MULTI-SOURCE DOMAIN ADAPTATION WITH VON NEUMANN CONDITIONAL DIVERGENCE

Ammar Shaker & Shujian Yu
NEC Laboratories Europe GmbH
{ammar.shaker,Shujian.Yu}@neclab.eu

ABSTRACT

The similarity of feature representations plays a pivotal role in the success of domain adaptation and generalization. Feature similarity includes both the invariance of marginal distributions and the closeness of conditional distributions given the desired response y (e.g., class labels). In this work, we introduce the recently proposed von Neumann conditional divergence to improve the transferability across multiple domains. We show that this new divergence is differentiable and eligible to easily quantify the functional dependence between features and y. We investigate the utility of this divergence in the problem of multi-source domain adaptation (for regression). A new learning objective and generalization bound are developed accordingly. We obtain favorable performance against state-of-the-art methods in terms of smaller generalization error on new tasks.

1 INTRODUCTION

The efficient transfer of information from one (or multiple) task to another is a fundamental technique for the successful deployment of a deep learning system (Yosinski *et al.*, 2014; Riemer *et al.*, 2018). Tremendous efforts have been made to improve transferability across multiple domains (Ganin *et al.*, 2016; Zhao *et al.*, 2018; 2019). Most of the work on transferability aim to learn domain-invariant features $\mathbf{t}$ without using y (e.g., class labels). Common techniques to match feature marginal distributions include the maximum mean discrepancy (MMD) (Pan *et al.*, 2010) and the moment matching (Zellinger *et al.*, 2017). However, it is still an open problem to explicitly capture the functional dependence between $\mathbf{t}$ and y for regression.

Consider a network consisting of a feature extractor $f_\theta : \mathcal{X} \to \mathcal{T}$ (parametrized by θ) and a predictor $h_\varphi : \mathcal{T} \to \mathcal{Y}$ (parameterized by φ), the similarity of the latent representation $\mathbf{t}$ includes two aspects: the invariance of marginal distributions (i.e., $p(f_\theta(\mathbf{x}))$) across different domains and the functional closeness of using $\mathbf{t}$ to predict y. The predictive power of h_φ can be characterized by the conditional distribution $p(y|\mathbf{t})$. From an information-theoretic perspective, the conditional entropy $H(y|\mathbf{t}) = -\mathbb{E}(\log(p(y|t)))$ also measures the dependence between y and $\mathbf{t}$ (Zhao *et al.*, 2020). On the other hand, the maximization of dependence amounts to the minimization of cross-entropy loss (Alemi *et al.*, 2016; Amjad and Geiger, 2019; Yu *et al.*, 2021).

We first introduce the recently proposed von Neumann conditional divergence (Yu *et al.*, 2020) (denote by D_{vN}) and present its properties when used as a loss function to train deep neural networks. We further propose a simple yet effective approach to model $p(y|\mathbf{t})$ by D_{vN} in multi-source domain adaptation (for regression). Our method performs favorably in encouraging positive forward transfer. Our main contributions are summarized as follows: *(i)* We introduce D_{vN} to the problem of domain adaptation and generalization. *(ii)* We show the utility of D_{vN} in a standard unsupervised domain adaptation setup in which multiple source tasks are observed simultaneously (*a.k.a.*, multi-source domain adaptation), and develop a novel learning objective and a generalization bound. The results confirm that the D_{vN}-based objective improves the unsupervised robustness for new tasks.

2 BACKGROUND KNOWLEDGE

Let $\mathcal{X}$ and $\mathcal{Y}$ be the input and the desired response (e.g., class labels) spaces. Given K source domains (or tasks) $\{D_i\}_{i=1}^K$, we obtain N_i training samples $\{\mathbf{x}_i^j, y_i^j\}_{j=1}^{N_i}$ in the i-th source D_i which follows a distribution $P_i(\mathbf{x}, y)$ (defined over $\mathcal{X} \times \mathcal{Y}$).

In a typical domain adaptation setup, the goal is to generalize a parametric model learned from data samples in $\{D_i\}_{i=1}^K$ to an unseen target domain D_{K+1} following a new distribution $P_{K+1}(\mathbf{x}, y)$: $\mathbb{E}_{(\mathbf{x},y)\sim D_{K+1}}[\ell(w; \mathbf{x}, y)]$. $\ell(w; \mathbf{x}, y) : \mathcal{W} \to \mathbb{R}$ is the loss function of w associated with $(\mathbf{x}, y)$, and $\mathcal{W} \subseteq \mathbb{R}^d$ is the model parameter space.

In this work, we consider multi-source domain adaptation for regression (i.e., $y \in \mathbb{R}$). Following common practice, we assume no access to the true response y in data sampled from $P_{K+1}(\mathbf{x}, y)$ in the multi-source domain adaptation.

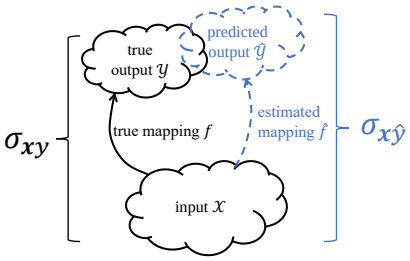


Figure 1: Geometry of loss $\mathcal{L} : \mathbb{S}^p_{++} \times \mathbb{S}^p_{++} \to \mathbb{R}_+$: $\sqrt{J_{vN}}$ searches an "optimal" predictor $\hat{f}$ minimizing the discrepancy between $\sigma_{\mathbf{x},f(\mathbf{x})}$ and $\sigma_{\mathbf{x},\hat{f}(\mathbf{x})}$. $\mathbb{S}^p_{++}$ is the set of $p \times p$ positive definite matrices.

