Multi-Source Domain Adaptation with Sinkhorn Barycenter

Tatsuya Komatsu
LINE Corporation
Tokyo, Japan

Tomoko Matsui
The Institute of Statistical Mathematics
Tokyo, Japan

Junbin Gao
The University of Sydney
Sydney, Australia

***Abstract*—We describe a multi-source and unsupervised domain adaptation method using Sinkhorn barycenters, which, given the labeled data in multi-source domains and unlabeled data in a target domain, uses the optimal transport Sinkhorn distance to measure gaps between data distributions in the source and target domains. For end-to-end classification learning, the feature extractor and classifier are simultaneously estimated on the basis of two criteria: the minimization of the Sinkhorn distance for the source and target domains and the minimization of the classification loss for the source domains. The first criterion is based on the assumptions that domain-invariant features would be captured in a latent feature space obtained by minimizing the Sinkhorn distance among all domains and that the space would be close to the Sinkhorn barycenter. Experiments on image classification using the Digit-Five dataset, which is comprised of digit datasets from five different domains, demonstrated that our method outperforms other state-of-the-art methods.**

***Index Terms*—unsupervised domain adaptation, multi-domain adaptation, optimal transport, Sinkhorn distance**

I. Introduction

Various kinds of data, such as image and speech data, from various sources are being collected and accumulated via the Internet and various devices. These data have different characteristics depending on the data source, making it difficult to combine and use them. Technology is thus needed for effectively utilizing multi-source data.

There are various strategies for dealing with multi-source data. In the multi-view/multi-source paradigm, one strategy is to transform multi-view/multi-source data into a target domain (to be learned) so that subsequent learning tasks can be performed in the target domain. This strategy has been successful, for example, in the context of multi-view clustering, such as multi-view matrix factorization [1], [2], multi-view K-means clustering [3], multi-view kernel learning [4], and multi-view spectral clustering [5].

Strategies are being actively studied for unsupervised domain adaptation (DA), which utilizes the knowledge of the source domain(s) to perform tasks in an unlabeled target domain, such as classification of unlabeled data. Research has mostly addressed the use of a single-source domain. Several methods use loss functions based on generative adversarial networks to reduce domain confusion and to estimate the joint distribution [6] and domain-invariant features [7]. Another commonly used approach to unsupervised DA involves bringing the data distributions of the source and target domains closer together and leveraging the source domain's rich knowledge regarding label information to improve classification performance in the target domain.

Several types of distance measures between two domains have been investigated, e.g., second order correlation [8], [9], moment matching [10], maximum mean discrepancy (MMD) [6], [7], [11]–[14], Kullback-Leibler divergence [15], and $\mathcal{H}$-divergence [16].

The success of single-source DA approaches has prompted interest in exploring domain adaptation for multi-source scenarios. Multi-source DA (MDA) extends the single-source setting to a framework in which labeled data from multiple sources with different distributions can be aggregated. Various methods have been proposed for MDA, e.g., $\mathcal{H}\Delta\mathcal{H}$-divergence between a weighted combination of multi-source domains [16], a method using a joint adaptation network [11], and a method using a deep cocktail network [17].

The key idea of MDA for multiple domains with diverse characteristics is to use constraints based on the distances among the domains in order to relax the gaps among the domain distributions. The most important thing when using the constraints is measuring the distances among the distributions. Peng *et al.* [18] proposed using moment matching and demonstrated that such a method outperformed ones using single-source and multi-source approaches, such as Long *et al.* [11] and Xu *et al.* [17]. However, with the method of Peng *et al.* [18], only a limited number of moments among the distributions are matched, so further study is necessary for relaxing the gaps among the distributions.

