accuracy (%) on the DomainNet dataset. Our model KD3A achieves 51.1% accuracy, significantly outperforming all other baselines. Moreover, KD3A achieves the oracle performance on two domains: clipart and sketch. *: The best results recorded in our re-implementation.
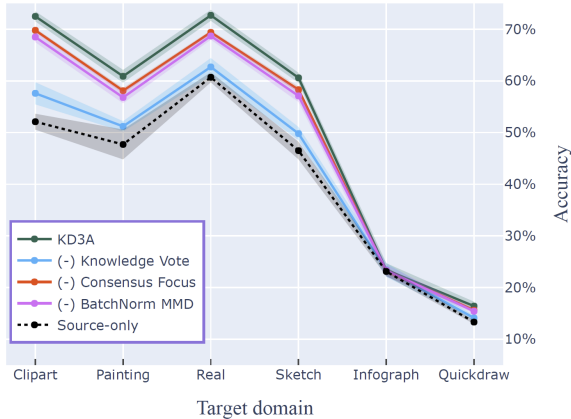

Figure 3. The ablation study of KD3A. Results show that *Knowledge Vote*, *Consensus Focus* and *BatchNorm MMD* all contribute to the UMDA performance in all target domains.

learning (DAEL) (Zhou et al., 2020). (3) Source selection based methods, i.e. the curriculum manager (CMSS) (Yang et al., 2020). (4) Decentralized UMDA, i.e. SHOT (Liang et al., 2020) and FADA (Peng et al., 2020). The DSBN proposes a domain-specific BatchNorm, which is similar to Batchnorm MMD, so we also take it into comparison. In addition, We report two baselines without domain adaptation, i.e. oracle and source-only. Oracle directly performs supervised learning on target domains and source-only naively combines source domains to train a single model.

Implementation details. Following the settings in previous

UMDA works (Peng et al., 2019; Yang et al., 2020), we use a 3-layer MLP as backbone for Amazon Review, a 3-layer CNN for Digit-5 and the ResNet101 pre-trained on ImageNet for Office-Caltech10 and DomainNet. The settings of communication rounds r is important in decentralized training. Since the models on different source domains have different convergence rates, we need to aggregate models r times per epoch. To perform the r-round aggregation, we uniformly divide one epoch into r stages and aggregate model after each stage. The KD3A Algorithm 1 is a decentralized training strategy with $r = 1$ and we use this setting in all experiments. For model optimization, We use the SGD with 0.9 momentum as the optimizer and take the cosine schedule to decay learning rate from high (i.e. 0.05 for Amazon Review and Digit5, and 0.005 for Office-Caltech10 and DomainNet) to zero. We conduct each experiment five times and report the results with the form mean$_{\pm \text{std}}$. Since SHOT and DSBN do not report the results on DomainNet, we re-implement them with the official code and report the best testing results.

DomainNet. The results on DomainNet are presented in Table 1. In general, our KD3A outperforms all the baselines by a large margin and achieves the oracle performance on *Clipart* and *Sketch*. In addition, compared with the oracle result (66.8%) and the source-only baseline (13.3%), all UMDA methods have failed in *Quickdraw*. Since the *Knowledge Vote* can provide good pseudo-labels for a few good samples and assign low weights to bad samples, KD3A slightly outperforms the source-only baseline. Table 1 also shows the UMDA performance can benefit from the knowledge

	$\mathcal{H}$-divergence	Info gain	Consensus focus	Domain drop
IR-qdr	57.9	57.7	58.1	**58.3**
MA-15	50.5	50.5	**52.1**	
MA-30	49.8	48.9	**51.1**	50.7
MA-50	47.6	46.3	50.6	

