**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 Knowledge Distillation based Decentralized Domain Adaptation (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× 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. Unsupervised Multi-source Domain Adaptation (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 Source-Target 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, Knowledge Distillation based Decentralized Domain Adaptation (KD3A), which aims to perform decentralized domain adaptation through the knowledge distillation on models from different source domains. Our KD3A approach...
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) establishes 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 $\kappa$, and optimize the $\mathcal{H}$-divergence by minimizing the MMD distance $d_{\text{MMD}}(\mathbb{D}_S, \mathbb{D}_T)$ on $\mathcal{H}_\kappa$. Using the kernel trick, MMD can be computed as

$$d_{\text{MMD}}(\mathbb{D}_S, \mathbb{D}_T) = -2\mathbb{E}_{(X_S,X_T) \sim \mathbb{D}_S \times \mathbb{D}_T} \kappa(X_S, X_T) + \mathbb{E}_{X_S \sim \mathbb{D}_S} \kappa(X_S, X'_S) + \mathbb{E}_{X_T \sim \mathbb{D}_T} \kappa(X_T, X'_T)$$

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., 2018a) 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...
Theorem 1 Let $\mathcal{H}$ be the model space, $\{\epsilon_{\mathcal{D}_s}^k(h)\}_{k=1}^K$ and $\epsilon_{\mathcal{D}_t}(h)$ be the task risks of source domains $\{\mathcal{D}_s^k\}_{k=1}^K$ and the target domain $\mathcal{D}_t$, and $\alpha \in \mathcal{R}_+^K$, $\sum_{k=1}^K \alpha_k = 1$ be the domain weights. Then for all $h \in \mathcal{H}$ we have:

$$\epsilon_{\mathcal{D}_t}(h) \leq \sum_{k=1}^K \alpha_k \left( \epsilon_{\mathcal{D}_s}^k(h) + \frac{1}{2} d_{\mathcal{H}}(\mathcal{D}_s^k, \mathcal{D}_t) \right) + \lambda_0$$

(2)

where $\lambda_0$ 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 $\mathcal{D}_t := \{X_t^i\}_{i=1}^{N_t}$. 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 $q_s^k(X)$ to denote the confidence for each class and use the class with the maximum confidence as label, i.e., $h_s^k(X) = \arg\max_c q_s^k(X)_c$. As shown in Figure 1(a), the knowledge distillation in UMDA consists of two steps. First, for each target domain data $X_t^i$, 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., $p_i = \frac{1}{K} \sum_{k=1}^K q_s^k(X_t^i)$. In order to utilize the consensus knowledge for domain adaptation, we define an extended source domain $\mathcal{D}_s^{K+1}$ with the consensus knowledge $p_i$ for each target domain data $X_t^i$ as

$$\mathcal{D}_s^{K+1} = \{(X_t^i, p_i)\}_{i=1}^{N_t}$$
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:

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

(4)

where $\lambda_1$ 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.

$$
\sup_{h \in \mathcal{H}} |\varepsilon_{D_T}(h) - \varepsilon_{D_S^{K+1}}(h)| \leq \lambda_1.
$$

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 $X^T \in \mathbb{D}_T$, we firstly use a high-level confidence gate to filter the predictions $\{q^T_S(X^T_i)\}_{k=1}^{K}$ 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 $p$, by conducting the mean ensemble on these supporting models. We also record the number of domains that support $p$, denoted by $n_p$. For those $X^T$ with all teacher models eliminated by the confidence gate, we simply use the mean ensemble to get $p$ and assign a relatively low weight to them as $n_p = 0.001$.