Optimal transport (OT) is attracting much attention as a promising and flexible solution to the problem of comparing probabilistic densities when measuring distances among distributions. OT-based methods for comparing two probability densities and generative models are vital in machine-learning research, where data are often presented in the form of point clouds, histograms, bags-of-features, or, more generally, even manifold-valued datasets. The methods initially investigated were aimed at solving the problem of optimally allocating a transportation source's data distribution to a destination (so that the transportation cost was minimized). Unfortunately, the strength of OT comes at an enormous computational cost, the cost of solving the OT problem, which is impractical for large-scale applications. Approximation or good solutions of the OT problem have been actively studied. One solution

is the Kantorovich-Rubinstein dual formulation. Li *et al.* [19] proposed using an OT-based measure, the Wasserstein-1 distance, for MDA, which is facilitated by the use of the dual formulation and approximation with a neural network. Another approximation approach to gap relaxation between data distributions [20]–[22] is to use the generalized Sinkhorn distance based on OT with entropy-type regularization. This approach has a reasonable calculation cost and has been widely used in computer vision, natural language processing, gene analysis, and so on.

This study introduces the OT-induced Sinkhorn distance with a fast and scalable algorithm to MDA. Our contributions are

- introducing the use of the OT-induced Sinkhorn distance to measure the closeness of datasets while taking data distributions into account;
- presenting a loss function combining cross-entropy loss for classification purposes and the loss of the pair-wise OT distance among the source domains and target domain.

Experiments on using MDA for digit classification under the same experimental conditions as those used by Peng *et al.* [18] demonstrated that the proposed method performs better than a state-of-the-art DA method [18], [19].

II. Multi-source Domain Adaptation Model with Sinkhorn Barycenter

A. Problem Formulation

1) Overview: Let us consider a classification problem in target domain $\mathcal{T}$ without labeled data when labeled data in N source domains $\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_N\}$ are available. In the source domain with labeled data $\mathcal{S}_n$, the data can be mapped to a latent feature space $\hat{\mathcal{S}}_n$ for classification using the label information. However, for the target domain without labeled data, data need to be mapped to the latent space without the label information. The MDA methods seek a model $\mathcal{M}_0$ to map data in the target domain $\mathcal{T}$ to the latent feature space $\hat{\mathcal{T}}$ by effectively using knowledge in the source domain with labels:

$$\mathcal{M}_0 : \mathcal{T} | \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_N\} \to \hat{\mathcal{T}}. \tag{1}$$

For $N = 1$, i.e., data in only one source domain is available, one may consider mapping the target domain $\hat{\mathcal{T}}$ close to the source domain $\hat{\mathcal{S}}_1$ where the label information can be useful. For $N > 1$, i.e., multi-source setting, the key is how to handle the relationship between the data in each domain to estimate $\hat{\mathcal{T}}$. In many studies, the relationship among domains has been formulated and measured as the distance between distributions, such as moment matching [10], maximum mean discrepancy (MMD) [6], [7], [11]–[14]. In this paper, we measure the relationship among domains based on Sinkhorn distance [20]–[22] and estimate the latent feature space $\hat{\mathcal{T}}$ as a barycenter $\hat{\mathcal{B}}$ in a Wasserstein space [23], i.e., $\hat{\mathcal{T}}$ is estimeted minimizing Sinkhorn distance between $\hat{\mathcal{T}}$ and $\hat{\mathcal{S}}_n$,

$$\mathcal{B} = \text{argmin}_{\hat{\mathcal{T}}} \sum_n OT_\epsilon\{\mathcal{S}_n, \hat{\mathcal{T}}\} \tag{2}$$

where $OT_\epsilon()$ is the Sinkhorn distance based on the OT distance with entropic regularization.

2) Model Components: Model $\mathcal{M}_S$ is represented as a feature extractor f_{θ_f}, and parameter θ_f is estimated by considering classifier g_{θ_g} for each domain of $\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_N, \mathcal{T}\}$. When using deep neural networks, each pair component can be embedded in a network, and the structure can be shared among all domains.

$$f_{\theta_f} : \mathbf{X}^n \to \mathbf{Z}^n \tag{3}$$

$$g_{\theta_g} : \mathbf{Z}^n \to \mathbf{Y}^n, \tag{4}$$

where $\mathbf{X}^n$ is data in the n-th domain, $\mathbf{Z}^n$ the latent feature, $\mathbf{Y}^n$ the labels, and $\theta = (\theta_f, \theta_g)$ the parameter set of the feature extractor and classifier. For domain $\mathcal{T}$ (that is, the $(N+1)$-th domain), the values of $\mathbf{Y}^{N+1}$ are unknown.