2.1 von Neumann Conditional Divergence

Let us draw N samples from two joint distributions $P_1(\mathbf{x}, y)$ and $P_2(\mathbf{x}, y)$, i.e., $\{\mathbf{x}_1^i, y_1^i\}_{i=1}^N$ and $\{\mathbf{x}_2^i, y_2^i\}_{i=1}^N$. Here, y refers to the response variable, and $\mathbf{x}$ can be either the raw input variable or the feature vector $\mathbf{z} = f_\theta(\mathbf{x})$ after a feature extractor $f_\theta : \mathcal{X} \to \mathcal{Z}$ parameterized by θ.

Yu *et al.* (2020) define the relative divergence from $P_1(y|\mathbf{x})$ to $P_2(y|\mathbf{x})$ as:

$$D(P_1(y|\mathbf{x}) \| P_2(y|\mathbf{x})) = D_{vN}(\sigma_{\mathbf{x}y} \| \rho_{\mathbf{x}y}) - D_{vN}(\sigma_{\mathbf{x}} \| \rho_{\mathbf{x}}), \tag{1}$$

where $\sigma_{\mathbf{x}y}$ and $\rho_{\mathbf{x}y}$ denote the covariance matrices estimated on $\{\mathbf{x}_1^i, y_1^i\}_{i=1}^N$ and $\{\mathbf{x}_2^i, y_2^i\}_{i=1}^N$, respectively. Similarly, σ_x and ρ_x refer to the covariance matrices estimated on $\{\mathbf{x}_1^i\}_{i=1}^N$ and $\{\mathbf{x}_2^i\}_{i=1}^N$, respectively. D_{vN} is the von Neumann divergence (Nielsen and Chuang, 2002; Kulis *et al.*, 2009) $D_{vN}(\sigma\|\rho) = \mathrm{Tr}(\sigma \log \sigma - \sigma \log \rho - \sigma + \rho)$, which operates on two symmetric positive definite (SPD) matrices σ and ρ. Eq. (1) is not symmetric. To achieve symmetry, one can take the form: $D(P_1(y|\mathbf{x}) : P_2(y|\mathbf{x})) = \frac{1}{2}(D(P_1(y|\mathbf{x}) \| P_2(y|\mathbf{x})) + D(P_2(y|\mathbf{x}) \| P_1(y|\mathbf{x})))$.

3 Interpreting the von Neumann Conditional Divergence as a Loss Function

In case $P_1(\mathbf{x}, y)$ and $P_2(\mathbf{x}, y)$ have the same marginal distribution $P(\mathbf{x})$ or share the same input variable $\mathbf{x}$, we have $\sigma_{\mathbf{x}} = \rho_{\mathbf{x}}$. The symmetric von Neumann conditional divergence reduces to:

$$D(P_1(y|\mathbf{x}) : P_2(y|\mathbf{x})) = \frac{1}{2} \mathrm{Tr}\left((\sigma_{\mathbf{x}y} - \rho_{\mathbf{x}y})(\log \sigma_{\mathbf{x}y} - \log \rho_{\mathbf{x}y})\right). \tag{2}$$

We term the r.h.s. of Eq. (2) as the Jeffery von Neumann divergence on $\sigma_{\mathbf{x}y}$ and $\rho_{\mathbf{x}y}$, and denote it as $J_{vN}(\sigma_{\mathbf{x}y} : \rho_{\mathbf{x}y})$. The squared root $\sqrt{J_{vN}(\sigma_{\mathbf{x},f(\mathbf{x})} : \sigma_{\mathbf{x},\hat{f}(\mathbf{x})})}$ can be interpreted and used as a loss function to train a deep neural network. Here, $\mathbf{x}$ refers to the input variable, $f : \mathbf{x} \to y$ is the true labeling or mapping function, and $\hat{f}$ is the estimated predictor. $\sigma_{\mathbf{x},f(\mathbf{x})}$ and $\sigma_{\mathbf{x},\hat{f}(\mathbf{x})}$ denote the covariance matrix for the pairs of variables $\{\mathbf{x}, f(\mathbf{x})\}$ and $\{\mathbf{x}, \hat{f}(\mathbf{x})\}$, respectively. This is because $J_{vN}(\sigma_{\mathbf{x}y} : \rho_{\mathbf{x}y})$ measures the closeness between the true mapping (or labeling) function f and the estimated predictor $\hat{f}$, see Fig. 1 for an illustrative explanation.Appendix A presents three appealing properties associated with J_{vN}.

4 Multi-Source Domain Adaptation with Matrix-based Discrepancy Distance

4.1 Bounding the von Neumann Conditional Divergence in Target Domain

Motivated by the discrepancy distance D_{disc} defined in Cortes and Mohri (2014) based on a loss function $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$, we first present our matrix-based discrepancy distance $D_{\text{M-disc}}$ to quantify the discrepancy of two distributions P and Q over $\mathcal{X}$ based on our new loss $\mathcal{L} : \mathbb{S}^p_{++} \times \mathbb{S}^p_{++} \to \mathbb{R}_+$ (i.e., $\sqrt{J_{vN}(\sigma_{\mathbf{x},f(\mathbf{x})} : \sigma_{\mathbf{x},\hat{f}(\mathbf{x})})}$).