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

$$
I^K(X^T_i, q) = n_p \cdot D_{KL}(p_i || q(X^T_i))
$$

(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 $\alpha$ determine the contribution of each source domain. Ben-David et al. (2010) proves the optimal $\alpha$ 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

$$
\alpha_k = N_k e^{-d_{H}(\mathbb{D}_k, \mathbb{D}_T)} / \sum_k N_k e^{-d_{H}(\mathbb{D}_k, \mathbb{D}_T)}.
$$

(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 $X^T \in \mathbb{D}_T$ with the consensus knowledge $(p_i(\mathcal{S}'), n_p(\mathcal{S}'))$ obtained from $\mathcal{S}'$, We define the related consensus quality as $n_p(\mathcal{S}') \cdot \max p_i(\mathcal{S}')$ and the total consensus quality $Q$ is

$$
Q(\mathcal{S}') = \sum_{X^T \in \mathbb{D}_T} n_p(\mathcal{S}') \cdot \max p_i(\mathcal{S}')
$$

(7)

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

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

(8)

$\text{CF}(\mathbb{D}_S^k)$ 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 \( D_{S}^{k+1} \) in Knowledge Vote, we compute the domain weights with two steps. First, we obtain \( \alpha_{K+1} \) with parameters \( \Theta^{(t-1)} \) and \( \Theta^{(t)} \) from the source domain models without accessing data. Second, \( \alpha_{K+1} \) 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 \( \pi \) extracted by the fully-connected (fc) layers to build kernel as \( \kappa(X^S, X^T) = \langle \pi^S, \pi^T \rangle \) and the related optimization target is

\[
\min_{h \in \mathcal{H}} \sum_{k=1}^{K+1} \alpha_k d_{\text{MMD}}^c(D^k_S, D_T) \tag{10}
\]

However, these methods are 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 \( \pi \), BatchNorm is expressed as \( \text{BN}(\pi) = \gamma \cdot \frac{\pi - \mathbb{E}(\pi)}{\sqrt{\text{Var}(\pi)}} + \beta \), where \( (\mathbb{E}(\pi), \text{Var}(\pi)) \) are estimated in training process. Supposing the model contains \( L \) BatchNorm layers, we consider the quadratic kernel for the feature \( \pi_t \) of the \( l \)-th BN-layer, i.e. \( \kappa(X^S, X^T) = \langle (\pi_t^S)^2, (\pi_t^T)^2 + \frac{1}{2} \rangle^L \). The MMD distance based on this kernel is

\[
d_{\text{MMD}}^c(D^k_S, D_T) = \| \mathbb{E}(\pi_t^S) - \mathbb{E}(\pi_t^T) \|_2^2 + \| \mathbb{E}(\pi_t^S)^2 - \mathbb{E}(\pi_t^T)^2 \|_2^2 \tag{11}
\]

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

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

\[
\sum_{l=1}^{L} \sum_{k=1}^{K+1} \alpha_k (\| \mu(\pi_t^S) - \mathbb{E}(\pi_t^S) \|_2^2 + \| \mu(\pi_t^T)^2 - \mathbb{E}(\pi_t^T)^2 \|_2^2)
\]

(12)

where \( \pi_t^S, \pi_t^T \) are the features of target model \( h_T \) from BatchNorm layers corresponding to the input \( X^T \). In training process, We use the mean value \( \mu \) 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.
in tandem: First, we obtain an extra source domain $\mathbb{D}^{K+1}_S$ and train the source model $h^{K+1}_S$ through Knowledge Vote. Then, we get the target model by aggregating $K+1$ source models through Consensus Focus, i.e. $h_T := \sum_{k=1}^{K+1} \alpha_k h^k_S$. 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, $\{\mathbb{D}^{k+1}_S\}_{k=1}^{K}$ be the extended source domains through Knowledge Vote and $\alpha^T \in \mathbb{R}_{+}^{K+1}$, $\sum_{k=1}^{K+1} \alpha^T_k = 1$ be the domain weights through Consensus Focus. Then we have:

$$
\varepsilon_T(h_T) \leq \sum_{k=1}^{K+1} \alpha^T_k \left( \varepsilon_{\mathbb{D}^{s}_S}(h_T) + \frac{1}{2} d_{\mathcal{H}}^2(\mathbb{D}^{k}_S, \mathbb{D}_T) \right) + \lambda^T
$$