Then, the parameter $\hat{\theta}$ is estimated so as to minimize loss function L:

$$\hat{\theta} = \underset{(\theta_f, \theta_g) \in \Theta}{\text{argmin}} \; L(\{f_{\theta_f}(\mathbf{X}^n)\}_{n=1}^{N+1}, \{g_{\theta_g}(\mathbf{Z}^n)\}_{n=1}^{N}, \{\mathbf{Y}^n\}_{n=1}^{N}). \tag{5}$$

3) Loss Function: An important factor in the above model assumption is defining an appropriate loss function. The loss should satisfy two major criteria; (1) it should measure the classification capability, and (2) it should measure the closeness of the source and target domains in the latent feature space.

We propose a loss function that meets both criteria. It uses the Sinkhorn distance (SD) as the loss between the domains (l_{SD}) and uses cross-entropy (CE) as the classification loss for the source domains (l_{CE}):

$$\begin{aligned} &L(\{f_{\theta_f}(\mathbf{X}^n)\}_{n=1}^{N+1}, \{g_{\theta_g}(\mathbf{Z}^n)\}_{n=1}^{N}, \{\mathbf{Y}^n\}_{n=1}^{N}) \\ &= l_{CE}(\{g_{\theta_g}(\mathbf{Z}^n)\}_{n=1}^{N}, \{\mathbf{Y}^n\}_{n=1}^{N}) + \lambda \cdot l_{SD}(\{f_{\theta_f}(\mathbf{X}^n)\}_{n=1}^{N+1}), \end{aligned} \tag{6}$$

where λ is a weight for regulating the scale of the difference between SD and CE. While l_{CE} is well-known, especially in the context of deep neural networks [24], we define l_{SD} for multi-source DA:

$$\begin{aligned} l_{SD}(\{f_{\theta_f}(\mathbf{X}^n)\}_{n=1}^{N+1}) = &\frac{1-\alpha}{N} \sum_{n=1}^{N} OT_\epsilon(f_{\theta_f}(\mathbf{X}^n), f_{\theta_f}(\mathbf{X}^{N+1})) \\ &+ \frac{\alpha}{\binom{N}{2}} \sum_{i \neq j}^{N} OT_\epsilon(f_{\theta_f}(\mathbf{X}^i), f_{\theta_f}(\mathbf{X}^j))). \end{aligned} \tag{7}$$

The first term corresponds to Eq. (2), which itself can be used to estimate the barycenter, but the proposed method also uses the second term, the OT distance between the source domains. The second term follows the loss function of MDA method with moment matching [18]. The parameter α is a weight parameter, where if $\alpha = 0$, Eq. (7) is consistent with Eq. (2). The effect of α on performance is shown in the experiment section. The purpose of minimizing loss l_{SD} is to bring together all the transformed source/target domain datasets, where the closeness between each pair is represented by the OT distance with entropic regularization such that, in

reality, the data distributions are matched. This differs greatly from general domain adaptation in which the data are transformed into the latent feature space without considering their precise distribution matching. In the algorithm implementation specified in the next section (Eq. (11)), the loss function is calculated in terms of batch data.

B. Training Procedure

The procedure for training the network parameters is shown in Algorithm 1. We use a single feature extractor f_{θ_f} and a classifier g_{θ_g} that are shared in all domains. Parameters θ_f and θ_g are estimated by stochastic gradient descent (SGD) to minimize Eq. (5).