Table 2. Average UMDA accuracy (%) with irrelevant and malicious domains. IR-qdr means to use the *Quickdraw* as the irrelevant source domain, while MA-m means to construct a malicious source domain with $m\%$ mislabeled data. With consensus focus, our KD3A is robust to negative transfer.
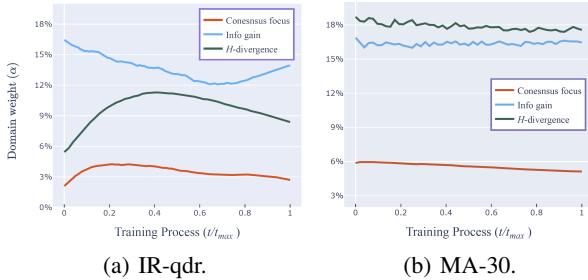

(a) IR-qdr. (b) MA-30.

Figure 4. Weights assigned to the irrelevant and malicious domains in the training process. Our consensus focus can identify these bad domains with the low weights.

ensemble (DAEL) and source domain selection (CMSS). Compared with DAEL, the KD3A provides better consensus knowledge on the high-quality domains such as *Clipart* and *Real*, while it also identifies the bad domains such as *Quickdraw*. CMSS select domains by checking the quality of each data with an independent network. Compared with CMSS, the KD3A does not introduce additional modules and can perform source selection in privacy-preserving scenarios. Moreover, our KD3A outperforms other decentralized models (e.g., SHOT and FADA) through the advantages in knowledge ensemble and source selection.

Ablation study. To evaluate the contributions of each component, We perform ablation study for KD3A , as shown in Figure 3. *Knowledge Vote*, *Consensus Focus* and *Batchnorm MMD* are all able to improve the accuracy, while most contributions are from *Knowledge Vote*, which indicates our KD3A can also perform well on those tasks that cannot use Batchnorm MMD.

5.2. Robustness To Negative Transfer

We construct irrelevant and malicious source domains on DomainNet and conduct synthesized experiments to show that with *Consensus Focus*, our KD3A is robust to negative transfer.

r	0.2	0.5	1	2	10	100
FADA	39.2	40.3	40.5	40.5	40.8	41.5
KD3A	**50.5**	**50.9**	**51.1**	**51.3**	**51.3**	**52.0**

Table 3. Average UMDA accuracy (%) with different communication rounds r for our KD3A and FADA. KD3A achieves good performance with low communication cost (e.g., $r \leq 1$).
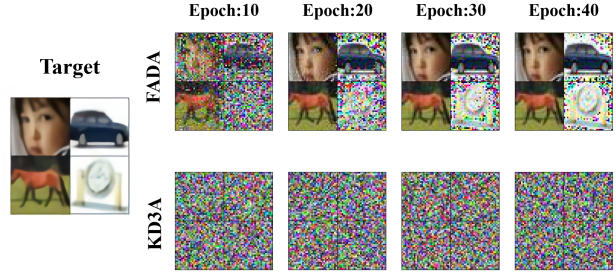

Figure 5. The gradient leakage attack (Zhu et al., 2019) on decentralized training strategy. KD3A is robust to this attack while FADA causes the privacy leakage.

Since *Quickdraw* is very different from other domains, and all models perform bad on it, we take *Quickdraw* as the irrelevant domain, denoted by **IR-qdr**. To construct malicious domains, we perform poisoning attack (Bagdasaryan et al., 2020) on the high-quality domain *Real* with $m\%$ wrong labels, denoted by **MA-m**. For the irrelevant domain IR-qdr, we select the remaining five domains in turn as target domains and train KD3A with the rest source domains. In training process, we plot the curve of the mean weight α assigned to IR-qdr by *Consensus Focus*. We also report the average UMDA accuracy across all target domains. For the malicious domain MA-m, we conduct the same process on the remained four domains except for *Quickdraw*. We report the same experiment results as IR-qdr.