(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 $\mathbb{D}^{k+1}_S$ 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, $\varepsilon^{k+1}_S(h)$ should satisfy:

$$
\sup_{h \in \mathcal{H}} |\varepsilon^{k+1}_S(h) - \varepsilon_T(h)| \leq \inf_{h \in \mathcal{H}} |\varepsilon^{k+1}_S(h) - \varepsilon_{\mathbb{D}^{s}_S}(h)| + \frac{1}{2} d_{\mathcal{H}}^2(\mathbb{D}^{k}_S, \mathbb{D}_T) + \lambda^T
$$

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

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

For good source domains, KD3A provides better consensus knowledge with Knowledge Vote, making $\varepsilon^{k+1}_S(h)$ closer to $\varepsilon_T$. (2) For bad domains, KD3A filters out their knowledge with Consensus Focus, making $\varepsilon^{k+1}_S(h)$ 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 sentiment 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., 2018a) and moment matching (M$^3$SDA) (Peng et al., 2019). (2) Knowledge ensemble based methods, i.e. the domain adaptive ensemble...
KD3A: Unsupervised Multi-Source Decentralized Domain Adaptation via Knowledge Distillation

| Standards        | Methods | Clipart | Infograph | Painting | Quickdraw | Real | Sketch | Avg   |
|------------------|---------|---------|-----------|----------|-----------|------|--------|-------|
| W/o DA           | Oracle  | 69.3±0.37 | 34.5±0.42 | 66.8±0.67 | 66.8±0.51 | 80.1±0.59 | 60.7±0.48 | 63.0  |
|                  | Source-only | 52.1±0.51 | 23.1±0.28 | 47.7±0.96 | 13.3±0.72 | 60.7±0.32 | 46.5±0.56 | 40.6  |
| ∣H−div.           | MDAN    | 60.3±0.41 | 25.0±0.43 | 50.3±0.36 | 8.2±1.92  | 61.5±0.46 | 51.3±0.58 | 42.8  |
|                  | M'SDA   | 58.6±0.53 | 26.0±0.89 | 52.3±0.55 | 6.3±0.58  | 62.7±0.51 | 49.5±0.76 | 42.6  |
| Knowledge Ensemble | DAEL   | 70.8±0.14 | 26.5±0.13 | 57.4±0.28 | 12.2±0.7  | 65.0±0.23 | 60.6±0.25 | 48.7  |
| Source Selection  | CMSS    | 64.2±0.18 | 28.0±0.2  | 53.6±0.39 | 16.0±0.12 | 63.4±0.21 | 53.8±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±0.7  | 16.3±0.8  | 38.9±0.7  | 7.9±0.4   | 46.7±0.4  | 26.8±0.4  | 30.3  |
|                  | KD3A    | 72.5±0.62 | 23.4±0.43 | 60.9±0.71 | 16.4±0.28 | 72.7±0.55 | 60.6±0.32 | 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.

Knowledge. Since the knowledge distillation (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±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
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.

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 $\alpha$ 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.

![Figure 4](image1.png)

**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.

![Figure 5](image2.png)

**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.

| $r$  | 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., $r \leq 1$).
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

This research has been supported by National Key Research and Development Program (2019YFB1404802) and National Natural Science Foundation of China (U1866602, 61772456).

7. Appendix

7.1. Appendix A

Claim For the extended source domain \( D_{S}^{K+1} = \{(X^{T}_{i}, p_{i})\}_{i=1}^{N_T} \), training the related source model \( h_{S}^{K+1} \) with the knowledge distillation loss \( L^{kd}(X^{T}_{i}, q_{S}^{K+1}) = D_{KL}(p_{i}||q_{S}^{K+1}(X^{T}_{i})) \) equals to optimizing the task risk \( \epsilon_{D_{S}^{K+1}}(h) = \Pr_{(X,p) \sim D_{S}^{K+1}}[h(X) \neq \arg_{c} \max p_{c}] \).