First, mini-batch from each source domain are sampled: $\{\mathbf{x}_i^n, \mathbf{y}_i^n\}_{i=1}^B \sim (\mathbf{X}^n, \mathbf{Y}^n)$. Here, $\mathbf{x}_i^n$ represents image data in the n-th domain, and $\mathbf{y}_i^n$ represents the d-dimensional one-hot vector for the class of $\mathbf{x}_i^n$. In the experiments, the size of $\mathbf{x}_i^n$ was 28×28, and the dimension of $\mathbf{y}_i^n$ was 10. Each image data is fed into the feature extractor, and feature $\left\{\mathbf{z}_i^n = f_{\theta_f}\left(\mathbf{x}_i^n\right)\right\}_{i=1}^B$ is obtained. The extracted features $\{\mathbf{z}_i^n\}_{i=1}^B$ are classified by class $\left\{\tilde{\mathbf{y}}_i^n = g_{\theta_g}\left(\mathbf{z}_i^n\right)\right\}_{i=1}^B$. Here, $\tilde{\mathbf{y}}_i^n$ is a d-dimensional vector, and each element represents the probability of the corresponding class. The CE loss for each mini-batch is calculated:

$$l_{CE}(\{g_{\theta_g}^n(\mathbf{Z}^n)\}_{n=1}^N, \{\mathbf{Y}^n\}_{n=1}^N) = \textstyle\sum_n l_{CE}^n \tag{8}$$

$$l_{CE}^n = \tfrac{1}{B} \textstyle\sum_{i=1}^B \sum_{c=0}^9 y_{i,c}^n \log \tilde{y}_{i,c}^n. \tag{9}$$

Next, mini-batch data in the target domain $\left\{\mathbf{x}_i^{N+1}\right\}_{i=1}^B \sim \mathbf{X}^{N+1}$ is sampled and fed into the feature extractor to obtain target features $\left\{\mathbf{z}_i^{N+1}\right\}_{i=1}^B$. To calculate the OT distance between the n-th and m-th domains in Eq. (7), we solve the OT problem by using an entropic regularization term:

$$OT_\epsilon\left(\{\mathbf{z}_i^n\}_{i=1}^B, \{\mathbf{z}_j^m\}_{j=1}^B\right) = \min_P \langle P, M\rangle - \varepsilon H(P), \tag{10}$$

where M is a metric matrix with elements $m_{i,j} = \left|\mathbf{z}_i^n - \mathbf{z}_j^n\right|^2$, and P is a joint probability matrix with elements $p_{i,j} = p(\mathbf{z}_i^n, \mathbf{z}_j^n)$. The $\langle\cdot,\cdot\rangle$ represents the Frobenius norm, and P is the plan for transporting the batch data from the n-th domain into the m-th domain. The OT problem in Eq. (10) is practically solved using

$$\begin{aligned} & OT_\epsilon\left(\{\mathbf{z}_i^n\}_{i=1}^B, \{\mathbf{z}_j^m\}_{j=1}^B\right) = \langle \hat{P}, M\rangle \\ = \; & \textstyle\sum_{i=1}^B \sum_{j=1}^B \left|\mathbf{z}_i^n - \mathbf{z}_j^m\right|^2 \cdot \hat{p}(\mathbf{z}_i^n, \mathbf{z}_j^m), \end{aligned} \tag{11}$$

where $\hat{P}$ is the transport plan estimated using the Sinkhorn algorithm [25].

Finally, the proposed loss in Eq. (6) is obtained from the CE and the SD, and the network parameters are updated by minimizing the loss by using SGD.

III. Experiments

A. Experimental Setting

We used the Digit-Five dataset [18], which comprises five digit-recognition-benchmark datasets; MNIST [26], MNIST-M [27], SVHN[1], USPS[2] and SYN [28]. The Digit-Five dataset consists of training and testing sets. We split the validation datasets in half and used them for testing and validation.

The characteristics of each dataset differed, and for MNIST-M and SVHN in particular, visually distinguishing the numbers was difficult. We conducted experiments with one of the five datasets as the target domain in turn and the other four datasets as the source domains.