Definition 1. *The matrix-based discrepancy distance ($D_{\text{M-disc}}$) measures the longest distance between two domains (with respect to the hypothesis space $\mathcal{H}$) in a metric space equipped with the square root of Jeffery von Neumann divergence J_{vN} as a distance function. Given D_s and D_t and their distributions P_s and P_t, for any two hypotheses $h, h' \in \mathcal{H}$, $D_{\text{M-disc}}$ takes the form:*

$$D_{\text{M-disc}}(P_s, P_t) = \max_{h,h' \in \mathcal{H}} \left| \sqrt{J_{vN}(\sigma^s_{x,h(x)} : \sigma^s_{x,h'(x)})} - \sqrt{J_{vN}(\sigma^t_{x,h(x)} : \sigma^t_{x,h'(x)})} \right|, \tag{3}$$

with $a \in \{s, t\}$ and $g \in \{h, h'\}$, the matrix $\sigma^a_{\mathbf{x},g(\mathbf{x})}$ is the covariance matrix for the pair of variable $\mathbf{x}, g(\mathbf{x})$ in domain D_a.

Same to the $\mathcal{H}\Delta\mathcal{H}$ divergence in binary classification (Ben-David *et al.*, 2010), $D_{\text{M-disc}}$ reaches its maximum if a predictor h' is very close to h on the source domain but far on the target domain (or vice-versa). Fixing h, $D_{\text{M-disc}}(P_s, P_t; h)$ simply searches only for $h' \in \mathcal{H}$ maximizing Eq. (3).

Theorem 2, in Appendix B, shows that given a set of K source domains $S = \{D_{s_1}, \ldots, D_{s_K}\}$, for any hypothesis $h \in \mathcal{H}$, the square root of J_{vN} on the target domain D_t is bound as follows:

$$\sqrt{J_{vN}(\sigma^t_{x,h(x)} : \sigma^t_{x,f_t(x)})} \leq \sum_{i=1}^{K} w_i \left(\sqrt{J_{vN}(\sigma^s_{x,h(x)} : \sigma^s_{x,f_{s_i}(x)})} \right) + D_{\text{M-disc}}(P_t, P_\alpha; h) + \eta_Q(f_\alpha, f_t), \tag{4}$$

where f_{s_i} is the ground truth mapping function for D_{s_i} associated with the weight w_i, f_α is the convex combination of all functions f_{s_i}, $\eta_Q(f_\alpha, f_t)$ is the minimum joint empirical losses on the source D_α and the target D_t, achieved by an optimal hypothesis h^*. This can be interpreted as bounding the square root of J_{vN} on the target domain D_t by quantities controlled by *(i)* a convex combination over the square root of J_{vN} in each of the sources, i.e., $\sqrt{J_{vN}(\sigma^s_{x,h(x)} : \sigma^s_{x,f_{s_i}(x)})}$; *(ii)* the mismatch between the weighted distribution P_α and the target distribution P_t, i.e., $D_{\text{M-disc}}(P_t, P_\alpha; h)$; and *(iii)* the optimal joint empirical risk evaluated on source and target, i.e., $\eta_Q(f_\alpha, f_t)$ (which is always assumed to be small or nearly zero (Zhao *et al.*, 2019)).

4.2 Optimization by Adversarial Min-Max Game

Similar to DANN (Ganin *et al.*, 2016) that implicitly perform distribution matching by an adversarial min-max game, we explicitly implement the idea exhibited by Theorem 2 and combine a feature extractor $f_\theta : \mathcal{X} \to \mathcal{T}$ and a class of predictor $\mathcal{H} : \mathcal{T} \to \mathcal{Y}$ in a unified learning framework:

$$\min_{\substack{f_\theta, h \in \mathcal{H} \\ ||\alpha||_1 = 1}} \max_{h' \in \mathcal{H}} \left(\sum_{i=1}^{K} w_i \sqrt{J_{vN}(\sigma^{s_i}_{x,h(f_\theta(x))} : \sigma^{s_i}_{x,y})} + \lambda ||\mathbf{w}||_2 + \left| \sqrt{J_{vN}(\sigma^t_{f_\theta(\mathbf{x}),h(f_\theta(\mathbf{x}))} : \sigma^t_{f_\theta(\mathbf{x}),h'(f_\theta(\mathbf{x}))})} \right. \right.$$
$$\left. \left. - \sum_{i=1}^{K} w_k \sqrt{J_{vN}(\sigma^{s_i}_{f_\theta(\mathbf{x}),h(f_\theta(x))} : \sigma^{s_i}_{f_\theta(\mathbf{x}),h'(f_\theta(x))})} \right| \right). \tag{5}$$

The first term of Eq. (5) enforces h to be a good predictor to all source tasks[1]; the second term is a regularization on weight from each source (to the target); the third term is an explicit instantiation of our $D_{\text{M-dist}}(P_t, P_\alpha)$. The general idea is to find a feature extractor $f_\theta(\mathbf{x})$ that for any given pair of h and h', it is hard to discriminate the target domain P_t from the weighted combination of source distribution P_α. We term our method the multi-source domain adaptation with matrix-based discrepancy distance (MDD). We also notice that a similar min-max training strategy has been used in Pei *et al.* (2018); Saito *et al.* (2019); Richard *et al.* (2020).

4.3 Comparison with State-of-the-Art Methods

We evaluate our MDD on two real-world datasets *(i)* Amazon review dataset[2], and *(ii)* the YearPredictionMSD data (Bertin-Mahieux, 2011).

[1] In practice, one can replace the J_{vN} loss with the root mean square error (RMSE) loss with negligible performance difference.