Proof:

First, we prove that \( \forall c = 1, \ldots, C \),

\[
|q_{S}^{K+1}(X^{T}_{i})_{c} - p_{i,c}| \leq \sqrt{\frac{1}{2} D_{KL}(p_{i}||q_{S}^{K+1}(X^{T}_{i}))} \tag{1}
\]

The widely used Pinsker’s inequality states that, if \( P \) and \( Q \) are two probability distributions on a measurable space \((X, \Sigma)\), then

\[
\delta(P, Q) \leq \sqrt{\frac{1}{2} D_{KL}(P||Q)}
\]

where

\[
\delta(P, Q) = \sup \{|P(A) - Q(A)||A \in \Sigma, \Sigma \text{ is a measurable event.} \}
\]

In our situation, we choose the event \( A \) as the probability of classifying the input \( X^{T}_{i} \) into class \( c \), and the related probability under \( P, Q \) is \( p_{i,c} \) and \( q_{S}^{K+1}(X^{T}_{i})_{c} \). With Pinsker’s inequality, it is easy to prove (1). Since the inequality (1) holds for all class \( c \), minimizing the knowledge distillation loss will make \( q_{S}^{K+1}(X^{T}_{i}) \rightarrow p_{i} \), that is, \( \epsilon_{D_{S}^{K+1}}(h) \rightarrow 0 \).

7.2. Appendix B

Proposition 1 (The generalization bound for knowledge distillation). Let \( \cal{H} \) be the model space and \( \epsilon_{D_{S}^{K+1}}(h) \) be the task risk of the new source domain \( D_{S}^{K+1} \) based on knowledge distillation. Then for all \( h_{T} \in \cal{H} \), we have:

\[
\epsilon_{D_{T}}(h_{T}) \leq \epsilon_{D_{S}^{K+1}}(h_{T}) + \frac{1}{2} d_{\cal{H} \Delta H}(D_{S}^{K+1}, D_{T}) + \min_{h_{T} \in \cal{H}} \sup \{\epsilon_{D_{S}^{K+1}}(h) - \epsilon_{D_{T}}(h)\} \tag{2}
\]

where \( \lambda_{1} \) is a constant for the task risk of the optimal model.

Proof:

Following the Theorem 2 in Ben-David et al. (2010), for the source domain \( D_{S}^{K+1} \) and the target domain \( D_{T} \), for all \( h_{T} \in \cal{H} \), we have

\[
\epsilon_{D_{T}}(h_{T}) \leq \epsilon_{D_{S}^{K+1}}(h_{T}) + \frac{1}{2} d_{\cal{H} \Delta H}(D_{S}^{K+1}, D_{T}) + \lambda_{1} \tag{3}
\]

where \( \lambda_{1} \) is constant of the optimal model on the source domain and the target domain as \( \lambda_{1} = \min_{h_{T} \in \cal{H}} \epsilon_{D_{S}^{K+1}}(h) + \epsilon_{D_{T}}(h) \).
In addition, the following inequality also holds for all $h_T \in \mathcal{H}$:

$$
\epsilon_{D_T}(h_T) - \epsilon_{D_S^{K+1}}(h_T) \leq \sup_{h \in \mathcal{H}} |\epsilon_{D_T}(h) - \epsilon_{D_S^{K+1}}(h)|
$$

where $\sup_{h \in \mathcal{H}} |\epsilon_{D_T}(h) - \epsilon_{D_S^{K+1}}(h)|$ is the upper bound of the task risk gap between the target domain $D_T$ and the extended domain $D_{S}^{K+1}$. Notice $D_{S}^{K+1}$ shares the same input space with $D_T$ since they all use $(X_i^T)_{i=1}^{N_T}$ as inputs. Therefore, we have

$$
d_{H,\Delta_{H}}(D_{S}^{K+1}, D_T) = 0
$$

Substituting (5) into (4), we have

$$
\epsilon_{D_T}(h_T) \leq \epsilon_{D_S^{K+1}}(h_T) + \frac{1}{2}d_{H,\Delta_{H}}(D_{S}^{K+1}, D_T) + \sup_{h \in \mathcal{H}} |\epsilon_{D_T}(h) - \epsilon_{D_S^{K+1}}(h)|
$$