For the model with the loss in Eq. (6), we used the same network architecture that was used by Peng *et al.* [18]. The network is composed of three convolution layers and three fully connected layers. The first five layers are for feature extraction, and the last layer is for classification. To implement the Sinkhorn algorithm, we used the GeomLoss toolkit [29]. The batch size for each domain was set to 128.

Algorithm 1 Procedure for training network parameters

Input: $(\mathbf{X}^n, \mathbf{Y}^n)$ in the N source domains and $\mathbf{X}^{N+1}$ in the target domain
Parameters: network parameters θ_f and θ_g
Hyper-parameters: mini-batch size B, weight parameters λ (Eq. (6)) and α (Eq. (7))
while not converged **do**
  for each source domain $n \in 1, ..., N$ **do**
    Sample mini-batch $\{(\mathbf{x}^n, \mathbf{y}^n)\}_{i=1}^B \sim (\mathbf{X}^n, \mathbf{Y}^n)$
    Extract features $\left\{\mathbf{z}_i^n = f_{\theta_f}(\mathbf{x}_i^n)\right\}_{i=1}^B$
    Classify features $\left\{\tilde{\mathbf{y}}_i^n = g_{\theta_g}(\mathbf{z}_i^n)\right\}_{i=1}^B$
    Calculate cross-entropy (Eq. (8))
  end for
  Sample mini-batch in target domain $\{\mathbf{x}_i^{N+1}\}_{i=1}^B \sim \mathbf{X}^{N+1}$
  Extract target features $\left\{\mathbf{z}_i^{N+1} = f_{\theta_f}(\mathbf{x}_i^{N+1})\right\}_{i=1}^B$
  Calculate Sinkhorn distance (Eqs. (7) and (11))
  Calculate total loss (Eq. (6))
  Update network parameters (θ_f, θ_g) by using stochastic gradient descent (Eq. (5))
end while

Fig. 1. Five-Digit dataset

B. Main Results: Comparison with State-of-the-Art Methods

Table I lists the average classification accuracy of digit recognition in the target domain and its standard deviation

[1]http://ufldl.stanford.edu/housenumbers/
[2]https://www.openml.org/d/41070

TABLE I

CLASSIFICATION ACCURACY (%) OF PROPOSED METHOD AND THREE CONVENTIONAL METHODS (M3SDA [18], WASSERSTEIN-1 [19], AND MMD).

Target	Proposed $\varepsilon = 0.01$	Proposed $\varepsilon = 0.1$	Proposed $\varepsilon = 1.0$	M3SDA [18]	Wasserstein-1 [19]	MMD
MNIST	98.7 ± 0.03	**98.8 ± 0.05**	98.7 ± 0.03	98.7 ± 0.11	98.5 ± 0.09	97.8 ± 0.12
MNIST-M	**69.4 ± 0.27**	**69.4 ± 0.20**	69.3 ± 0.28	65.3 ± 0.37	68.9 ± 0.18	65.7 ± 0.07
SVHN	74.5 ± 0.04	**74.6 ± 0.19**	**74.6 ± 0.29**	73.3 ± 0.34	73.2 ± 0.21	71.9 ± 0.40
USPS	97.1 ± 0.05	**97.3 ± 0.05**	**97.3 ± 0.05**	97.0 ± 0.19	97.1 ± 0.11	95.5 ± 0.10
SYN	86.6 ± 0.57	**87.0 ± 0.52**	**87.0 ± 0.64**	84.7 ± 0.43	86.7 ± 0.31	82.0 ± 0.47

TABLE II

CLASSIFICATION ACCURACY (%) FOR VARIOUS VALUES OF λ AND α IN EQ. (6)