[2] https://www.cs.jhu.edu/~mdredze/datasets/sentiment/

Table 1: Performance comparison in terms of mean absolute error (MAE) over 5 iterations on the Amazon data. **ba**:baby, **be**:beauty, **ca**:camera, **co**:computer&video, **al**:electronics, **go**:gourmet-food, **gr**:grocery.

	AHD-1S	DANN-1S	AHD-MSDA	MDAN-Max	MDAN-Dyn	MDD
ba	0.627(0.003)	2.921(1.307)	0.586(0.003)	0.591(0.015)	0.711(0.006)	**0.583**(0.007)
be	0.614(0.003)	1.133(0.213)	0.608(0.005)	0.628(0.003)	0.656(0.004)	**0.591**(0.005)
ca	0.559(0.003)	1.024(0.134)	0.534(0.006)	0.522(0.005)	0.598(0.006)	**0.508**(0.004)
co	0.617(0.005)	2.185(0.833)	0.61(0.004)	0.682(0.016)	0.829(0.055)	**0.605**(0.008)
el	0.669(0.002)	0.663(0.011)	0.657(0.002)	0.654(0.001)	0.670(0.003)	**0.651**(0.002)
go	0.585(0.002)	0.856(0.316)	0.566(0.003)	0.552(0.003)	0.553(0.003)	**0.549**(0.001)
gr	0.543(0.003)	1.458(0.786)	0.527(0.002)	0.519(0.002)	0.538(0.003)	**0.514**(0.007)

Table 2: Performance comparison in terms of mean absolute error (MAE) over five iterations on YearPredictionMSD data. DANN-1S fails on Dom1 and Dom5.

	AHD-1S	DANN-1S	AHD-MSDA	MDAN-Max	MDAN-Dyn	MDD
Dom1	7.10(0.07)	—	7.04(0.07)	18.1(9.2)	16.8(8.6)	**6.69**(0.18)
Dom2	8.42(0.07)	34.9(14)	8.28(0.02)	42.8(14.7)	43.4(14.7)	**8.14**(0.11)
Dom3	7.95(0.09)	30.2(0.04)	7.8(7.4)	33.4(9)	33.8(9.4)	**7.69**(0.12)
Dom4	7.74(0.04)	22.3(7.6)	**7.61**(0.04)	28.5(10.2)	29.9(11)	7.63(0.04)
Dom5	7.56(0.06)	—	7.5(0.05)	23.5(8.3)	24.6(8)	**7.45**(0.09)

We compare with: *(1)* DANN (Ganin *et al.*, 2016) by merging all sources; *(2)* MDAN-Max and *(3)* MDAN-Dyn (Zhao *et al.*, 2018). *(4)* AHD-MSDA (Richard *et al.*, 2020) and its baseline *(5)* AHD-1S that merges all sources. In a hold-out manner, each domain is kept once aside as a target domain while using the remaining domains as sources. The Amazon contains review texts for bought products and their associated ratings. The YearPredictionMSD dataset aims to predict the release year of songs from 90 "timbre" features. A multi-source problem is created by applying k-means (on 30 features), and then mapping the cluster to domains (5 domains), see Appendix C for details on the used architecture and data.

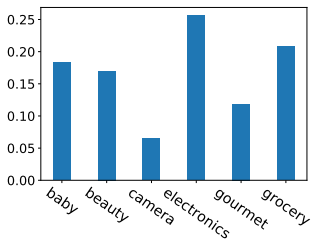


Figure 2: The learned source weight by our MDD when the target is computer&video-games.

The quantitative results on these two datasets are summarized in Table 1 and Table 2, respectively. In general, our MDD can always achieve the smallest mean absolute error for all target domains.

We analyze the weights learned by MDD on the Amazon data. Fig. 2 shows one representative result; when the target domain is "computer&video-games", MDD selects "electronics" as the source with the richest information. This is understandable. First, these two domains have more semantic similarity. Second, "electronics" has $23,009$ training samples, significantly larger than the second largest domain ("camera&photo" with $7,408$ samples). These results confirm that our weight values are interpretable and can reflect the strength of relatedness between each source to the target.

5 Conclusions and Future Work

We introduced the recently proposed von Neumann conditional divergence D_{vN} to match the functional similarity of latent representation $\mathbf{t}$ to response variable y across different domains in multi-source domain adaptation (MSDA). For MSDA, we derived a new generalization bound based on a new loss induced by D_{vN} that gives guarantees for the unsupervised robustness over unlabeled tasks. We proposed a min-max game implementing the resulting objective. In the future, we will explore more the advantage of D_{vN} loss to train deep neural networks. We expect its applications in a variety of machine learning applications, beyond MSDA.

REFERENCES

M. Abadi and P. Barham others. Tensorflow: A system for large-scale machine learning. In *12th USENIX symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283, 2016.

Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*, 2016.

Rana Ali Amjad and Bernhard Claus Geiger. Learning representations for neural network-based classification using the information bottleneck principle. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

Heinz H Bauschke and Jonathan M Borwein. Joint and separate convexity of the bregman distance. In *Studies in Computational Mathematics*, volume 8, pages 23–36. Elsevier, 2001.

S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010.

T. Bertin-Mahieux. Yearpredictionmsd data set, 2011.

J. Blitzer, M. Dredze, and F. Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, pages 440–447, 2007.

C. Cortes and M. Mohri. Domain adaptation and sample bias correction theory and algorithm for regression. *Theoretical Computer Science*, 519:103–126, 2014.

Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

D. Dua and C. Graff. UCI machine learning repository, 2017.

Y. Ganin, E. Ustinova, et al. Domain-adversarial training of neural networks. *JMLR*, 17(1):2096–2030, 2016.

Geoffrey E Hinton and Sam Roweis. Stochastic neighbor embedding. *NeurIPS*, 15:857–864, 2002.

B. Kulis, M. A. Sustik, and I. S. Dhillon. Low-rank kernel learning with bregman matrix divergences. *JMLR*, 10(2), 2009.

Frank Nielsen and Rajendra Bhatia. *Matrix information geometry*. Springer, 2013.

M. A. Nielsen and I. Chuang. Quantum computation and quantum information, 2002.

F. Nielsen and R. Nock. Sided and symmetrized bregman centroids. *IEEE transactions on Information Theory*, 55(6):2882–2904, 2009.

S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2010.

Adam Paszke, S. Gross, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8026–8037, 2019.

Z. Pei, Z. Cao, M. Long, and J. Wang. Multi-adversarial domain adaptation. In *AAAI*, volume 32, 2018.

G. Richard, A. de Mathelin, G. Hébrail, M. Mougeot, and N. Vayatis. Unsupervised multi-source domain adaptation for regression. In *ECML*, 2020.

M. Riemer, I. Cases, et al. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018.

K. Saito, K. Kim, et al. Semi-supervised domain adaptation via minimax entropy. In *IEEE ICCV*, pages 8050–8058, 2019.

J. Taghia, M. Bånkestad, F. Lindsten, and T. B. Schön. Constructing the matrix multilayer perceptron and its application to the vae. *arXiv preprint arXiv:1902.01182*, 2019.

J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *NeurIPS*, pages 3320–3328, 2014.

Y. Yu, A. Shaker, F. Alesiani, and J. C. Principe. Measuring the discrepancy between conditional distributions: Methods, properties and applications. In *IJCAI*, pages 2777–2784, 2020.

Xi Yu, Shujian Yu, and Jose C Principe. Deep deterministic information bottleneck with matrix-based entropy functional. *arXiv preprint arXiv:2102.00533*, 2021.

Werner Zellinger, Thomas Grubinger, Edwin Lughofer, Thomas Natschläger, and Susanne Saminger-Platz. Central moment discrepancy (cmd) for domain-invariant representation learning. *arXiv preprint arXiv:1702.08811*, 2017.

H. Zhao, S. Zhang, et al. Adversarial multiple source domain adaptation. *NeurIPS*, 31:8559–8570, 2018.

H. Zhao, R. T. Des Combes, K. Zhang, and G. Gordon. On learning invariant representations for domain adaptation. In *ICML*, pages 7523–7532, 2019.

S. Zhao, M Gong, et al. Domain generalization via entropy regularization. *NeurIPS*, 33, 2020.

This document contains the supplementary material for the *"Multi-Source Domain Adaptation with von Neumann Conditional Divergence"* manuscript.

A THREE APPEALING PROPERTIES ASSOCIATED WITH J_{vN}

We present three appealing properties associated with J_{vN} (see supplementary material for details and proofs). For three SPD matrices X, Y, Z of the same size, we have:

- $J_{vN}(X:Y)$ has an analytical gradient and is automatically differentiable;
- $\sqrt{J_{vN}(X:Y)} \leq \sqrt{J_{vN}(X:Z)} + \sqrt{J_{vN}(Z:Y)}$ (triangle inequality; see Nielsen and Nock (2009); Taghia *et al.* (2019));
- Taking $X = \sigma_{\mathbf{x},f(\mathbf{x})}$ and $Y = \sigma_{\mathbf{x},\hat{f}(\mathbf{x})}$, $\sqrt{J_{vN}(\sigma_{\mathbf{x},f(\mathbf{x})} : \sigma_{\mathbf{x},\hat{f}(\mathbf{x})})}$ can be interpreted and used as a loss function to train a deep neural network. Here, $\mathbf{x}$ refers to the input variable, $f : \mathbf{x} \rightarrow y$ is the true labeling or mapping function, $\hat{f}$ is the estimated predictor, $f(\mathbf{x}) = y$ is the true label or response variable, $\hat{f}(\mathbf{x}) = \hat{y}$ is the predicted output. $\sigma_{\mathbf{x},f(\mathbf{x})}$ denotes the covariance matrix for the pair of variables $\{\mathbf{x}, f(\mathbf{x})\}$. Similarly, $\sigma_{\mathbf{x},\hat{f}(\mathbf{x})}$ denotes the covariance matrix for the pair of variables $\{\mathbf{x}, \hat{f}(\mathbf{x})\}$. See Fig. 1 an illustrative explanation. Compared with classic loss functions like the mean square error (MSE) loss and the cross-entropy loss, $\sqrt{J_{vN}(\sigma_{\mathbf{x},f(\mathbf{x})} : \sigma_{\mathbf{x},\hat{f}(\mathbf{x})})}$ enjoys more interpretability.

A.1 PROOFS AND ADDITIONAL REMARKS TO PROPERTIES OF THE JEFFERY VON NEUMANN DIVERGENCE J_{vN}

A.1.1 DIFFERENTIABILITY OF $\sqrt{J_{vN}(X;Y)}$

For simplicity, we consider the argument of the square root, i.e., $J_{vN}(X;Y)$.