Combining (3) and (6), we get the Proposition 1.

The learning bound with empirical risk error. Proposition 1 shows how to relate the extended source domain $D_{S}^{K+1}$ and the target domain $D_T$. Since we use the finite samples to empirically estimate the $\hat{\epsilon}_{D_S^{K+1}}(h)$ and $\hat{d}_{H}(D_{S}^{K+1}, D_T)$ at the training time, we now proceed to give a learning bound for empirical risk minimization using $N_T$ sampled training data.

Following the learning bound Lemma 1.5 in Ben-David et al. (2010), for all $0 < \delta < 1$, with probability at least $1 - \delta$, we have:

$$
\epsilon_{D_S^{K+1}}(h) \leq \epsilon_{D_S^{K+1}}(h) + \frac{1}{N_T}(d \log \frac{2eN_T}{d} + \log \frac{4}{\delta})
$$

and

$$
d_{H,\Delta_{H}}(D_{S}^{K+1}, D_T) \leq \hat{d}_{H,\Delta_{H}}(D_{S}^{K+1}, D_T) + \frac{d \log(2N_T + \log(\frac{6}{\delta}))}{N_T} + \frac{d \log(2N_T + \log(\frac{6}{\delta}))}{N_T}
$$

where $d$ is the VC-dimension of model space $\mathcal{H}$.

Combining (2) and (7), we get the generalization bound for knowledge distillation with the empirical learning error as follows:

$$
\epsilon_{D_T}(h_T) \leq \epsilon_{D_S^{K+1}}(h) + \frac{1}{2}d_{H,\Delta_{H}}(D_{S}^{K+1}, D_T) + C_1
$$

where $C_1$ is a constant as

$$
C_1 = \min\{\lambda_1 + \frac{2}{N_T}(d \log \frac{2eN_T}{d} + \log \frac{4}{\delta}) + \frac{d \log(2N_T + \log(\frac{6}{\delta}))}{N_T}, \sup_{h \in \mathcal{H}} |\epsilon_{D_T}(h) - \epsilon_{D_S^{K+1}}(h)| + \frac{4}{N_T}(d \log \frac{2eN_T}{d} + \log \frac{4}{\delta})\}
$$

7.3. Appendix C

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

$$
\sup_{h \in \mathcal{H}} |\epsilon_{D_S^{K+1}}(h) - \epsilon_{D_T}(h)| \leq \inf_{h \in \mathcal{H}} |\epsilon_{D_S^{K+1}}(h) - \epsilon_{D_T}(h)|
$$

$$
+ \frac{1}{2}d_{H,\Delta_{H}}(D_{S}^{k}, D_T) + \lambda_S^k
$$

**Proof:**

Following the Theorem 2 in Ben-David et al. (2010), for each source domain $D_{S}^{k}$ and for all $h_T \in \mathcal{H}$, we have

$$
\epsilon_{D_T}(h_T) \leq \epsilon_{D_S^{k}}(h_T) + \frac{1}{2}d_{H,\Delta_{H}}(D_{S}^{k}, D_T) + \lambda_S^k
$$

where $\lambda_S^k = \min_{h \in \mathcal{H}} \epsilon_{D_S^{k}}(h) + \epsilon_{D_T}(h)$ is the optimal task risk of $D_{S}^{k}$ and $D_T$.