Target	($\alpha = 0$) $\lambda = 1/1000$	$\lambda = 1/2000$	$\lambda = 1/4000$	($\lambda = 1/2000$) $\alpha = 0.25$	$\alpha = 0.5$	$\alpha = 0.75$
MNIST	98.5 ± 0.10	**98.8 ± 0.05**	98.8 ± 0.07	98.5 ± 0.11	98.6 ± 0.07	98.5 ± 0.03
MNIST-M	66.7 ± 0.09	69.3 ± 0.20	**69.8 ± 0.61**	69.2 ± 0.36	68.8 ± 0.43	68.3 ± 0.25
SVHN	72.0 ± 0.51	74.6 ± 0.19	**76.0 ± 0.62**	**74.7 ± 0.26**	74.7 ± 0.60	74.5 ± 0.18
USPS	96.8 ± 0.05	**97.3 ± 0.05**	97.3 ± 0.23	97.1 ± 0.22	97.2 ± 0.27	97.1 ± 0.15
SYN	84.4 ± 0.31	**87.0 ± 0.52**	86.1 ± 0.14	85.5 ± 0.32	85.3 ± 0.68	85.0 ± 0.63

when a deep network was trained with five random initial values for our method and three conventional methods: M3SDA, MMD (for details refer to [18]), and wasserstein-1 [19]. The difference between our method and the other methods is whether l_{SD}, moment matching, the MMD with the Gaussian kernel, or the Wasserstein-1 distance is used in the second term of Eq. (6). For our method, α in Eq. (6) was set to 0; that is, l_{SD} was calculated only between the source and target domains. Weight λ was set to $1/2000$ on the basis of the results of a preliminary experiment. Parameter ε in the algorithm for SD was varied from 0.01 to 1.0. The best performance was obtained with $\varepsilon = 0.1$; ε scales M in the algorithm and should be set appropriately for the task. For all values of ε, the proposed method performed better than the other distance measures, demonstrating that our method with l_{SD} is stably effective.

C. Results for Various Values of Parameters λ and α

We also examined the difference in classification accuracy for various values of λ and α, where λ is a parameter for balancing the scale difference between the values of l_{CE} and l_{SD}. In particular, the value size of l_{SD} varied depending on the mini-batch size and the feature values in the calculation of M. Table II lists the classification accuracy for various values of λ in Eq. (6). When $\lambda = 1/2000$, the accuracy was highest for all domains, so we set λ to this value in all the experiments. It can be seen that our method with $\alpha = 0$ performed stably well for all domains. Some of the source domains seemed to be distant from each other, so the first and second terms on the right side of Eq. (7) might have conflicted. Further investigation is needed to set λ and α properly depending on the target and source domains' distributional properties.

D. Analysis of Classification and OT loss

Figure 2 shows the changes in OT loss $l_{SD}^{(1)}$ between the target and source domains (the 1st term on the right side of Eq. 7) and OT loss $l_{SD}^{(2)}$ between the source domains (the 2nd term on the right side of Eq. (7)) when setting SVHN as the target domain and learning the deep network up to 100 iterations. The α in Eq. 7 was set to 0. The OT losses of both $l_{SD}^{(1)}$ and $l_{SD}^{(2)}$ decreased as the number of iterations increased for all domains. Since our method uses a common classifier for all domains, the classier implicitly had the effect of shifting the data of all domains to the same space, and the inter-distributional distance among all sources was minimized. The reduction rate tended to be smaller when the number of repetitions exceeded 60. These results indicate that MDA with our method is stable and effective. The difference in scale between l_{CE} and l_{SD} was $O(10^3)$, so $\lambda(= 1/2000)$ in 6 can be varied to adjust the difference.