By definition, we have:

$$D_{vN}(X;Y) = \mathrm{Tr}(X \log X - X \log Y - X + Y), \tag{6}$$

and

$$J_{vN}(X;Y) = \frac{1}{2}\left(D_{vN}(X;Y) + D_{vN}(Y;X)\right) = \frac{1}{2}\,\mathrm{Tr}\left((X-Y)\left(\log X - \log Y\right)\right). \tag{7}$$

We thus have (Nielsen and Bhatia, 2013, Chapter 6):

$$\frac{\partial D_{vN}(X;Y)}{\partial X} = \log X - \log Y, \tag{8}$$

and

$$\frac{\partial D_{vN}(X;Y)}{\partial Y} = -XY^{-1} + I, \tag{9}$$

where I denotes an identity matrix with the same size as X.

Therefore,

$$\frac{\partial J_{vN}(X;Y)}{\partial X} = \frac{1}{2}\left(\log X - \log Y - YX^{-1} + I\right). \tag{10}$$

Since $J_{vN}(X;Y)$ is symmetric, the same applies for $\frac{\partial J_{vN}(X;Y)}{\partial Y}$ with exchanged roles between X and Y.

In practice, taking the gradient of $J_{vN}(X;Y)$ is simple with any automatic differentiation software, like PyTorch (Paszke *et al.*, 2019) or Tensorflow (Abadi and others, 2016). We use PyTorch in this work. We also provide an implementation in the attachment of supplementary material.

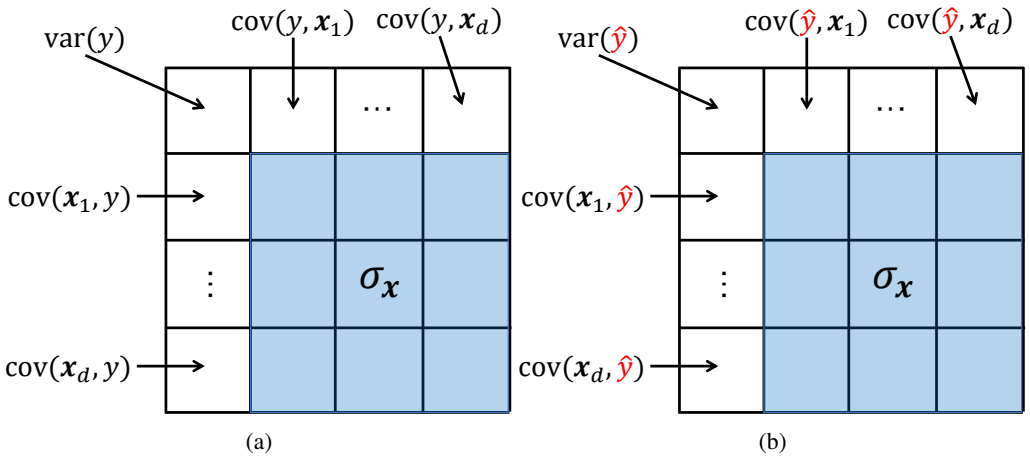


Figure 3: The joint covariance matrices $\sigma_{\mathbf{x},y}$ (a) and $\sigma_{\mathbf{x},\hat{y}}$ (b). Two matrices only differ in the first column and row associated with y or $\hat{y}$.

A.1.2 Interpretability of $\sqrt{J_{vN}(\sigma_{\mathbf{x},f(\mathbf{x})} : \sigma_{\mathbf{x},\hat{f}(\mathbf{x})})}$ as a Loss Function

Again, we consider the argument of the square root, i.e., $J_{vN}(\sigma_{x,y} : \sigma_{x,\hat{y}})$. Suppose $\mathbf{x} \in \mathbb{R}^d$ and $y \in \mathbb{R}$, then both the joint covariance matrices $\sigma_{x,y}$ and $\sigma_{x,\hat{y}}$ are symmetric positive definite and of size $(d+1) \times (d+1)$. At first, one should note that $\sigma_{x,y}$ differs from $\sigma_{x,\hat{y}}$ only in the first row and the first column associated with y (or $\hat{y}$). This is just because the remaining elements of both matrices is the covariance matrix σ_x that only depends on the input. See Fig. 3 for an illustration.

If we look deeper, the first row and column in $\sigma_{x,y}$ (or $\sigma_{x,\hat{y}}$) quantify the variance of y (or $\hat{y}$) and the covariance between y (or $\hat{y}$) and each dimension of x (denote x_i the i-th dimension of x). In this sense, our matrix-based loss reduces to zero if and only if *(i)* the variance of y and $\hat{y}$ are the same; and *(ii)* for an arbitrary dimension x_i, the covariance $cov(y, x_i)$ is the same to the covariance $cov(\hat{y}, x_i)$.

On the other hand, suppose y and $\hat{y}$ are Gaussian distributed with $y \sim N(\mu_y, \sigma_y)$ and $\hat{y} \sim N(\mu_{\hat{y}}, \sigma_{\hat{y}})$, then the Kullback–Leibler (KL) divergence reduces to (Cover, 1999):

$$\begin{aligned} D_{KL}(p(y), p(\hat{y})) &= -\int p(y) \log \left(\frac{p(\hat{y})}{p(y)} \right) dy \\ &= \log \frac{\sigma_{\hat{y}}}{\sigma_y} + \frac{\sigma_y^2 + (\mu_y - \mu_{\hat{y}})^2}{2\sigma_{\hat{y}}^2}. \end{aligned} \tag{11}$$

If y and $\hat{y}$ are mean centered, then the KL divergence only relies on the variance of y and $\hat{y}$.