We consider two advanced weighting strategies for comparison: the $\mathcal{H}$-divergence re-weighting in equation (6) and the *Info Gain* in FADA (Peng et al., 2020). In addition, we also report the average UMDA accuracy of KD3A model with the bad domain dropped. According to the results provided in Table 2 and Figure 4, we can get the following insights: (1) For IR-qdr and MA-(30,50), the negative transfer occurs since the domain-drop outperforms the others. (2) The three weighting strategies are robust to the irrelevant domain since they all assign low weights to IR-qdr. (3) *Consensus Focus* outperforms other strategies in malicious domains since it assigns extremely low weights to the bad domain (i.e. 5% for MA-30), while other strategies can not identify the malicious domain. Moreover, our KD3A can use the correct information of less malicious domains (i.e. MA-(15,30)) and achieves better performance than the domain-drop.

5.3. Communication Efficiency And Privacy Security

To evaluate the communication efficiency, We train the KD3A with different communication rounds r and report the average UMDA accuracy on DomainNet. We take the FADA method as a comparison. The results in Table 3 show the following properties: (1) Due to the adversarial training strategy, FADA works under large communication rounds (i.e. $r = 100$). (2) Our KD3A works under the low communication cost with $r = 1$, leading to a $100 \times$ communication reduction. (3) KD3A is robust to communication rounds. For example, the accuracy only drops 0.9% when r decreases from 100 to 1. Moreover, we consider two extreme cases where we synchronize models every 2 and 5 epochs, i.e. $r = 0.5$ and 0.2. In these cases, FADA performs worse than the source-only baseline while our KD3A can still achieve state-of-the-art results.

In decentralized training process, the frequent communication will cause privacy leakage (Wang et al., 2019), making the training process insecure. To verify the privacy protection capabilities, we perform the advanced gradient leakage attack (Zhu et al., 2019) on KD3A and FADA. As shown in Figure 5, the source images used in FADA are recovered under the attack, which causes privacy leakage. However, due to the low communication cost, our KD3A is robust to this attack, which demonstrates high privacy security.

6. Conclusions

We propose an effective approach KD3A to address the problems in decentralized UMDA. The main idea of KD3A is to perform domain adaptation through the knowledge distillation without accessing the source domain data. Extensive experiments on the large-scale DomainNet demonstrate that our KD3A outperforms other *state-of-the-art* UMDA approaches and is robust to negative transfer. Moreover, KD3A has a great advantage in communication efficiency and is robust to the privacy leakage attack.

Acknowledgments

References

Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., and Shmatikov, V. How to backdoor federated learning. In Chiappa, S. and Calandra, R. (eds.), *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020*, volume 108 of *Proceedings of Machine Learning Research*, pp. 2938–2948. PMLR, 2020.

Ben-David, S., Blitzer, J., Crammer, K., and Pereira, F. Analysis of representations for domain adaptation. In Schölkopf, B., Platt, J. C., and Hofmann, T. (eds.), *Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, 2006*, pp. 137–144. MIT Press, 2006.

Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., et al. A theory of learning from different domains. *Machine Learning*, 79(1-2):151–175, 2010.

Chang, W., You, T., Seo, S., Kwak, S., and Han, B. Domain-specific batch normalization for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*, pp. 7354–7362. Computer Vision Foundation / IEEE, 2019.

Chen, D., Mei, J., Wang, C., Feng, Y., and Chen, C. Online knowledge distillation with diverse peers. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*, pp. 3430–3437. AAAI Press, 2020.

Chen, J., Zheng, X., Yu, H., Chen, D. Z., and Wu, J. Electrocardio panorama: Synthesizing new ecg views with self-supervision. *arXiv preprint arXiv:2105.06293*, 2021.

Gong, B., Shi, Y., Sha, F., and Grauman, K. Geodesic flow kernel for unsupervised domain adaptation. In *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, pp. 2066–2073. IEEE Computer Society, 2012.

Hinton, G. E., Vinyals, O., Dean, J., et al. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Bach, F. R. and Blei, D. M. (eds.), *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 448–456. JMLR.org, 2015.