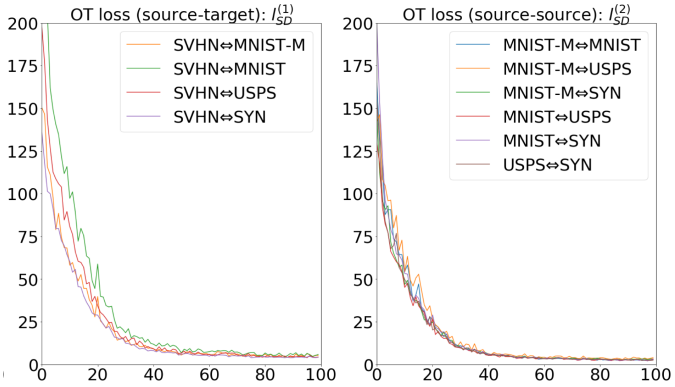

Fig. 2. OT loss $l_{SD}^{(1)}$ between target and source domains (1st term on right side of Eq. (7)), and OT loss $l_{SD}^{(2)}$ between source domains (2nd term on right side of Eq. (7)) when setting SVHN as target domain and with up to 100 iterations of learning deep network.

E. Complexity Analysis

The overall computational cost consists of the loss computations for l_{CE} and l_{SD}. The overhead of computing l_{CE} is

roughly equivalent to training a normal deep neural network classifier with all data coming from a single source domain. However, one bottleneck in the computation is from computing loss l_{SD} as all pair-wise distances among all source domains, which is $O(N^2)$, have to be calculated.

For each pair of domains, the cost of computing the SD in Eq. (11) includes the cost of computing M between the data from two domains, roughly $O(B^2)$, where the B is the mini-batch size, and the cost consumed by the Sinkhorn algorithm [30]. The complexity of the Sinkhorn algorithm giving an ϵ-approximate solution is $O(B\log(B)\epsilon^{-3})$ [31]. Overall, the final cost for l_{SD} is about $O(N^2B^2)$, where N is the number of domains.

Since the number of domains N is not large in most application problems, the overall complexity is comparable to that of other multi-domain training tasks. However, as our model sufficiently exploits data distribution matching, the extra cost is arguably worthwhile as it results in better classification, as demonstrated by the results of our experiments. Our method, with $\alpha = 0$ in Eq. (6), showed stable performance in all domains in the experiment. In other words, the terms involving pairs of source domains in l_{SD} had an almost negligible effect on model performance in our experiments, so we can reduce the complexity by excluding the term.

IV. Conclusion

We have presented a novel multi-source domain adaptation method using the Sinkhorn barycenter. The data distributions of multi-source domains and the target domain are matched by minimizing both the loss of the cross-entropy and the loss of the Sinkhorn distance between the distributions and then shifting the data of all domains to the same space. Digit classification experiments demonstrated that our method outperforms other state-of-the-art methods.

References

[1] H. Zhao, Z. Ding, and Y. Fu, "Multi-view clustering via deep matrix factorization," in *Proc of AAAI*, 2017.

[2] K. Zhan, J. H. Shi, J. Wang, and F. Tian, "Graph-regularized concept factorization for multi-view document clustering," *J VIS COMMUN IMAGE REPRESENT*, vol. 48, pp. 411–418, 2017.

[3] J. Xu, J. Han, F. Nie, and X. Li, "Re-weighted discriminantively embedded k-means for multi-view clustering," *IEEE Trans. Image Process.*, vol. 26, no. 6, pp. 3016–3027, 2017.

[4] G. Tzortzis and A. Likas, "Kernel-based weighted multi-view clustering," in *Proc of ICDM*, 2012.

[5] Z. Yang, Q. Xu, W. Zhang, X. Cao, and Q. Huang, "Split multiplicative multi-view subspace clustering," *IEEE Trans. Image Process.*, vol. 28, no. 10, pp. 5147–5160, 2019.

[6] Ming-Yu Liu and Oncel Tuzel, "Coupled generative adversarial networks," in *Proc of NIPS*, 2016, pp. 469–477.

[7] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell, "Adversarial discriminative domain adaptation.," in *Proc of CVPR*, 2017, vol. 1, pp. 4–12.

[8] Xingchao Peng and Kate Saenko, "Synthetic to real adaptation with generative correlation alignment networks," in *Proc of WACV*, 2018, pp. 1982–1991.

[9] Baochen Sun, Jiashi Feng, and Kate Saenko, "Return of frustratingly easy domain adaptation," in *Proc of AAAI*, 2016, pp. 1–8.