Moreover, we have:

$$\begin{aligned} D_{KL}(p(y), p(\hat{y})) &= -\int p(y) \log \left(\frac{p(\hat{y})}{p(y)} \right) dy \\ &= -\int p(y) \log \left(p(\hat{y}) \right) dy + \int p(y) \log \left(p(y) \right) dy \\ &= H(p(y), p(\hat{y})) - H(p(y)). \end{aligned} \tag{12}$$

The first term on the r.h.s. of Eq. (12) is exactly the cross entropy, and the second term is the entropy of $p(y)$, a constant that only depends on the training data. In this context, we can view the cross-entropy and the KL divergence are optimizing the same quantity when they are used as loss functions.

To summarize, we can conclude that, in contrast to the popular KL divergence loss or cross-entropy loss that matches $p(y)$ to $p(\hat{y})$, our matrix-based loss adds an additional penalty on $(\hat{y}, x_i)|_{i=1}^d$.

We know that the covariance can be interpreted as a linear dependence (although it is not upper bounded). In this sense, our matrix-based loss also encourages the dependence between each dimension of input and the predicted variable $\hat{y}$ matches to the ground truth. Therefore, when our matrix-based loss is minimized, the trained model is more faithful and enjoys more interpretability.

B Multi-Source Domain Adaptation with Matrix-based Discrepancy Distance

Theorem 2. *Given a set of K source domains $S = \{D_{s_1}, \ldots, D_{s_K}\}$ and denote the ground truth mapping function in D_{s_i} as f_{s_i}. Let us attribute weight w_i to source D_{s_i} (subject to $\sum_{i=1}^{K} w_i = 1$) and generate a weighted source domain D_α, such that the source distribution $P_\alpha = \sum_{i=1}^{K} w_i P_{s_i}$ and the mapping function $f_\alpha : x \to \left(\sum_{i=1}^{K} w_i P_{s_i}(x) f_{s_i}(x)\right) / \left(\sum_{i=1}^{K} w_i P_{s_i}(x)\right)$. For any hypothesis $h \in \mathcal{H}$, the square root of J_{vN} on the target domain D_t is bound in the following way:*

$$\sqrt{J_{vN}(\sigma^t_{x,h(x)} : \sigma^t_{x,f_t(x)})} \leq \sum_{i=1}^{K} w_i \left(\sqrt{J_{vN}(\sigma^s_{x,h(x)} : \sigma^s_{x,f_{s_i}(x)})}\right) + D_{\text{M-disc}}(P_t, P_\alpha; h) + \eta_Q(f_\alpha, f_t), \tag{13}$$

where $\eta_Q(f_\alpha, f_t) = \min_{h^* \in \mathcal{H}} \sqrt{J_{vN}(\sigma^t_{x,h^*(x)} : \sigma^t_{x,f_t(x)})} + \sqrt{J_{vN}(\sigma^\alpha_{x,h^*(x)} : \sigma^\alpha_{x,f_\alpha(x)})}$ is the minimum joint empirical losses on source D_α and the target D_t, achieved by an optimal hypothesis h^*.

Proof of Theorem 2. For the weighted source D_α with distribution P_α and true mapping function f_α, the following bound holds for each $h \in \mathcal{H}$:

$$\sqrt{J_{vN}(\sigma^t_{x,h(x)} \| \sigma^t_{x,f_t(x)})} \leq \sqrt{J_{vN}(\sigma^\alpha_{x,h(x)} \| \sigma^\alpha_{x,f_\alpha(x)})} + \left|\sqrt{J_{vN}(\sigma^t_{x,h(x)} \| \sigma^t_{x,f_t(x)})} - \sqrt{J_{vN}(\sigma^\alpha_{x,h(x)} \| \sigma^\alpha_{x,f_\alpha(x)})}\right| \tag{14}$$

$$\leq \sqrt{J_{vN}(\sigma^\alpha_{x,h(x)} \| \sigma^\alpha_{x,f_\alpha(x)})} + \left|\sqrt{J_{vN}(\sigma^t_{x,h(x)} \| \sigma^t_{x,h^*(x)})} - \sqrt{J_{vN}(\sigma^t_{x,h(x)} \| \sigma^t_{x,f_t(x)})}\right| + \left|\sqrt{J_{vN}(\sigma^\alpha_{x,h(x)} \| \sigma^\alpha_{x,h^*(x)})} - \sqrt{J_{vN}(\sigma^\alpha_{x,h(x)} \| \sigma^\alpha_{x,f_\alpha(x)})}\right| + \left|\sqrt{J_{vN}(\sigma^t_{x,h(x)} \| \sigma^t_{x,h^*(x)})} - \sqrt{J_{vN}(\sigma^\alpha_{x,h(x)} \| \sigma^\alpha_{x,h^*(x)})}\right| \tag{15}$$

$$\leq \sqrt{J_{vN}(\sigma^\alpha_{x,h(x)} \| \sigma^\alpha_{x,f_\alpha(x)})} + \eta_Q(f_\alpha, f_t) + D_{\text{M-disc}}(P_t, P_\alpha; h), \tag{16}$$

where $\eta_Q(f_\alpha, f_t) = \min_{h^* \in \mathcal{H}} \sqrt{J_{vN}(\sigma^t_{x,h^*(x)} \| \sigma^t_{x,f_t(x)})} + \sqrt{J_{vN}(\sigma^\alpha_{x,h^*(x)} \| \sigma^\alpha_{x,f_\alpha(x)})}$ is the minimum joint empirical losses on source D_α and the target D_t, achieved by an optimal hypothesis h^*.