Konecný, J., McMahan, H. B., Yu, F. X., Richtárik, P., et al. Federated learning: Strategies for improving communication efficiency. *CoRR*, abs/1610.05492, 2016.

Li, R., Jiao, Q., Cao, W., Wong, H., and Wu, S. Model adaptation: Unsupervised domain adaptation without source data. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020*, pp. 9638–9647. IEEE, 2020. URL https://doi.org/10.1109/CVPR42600.2020.00966.

Liang, J., Hu, D., and Feng, J. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *Proceedings of the 37th*

International Conference on Machine Learning, ICML 2020, volume 119 of *Proceedings of Machine Learning Research*, pp. 6028–6039. PMLR, 2020.

Long, M., Cao, Y., Wang, J., and Jordan, M. I. Learning transferable features with deep adaptation networks. In Bach, F. R. and Blei, D. M. (eds.), *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 97–105. JMLR.org, 2015.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In Singh, A. and Zhu, X. J. (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 2017.

Meng, Z., Li, J., Gong, Y., and Juang, B. Adversarial teacher-student learning for unsupervised domain adaptation. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018*, pp. 5949–5953. IEEE, 2018.

Pan, S. J. and Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22 (10):1345–1359, 2010.

Peng, X., Bai, Q., Xia, X., et al. Moment matching for multi-source domain adaptation. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019*, pp. 1406–1415. IEEE, 2019.

Peng, X., Huang, Z., Zhu, Y., and Saenko, K. Federated adversarial domain adaptation. In *8th International Conference on Learning Representations, ICLR 2020*. OpenReview.net, 2020.

Saito, K., Watanabe, K., Ushiku, Y., and Harada, T. Maximum classifier discrepancy for unsupervised domain adaptation. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018*, pp. 3723–3732. IEEE Computer Society, 2018.

Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., and Darrell, T. Deep domain confusion: Maximizing for domain invariance. *CoRR*, abs/1412.3474, 2014.

Wang, Z., Song, M., Zhang, Z., Song, Y., Wang, Q., and Qi, H. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *2019 IEEE Conference on Computer Communications, INFOCOM 2019, Paris, France, April 29 - May 2, 2019*, pp. 2512–2520. IEEE, 2019.

Yan, H., Ding, Y., Li, P., et al. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pp. 945–954. IEEE Computer Society, 2017.

Yang, L., Balaji, Y., Lim, S., and Shrivastava, A. Curriculum manager for source selection in multi-source domain adaptation. In Vedaldi, A., Bischof, H., Brox, T., and Frahm, J. (eds.), *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XIV*, volume 12359 of *Lecture Notes in Computer Science*, pp. 608–624. Springer, 2020.

Zhang, K., Gong, M., Schölkopf, B., et al. Multi-source domain adaptation: A causal view. In Bonet, B. and Koenig, S. (eds.), *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 3150–3157. AAAI Press, 2015.

Zhang, S., Yao, D., Zhao, Z., Chua, T.-S., and Wu, F. Causerec: Counterfactual user sequence synthesis for sequential recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.

Zhao, H., Zhang, S., Wu, G., Costeira, J. P., Moura, J. M. F., and Gordon, G. J. Multiple source domain adaptation with adversarial learning. In *6th International Conference on Learning Representations, ICLR 2018*. OpenReview.net, 2018.

Zhao, S., Wang, G., Zhang, S., Gu, Y., Li, Y., Song, Z., Xu, P., Hu, R., Chai, H., and Keutzer, K. Multi-source distilling domain adaptation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*, pp. 12975–12983. AAAI Press, 2020.

Zhou, K., Yang, Y., Qiao, Y., and Xiang, T. Domain adaptive ensemble learning. *CoRR*, abs/2003.07325, 2020.

Zhu, L., Liu, Z., Han, S., et al. Deep leakage from gradients. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pp. 14747–14756, 2019.