[10] Werner Zellinger, Edwin Lughofer, Susanne Saminger-Platz, Thomas Grubinger, and Thomas Natschlager, "Central moment discrepancy (CMD) for domain-invariant representation learning," in *Proc of ICLR*, 2017, pp. 1–13.

[11] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan, "Deep transfer learning with joint adaptation networks," in *Proc of ICML*, 2017, pp. 2208–2217.

[12] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell, "Deep domain confusion: Maximizing for domain invariance," *arXiv:1412.3474*, 2014.

[13] Muhammad Ghifary, W. Bastiaan Kleijn, and Mengjie Zhang, "Domain adaptive neural networks for object recognition," in *Proc of PRICAI*, 2014, pp. 898–904.

[14] Jindong Wang, Wenjie Feng, Yiqiang Chen, Han Yu, Meiyu Huang, and Philip S Yu., "Visual domain adaptation with manifold embedded distribution alignment," in *Proc of ACM Multimedia*, 2018.

[15] Fuzhen Zhuang, Xiaohu Cheng, Ping Luo, Sinno Jialin Pan, and Qing He, "Supervised representation learning: Transfer learning with deep autoencoders," in *Proc of IJCAI*, 2015, pp. 4119–4125.

[16] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan, "A theory of learning from different domains," *Machine Learning*, vol. 79, no. 1-2, pp. 151–175, 2010.

[17] Ruijia Xu, Ziliang Chen, Wangmeng Zuo, Junjie Yan, and Liang Lin, "Deep cocktail network: Multi-source unsupervised domain adaptation with category shift," in *Proc of CVPR*, 2018, pp. 3964–3973.

[18] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang, "Moment matching for multi-source domain adaptation," in *Proc of ICCV*, 2019, pp. 1406–1415.

[19] Yitong Li, David E Carlson, et al., "Extracting relationships by multi-domain matching," in *Proc. of NIPS*, 2018, pp. 6798–6809.

[20] Marco Cuturi and Arnaud Doucet, "Fast computation of Wasserstein barycenters," in *Proc. of ICML*, Eric P. Xing and Tony Jebara, Eds., 2014, pp. 685–693.

[21] Jean-David Benamou, Guillaume Carlier, Marco Cuturi, Luca Nenna, and Gabriel Peyré, "Iterative Bregman projections for regularized transportation problems," *SIAM J. Scientific Computing*, vol. 37, no. 2, pp. A1111–A1138, 2015.

[22] Giulia Luise, Saverio Salzo, Massimiliano Pontil, and Carlo Ciliberto, "Sinkhorn barycenters with free support via Frank-Wolfe algorithm," in *Proc of NIPS*, 2020.

[23] M. Agueh and G. Carlier, "Barycenters in the Wasserstein space," *SIAM J. Math. Analysis*, vol. 43, no. 2, pp. 904–924, 2011.

[24] Kevin P. Murphy, *Machine Learning: A Probabilistic Perspective*, The MIT Press, 2012.

[25] Marco Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *Proc. of NIPS*, 2013, pp. 2292–2300.

[26] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[27] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, 2011.

[28] Yaroslav Ganin and Victor Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proc of ICML*, 2015, pp. 1–10.

[29] Jean Feydy, Thibault Séjourné, François-Xavier Vialard, Shun-ichi Amari, Alain Trouve, and Gabriel Peyré, "Interpolating between optimal transport and mmd using sinkhorn divergences," in *Proc of AISTATS*, 2019, pp. 2681–2690.

[30] G. Peyre and M. Cuturi, *Computational Optimal Transport: With Applications to Data Science*, Foundations and Trends in Machine Learning Series. Now Publishers, 2019.

[31] Jason Altschuler, Jonathan Weed, and Philippe Rigollet, "Near-linear time approximation algorithms for optimal transport via Sinkhorn iteration," in *Proc of NIPS*, 2017, pp. 1961–1971.