The original bound states that for all $h_T \in \mathcal{H}$, we have

$$
\epsilon_{D_T}(h_T) \leq \sum_{k=1}^{K} \alpha_k \left( \epsilon_{D_S^{k}}(h_T) + \frac{1}{2}d_{H,\Delta_{H}}(D_{S}^{k}, D_T) \right) + \lambda_0
$$

where $\lambda_0 = \min_{h \in \mathcal{H}} \sum_{k=1}^{K} \alpha_k \epsilon_{D_S^{k}}(h_T) + \epsilon_{D_T}(h_T)$ and we have the following relations between $\lambda_0$ and $\lambda_S^k$:

$$
\lambda_0 = \min_{h \in \mathcal{H}} \sum_{k=1}^{K} \alpha_k \epsilon_{D_S^{k}}(h) + \epsilon_{D_T}(h)
$$

$$
\geq \sum_{k=1}^{K} \alpha_k \min_{h \in \mathcal{H}} \epsilon_{D_S^{k}}(h) + \epsilon_{D_T}(h)
$$

$$
= \sum_{k=1}^{K} \alpha_k \lambda_S^k
$$

With (11 – 13), the original bound (12) can be considered as the weighted combination of the source domains. In addition, the KD3A bound is also the combination of the original bound (12) and the knowledge distillation bound (2). Then we get that the KD3A bound is a tighter bound than the original bound if the knowledge distillation bound (2) is tighter than the single source bound (11) for each source domain $D_{S}^{k}$, that is, for all source domain $k \in \{1, \cdots, K\}$ and all $h_T \in \mathcal{H}$, the knowledge distillation bound should satisfy:

$$
\epsilon_{D_S^{k}}(h_T) + \frac{1}{2}d_{H,\Delta_{H}}(D_{S}^{k+1}, D_T) + \lambda_S^k
$$

$$
\leq \epsilon_{D_S^{k}}(h_T) + \frac{1}{2}d_{H,\Delta_{H}}(D_{S}^{k}, D_T) + \lambda_S^k
$$
Table 4. The 3-layers CNN backbone for Digit-5.

Table 5. Implementation details of our KD3A on four benchmark datasets: Amazon Review, Digit-5, Office-Caltech10 and DomainNet.

Since $d_{H,\mathcal{D}}(\mathcal{D}^{K+1}_S, \mathcal{D}_T) = 0$ and $\lambda_1$ is a constant, the task risk gap $\sup_{h \in \mathcal{H}} |\epsilon_{\mathcal{D}^{K+1}_S}(h) - \epsilon_{\mathcal{D}_T}(h)|$ should satisfy the following condition for all $h_T \in \mathcal{H}$, that is:

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

(15)

Since condition (15) holds for all $h_T \in \mathcal{H}$, we have the tighter bound condition as

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

(16)

7.4. Appendix D: Representation Invariant Bounds For KD3A.

One reviewer argues that the generalization bound in proposition 1 is not rigorous since the optimization process may change the value of $\lambda$. The optimal joint risk $\lambda$ between source and target domain is defined as $\lambda := \min_{h \in \mathcal{H}} \epsilon_S(h) + \epsilon_T(h)$. $\lambda$ is based on the hypothesis space $\mathcal{H}$ and is usually intractable to compute. Considering the fixed model backbones are used in in practice (where the hypothesis space $\mathcal{H}$ is implicitly determined), we follow previous works (i.e. Theorem 1 in Long et al. (2015) and Theorem 2 in Zhao et al. (2018b)) and consider $\lambda$ as a constant. However, we agree with the fact proposed in Zhao et al. (2019) (Section 4.1) that optimizing the $\mathcal{H}$–divergence can learn domain invariant representations, but can also change the representation space. This may change the value of $\lambda$. As such, we take the suggestions of the reviewer and replace the original bound with the new bound in Zhao et al. (2019), which utilizes the $\mathcal{H}$–divergence and the constant term $C$.