Inequality (14) holds since $\sqrt{J_{vN}}$ is always non-negative. Inequality (16) follows from the triangular inequality of $\sqrt{J_{vN}}$ (i.e., $\left|\sqrt{J_{vN}(\sigma^t_{x,h(x)} \| \sigma^t_{x,h^*(x)})} - \sqrt{J_{vN}(\sigma^t_{x,h(x)} \| \sigma^t_{x,f_t(x)})}\right| \leq \sqrt{J_{vN}(\sigma^t_{x,h^*(x)} \| \sigma^t_{x,f_t(x)})}$ and $\left|\sqrt{J_{vN}(\sigma^\alpha_{x,h(x)} \| \sigma^\alpha_{x,h^*(x)})} - \sqrt{J_{vN}(\sigma^\alpha_{x,h(x)} \| \sigma^\alpha_{x,f_\alpha(x)})}\right| \leq \sqrt{J_{vN}(\sigma^\alpha_{x,h^*(x)} \| \sigma^\alpha_{x,f_\alpha(x)})}$) and $\left|\sqrt{J_{vN}(\sigma^t_{x,h(x)} \| \sigma^t_{x,h^*(x)})} - \sqrt{J_{vN}(\sigma^\alpha_{x,h(x)} \| \sigma^\alpha_{x,h^*(x)})}\right| \leq D_{\text{M-disc}}(P_t, P_\alpha; h)$ by definition of matrix-based discrepancy distance.

On the other hand, by definition we have:

$$f_\alpha(x) = \sum_{i=1}^{K} w_i f_{s_i}(x), \text{s.t.}, \sum_{i}^{K} w_i = 1. \tag{17}$$

Therefore, for each $h \in \mathcal{H}$, we have:

$$f_\alpha(x) - h(x) = \sum_{i=1}^{K} w_i \left(f_{s_i}(x) - h(x) \right), \tag{18}$$

hence, the prediction residual on domain D_α is also a weighted combination of the prediction residual from each source domain D_{s_i}. If one evaluates prediction residual with a convex function ϵ, such as the mean absolute error (MAE) loss, the mean square error (MSE) loss or the loss defined by von Neumann divergence (Bauschke and Borwein, 2001; Nielsen and Bhatia, 2013), it follows that:

$$\epsilon_\alpha(f_\alpha(x), h(x)) \leq \sum_{i=1}^{K} w_i \epsilon_i \left(f_{s_i}(x), h(x) \right). \tag{19}$$

In our case, it suggests that:

$$\sqrt{J_{vN}(\sigma^\alpha_{x,h(x)} \| \sigma^\alpha_{x,f_\alpha(x)})} \leq \sum_{i=1}^{K} w_i \left(\sqrt{J_{vN}(\sigma^s_{x,h(x)} : \sigma^s_{x,f_{s_i}(x)})} \right) \tag{20}$$

Combining inequalities (16) and (20), we conclude the proof.

C Evaluation Details

C.1 Experimental Setting

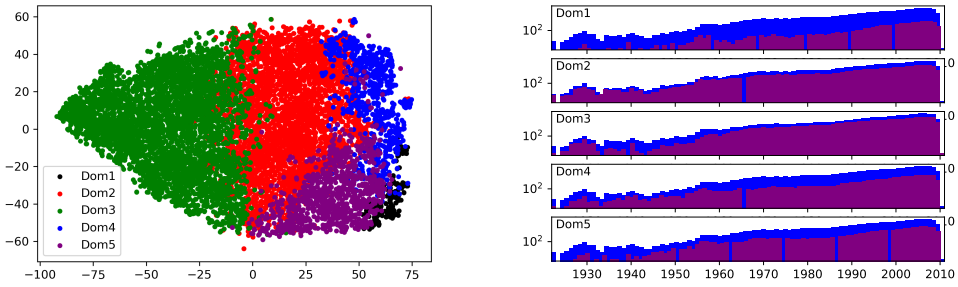
In both experiments, we employ a shallow neural network with two fully-connected hidden layers of size 500, and a dropout rate of 10%. The hidden layers use ReLU activation function. The Adam optimizer is used with learning rate $lr = 0.001$, and batch size of 300. We set the number of training epochs to 20, and we repeat each experiment five times with different random seeds. In a hold-out manner, each domain is kept once aside as a target domain while using the remaining domains as sources.

C.2 Amazon Review Dataset

The Amazon review dataset was originally constructed and introduced in Blitzer *et al.* (2007); it contains review texts for bought products and their associated ratings. Products are grouped into categories. Similar to Zhao *et al.* (2018); Richard *et al.* (2020), we perform tf-idf transformation and select the top 1000 term frequency words.

C.3 YearPredictionMSD Dataset

The task beyond the YearPredictionMSD dataset is to predict the release year of a song based on 90 "timbre" features. It includes about 515k songs with release year ranging from 1922 to 2011. We obtain the version hosted at the UCI repository (Dua and Graff, 2017). In order to create a multi-source problem, we try to create a set of distinctive domains. To this end, we apply k-means on the first 30 features and, thereafter, assign the songs of each cluster to a domain. The resulting domains are $\{\text{Dom}1, \ldots, \text{Dom}5\}$. Figure 4 presents how this approach creates five distinguishable domains when shown in a t-distributed stochastic neighbor embedding (t-SNE) applied on the whole features of the dataset (Hinton and Roweis, 2002); similarly, the histograms depict how the target distributions vary considerably between the different domains.



(a) (b)

Figure 4: The transformation of the YearPredictionMSD data into multiple domains. (Left) t-SNE visualization of the different domains discoverd in the YearPredictionMSD data. (Right) The histograms of the release year in each domain compared to that of the whole dataset (seen in the background in blue). The log scale is applied on the year in the y-axis.