With this upper bound, we propose a new version for our Proposition 1, Theorem 2 and Proposition 2 as follows:

**Proposition 1.** Denoting $C_1 := \min\{E_{\mathcal{D}^{K+1}_S}[\|f^{K+1}_S - f_T]\], E_{\mathcal{D}_T}[\|f^{K+1}_S - f_T]\}$, we have

$$\epsilon_{\mathcal{D}_T}(h_T) \leq \epsilon_{\mathcal{D}^{K+1}_S}(h_T) + d_{\mathcal{H}}(\mathcal{D}^{K+1}_S, \mathcal{D}_T) + \min\{C_1, \sup_{h \in \mathcal{H}} |\epsilon_{\mathcal{D}^{K+1}_S}(h) - \epsilon_{\mathcal{D}_T}(h)|\}$$

**Theorem 2.** Denoting $C_2 := \sum_{k=1}^{K+1} \alpha^C_k \min\{E_{\mathcal{D}^{k}_S}[\|f^{k}_S - f_T]\], E_{\mathcal{D}_T}[\|f^{k}_S - f_T]\}$, we have

$$\epsilon_{\mathcal{D}_T}(h_T) \leq \sum_{k=1}^{K+1} \alpha^C_k \left(\epsilon_{\mathcal{D}^{k}_S}(h_T) + d_{\mathcal{H}}(\mathcal{D}^{k}_S, \mathcal{D}_T)\right) + C_2$$

**Proposition 2.** Denoting $C^k_S := \min\{E_{\mathcal{D}^{k}_S}[\|f^{k}_S - f_T]\],
Table 6. UMDA accuracy (%) on the Digit-5. *: The best results recorded in our re-implementation. †: Methods trained without data-augmentation. Our model KD3A achieves 92.0% accuracy and outperforms all other baselines.

Table 7. The ablation study for data-augmentation strategies on DomainNet. †: Methods trained without data-augmentation.

Table 8. The UMDA performance on Amazon Review dataset.

7.5. Appendix E: The Implementation of BatchNorm MMD

We have introduced the BatchNorm MMD with the following loss:

$$L = \sum_{l=1}^{K+1} \sum_{k=1}^{\pi} \alpha_k \left( \| \mu(\pi^T_l) - \mathbb{E}(\pi^T_k) \|_2^2 + \| \mu(\pi^T_k) - \mathbb{E}(\pi^T_k) \|_2^2 \right)$$

(17)

However, directly optimizing the loss (17) requires to traverse all Batchnorm layers, which is time-consuming. Inspired by the suggestions of reviewers, we propose a computation-efficient method containing two steps. First, we directly derive the global optimal solution of $\mu(\pi^T)$ for loss (17), that is, $\forall l, 1 \leq l \leq L$, the optimal model $b_{\text{op}}$ on target domain $D_T$ should satisfy

$$\mu_{\text{op}}(\pi^T_l) = \sum_{k=1}^{K+1} \alpha_k \mathbb{E}(\pi^T_k)$$

(18)

Then we calculate the optimal solution from (18) as $\{\mu_{\text{op}}(\pi^T_l), \mu_{\text{op}}(\pi^T_l)^2\}_{l=1}^{K+1}$. Directly substitute this solution into every Batchnorm layer of $h^T_T$ and use it as global model. Although this computation-efficient implementation may seem heuristic, we find it practically work and can achieve the same performance as the original maximization step.
Table 9. UMDA accuracy (%) on the Office-Caltech10. *: The best results recorded in our re-implementation. †: Methods trained without data-augmentation.

| Methods | A    | C    | D    | W    | Avg |
|---------|------|------|------|------|-----|
| Oracle  | 99.7 | 98.4 | 99.8 | 99.7 | 99.4 |
| Source-only | 86.1 | 87.8 | 98.3 | 99.0 | 92.8 |
| MDAN    | 98.9 | 98.6 | 91.8 | 95.4 | 96.1 |
| M³SDA   | 94.5 | 92.2 | 99.2 | 99.5 | 96.4 |
| CMSS    | 96.0 | 93.7 | 99.3 | 99.6 | 97.2 |
| DSBN*   | 93.2 | 91.6 | 98.9 | 99.3 | 95.8 |
| FADA    | 84.2±0.5 | 88.7±0.5 | 87.1±0.6 | 88.1±0.4 | 87.1 |
| SHOT    | 96.4 | 96.2 | 98.5 | 99.7 | 97.7 |
| KD3A†   | 96.0±0.07 | 95.2±0.08 | 97.9±0.11 | 99.6±0.03 | 97.2 |
| KD3A    | 97.4±0.08 | 96.4±0.11 | 98.4±0.08 | 99.7±0.02 | 97.9 |

7.6. Appendix F

7.6.1. Implementation Details.

We perform UMDA on those datasets with multiple domains. During experiments, we choose one domain as the target domain, and use the remained domains as source domains. Finally, we report the average UMDA results among all domains. The code, with which the most important results can be reproduced, is available at Github³. In this section, we discuss the implementation details. Following previous settings (Peng et al., 2019), 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 details of hyper-parameters are provided in Table 5 and the backbones and training epochs are set to same in all method comparison experiments. In training process, We use the SGD as optimizer and take the cosine schedule to decay learning rate from high (0.05 for Amazon Review and Digit-5, and 0.005 for Office-Caltech10 and DomainNet) to zero.

Data augmentations. Data augmentations are important in deep network training process. Since different datasets require different augmentation strategies (e.g. rotate, scale, and crop), which introduces extra hyper-parameters, we use mixup (Zhang et al., 2017) as a unified augmentation strategy and simply set the mix-parameter \( \alpha = 0.2 \) in all experiments. For fair comparison, we report the results on both conditions, i.e. with/without data-augmentations. The results are shown in Table 7, 6 and 9. The ablation study in data augmentations indicates that mixup strategy can unify different augmentation strategies on different domain adaptation datasets with only one hyper-parameter. Moreover, KD3A can achieve good results even without data-augmentation.

7.6.2. Results on Amazon Review, Digit-5 and Office-Caltech10.

In this part, we report the experiment results on Amazon Review, Digit-5 and Office-Caltech10. Amazon Review is a sentimental analysis dataset including four domains: Books, DVDs, Electronics and Kitchen Appliances. Digit-5 is a digit classification dataset including MNIST (mt), MNIST(mmm), SVHN (sv), Synthetic (syn), and USPS (up). Office-Caltech10 contains 10 object categories from four domains, i.e. Amazon (A), Caltech (C), DSLR (D), and Webcam (W). Note that results are directly cited from published papers if we follow the same setting. The results on Table 8, 6 and 9 show that our KD3A outperforms other UMDA methods and advanced decentralized UMDA methods. Moreover, our KD3A provides better consensus knowledge on the hard domains such as the MNISTM domain on the Digit-5, which outperforms other methods by a large margin.

³github.com/FengHZ/KD3A
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 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.

Konečný, J., McMahan, H. B., Yu, F. X., Richtárik, P., et al. Federated learning: Strategies for improving communication efficiency. CoRR, abs/1610.05492, 2016.
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, H., Cissé, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *CoRR*, abs/1710.09412, 2017. URL http://arxiv.org/abs/1710.09412.

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, 2018a.

Zhao, H., Zhang, S., Wu, G., Moura, J. M. F., Costeira, J. P., and Gordon, G. J. Adversarial multiple source domain adaptation. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018*, pp. 8568–8579, 2018b.

Zhao, H., des Combes, R. T., Zhang, K., and Gordon, G. J. On learning invariant representations for domain adaptation. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7523–7532. PMLR, 2019.