Generating Synthetic Datasets by Interpolating along Generalized Geodesics (Supplementary Material)

Jiaojiao Fan[1] **David Alvarez-Melis**[2]

[1] Georgia Tech, Atlanta, Georgia, USA
[2]Microsoft Research & Harvard University, Cambridge, Massachusetts, USA

A PROOFS

Proof of Lemma 1. By Santambrogio [2017, §4.4], the result holds when $m = 2$. Then Proposition 7.5 in Agueh and Carlier [2011] extends the result to the case of $m > 2$. □

Proof of Proposition 1. Since linear combination preserves cyclically monotonicity, $\sum_{i=1}^m a_i T_i^*(x)$ is the optimal map from ν to ρ_a^G [McCann, 1995]. Then according to the definition of $W_{2,\nu}(\cdot,\cdot)$, we can write

$$W_{2,\nu}^2(\rho_a^G,\nu) = \int \left\| x - \sum_{i=1}^m a_i T_i^*(x) \right\|^2 \mathrm{d}\nu(x). \tag{1}$$

For scalars $p, q_1, \ldots, q_m$, it holds that

$$\begin{aligned}
\left(p - \sum_{i=1}^m a_i q_i\right)^2 &= p^2 + \sum_{i=1}^m a_i^2 q_i^2 - 2\sum_{i=1}^m a_i p q_i + \sum_{i\neq j} a_i a_j q_i q_j \\
&= p^2 + \sum_{i=1}^m (a_i - a_i \sum_{j\neq i} a_j) q_i^2 - 2\sum_{i=1}^m a_i p q_i + \sum_{i\neq j} a_i a_j q_i q_j \\
&= \sum_{i=1}^m a_i (p - q_i)^2 - \frac{1}{2}\sum_{i\neq j} a_i a_j (q_i - q_j)^2.
\end{aligned}$$

Plugging this equality into (1) gives

$$\begin{aligned}
W_{2,\nu}^2(\rho_a^G,\nu) &= \int \left(\sum_{i=1}^m a_i \|x - T_i^*(x)\|^2 - \frac{1}{2}\sum_{i\neq j} a_i a_j \|T_i^*(x) - T_j^*(x)\|^2 \right) \mathrm{d}\nu(x) \\
&= \sum_{i=1}^m a_i \int \|x - T_i^*(x)\|^2 \mathrm{d}\nu(x) - \frac{1}{2}\sum_{i\neq j} a_i a_j \int \|T_i^*(x) - T_j^*(x)\|^2 \mathrm{d}\nu(x) \\
&= \sum_{i=1}^m a_i W_{2,\nu}^2(\mu_i,\nu) - \frac{1}{2}\sum_{i\neq j} a_i a_j W_{2,\nu}^2(\mu_i,\mu_j).
\end{aligned}$$

□

Proof of Proposition 2. Firstly, $\mathcal{W}_{2,Q}$ is symmetric and nonnegative by definition. It is non-degenerate since

$\mathcal{W}_{2,Q}(P_i, P_j) \geq d_{\text{OT}}(P_i, P_j)$ and d_{OT} is a metric. Finally, we show it satisfies the triangular inequality. Indeed,

$$
\begin{aligned}
&\mathcal{W}_{2,Q}(P_1, P_3) \\
&= \left(\int \|x_1 - x_3\|^2 + W_2^2(\alpha_{y_1}, \alpha_{y_3}) \mathrm{d}Q(z) \right)^{1/2} \\
&\leq \left(\int (\|x_1 - x_2\| + \|x_2 - x_3\|)^2 + (W_2(\alpha_{y_1}, \alpha_{y_2}) + W_2(\alpha_{y_2}, \alpha_{y_3}))^2 \mathrm{d}Q(z) \right)^{1/2} \\
&\leq \left(\int \|x_1 - x_2\|^2 + W_2^2(\alpha_{y_1}, \alpha_{y_2}) \mathrm{d}Q(z) \right)^{1/2} + \left(\int \|x_2 - x_3\|^2 + W_2^2(\alpha_{y_2}, \alpha_{y_3}) \mathrm{d}Q(z) \right)^{1/2} \\
&= \mathcal{W}_{2,Q}(P_1, P_2) + \mathcal{W}_{2,Q}(P_2, P_3),
\end{aligned}
$$

where the first inequality is the triangular inequality and the second inequality is the Minkowski inequality. □

B IMPLEMENTATION DETAILS OF OTDD MAP

OTDD barycentric projection We use the implementation `https://github.com/microsoft/otdd` to solve OTDD coupling. The rest part is straightforward.

OTDD neural map To solve the problem (4.1), we parameterize f, G, ℓ to be three neural networks. In NIST dataset experiments, we parameterize f as ResNet [1] from WGAN-QC [Liu et al., 2019], and take feature map G to be UNet[2] [Ronneberger et al., 2015]. We generate the labels $\bar{y}$ with a pre-trained classifier $\ell(\cdot)$, and use a LeNet or VGG-5 with Spinal layers[3] [Kabir et al., 2022] to parameterize $\ell(\cdot)$. In 2D Gaussian mixture experiments, we use Residual MLP to represent all of them.

We remove the discriminator's condition on label to simplify the loss function as

$$
\sup_f \inf_G \int \Big(\underbrace{\|x - G(z)\|_2^2}_{\text{feature loss}} + \underbrace{W_2^2(\alpha_y, \alpha_{\bar{y}})}_{\text{label loss}} \Big) \mathrm{d}Q(z) \underbrace{- \int f(\bar{x}) \mathrm{d}Q(z) + \int f(x') \mathrm{d}P(z')}_{\text{discriminator loss}} .
$$

In this formula, we assume both y and $\bar{y}$ are hard labels, but in practice, the output of $\ell(\cdot)$ is a soft label. Simply taking the `argmax` to get a hard label can break the computational graph, so we replace the label loss $W_2^2(\alpha_y, \alpha_{\bar{y}})$ by $y^\top M \bar{y}$, where y is the one-hot label from dataset Q. And $M \in \mathbb{R}_{\geq 0}^{C_Q \times C_P}$ is the label-to-label matrix where $M(i,j) := W_2^2(\alpha_{y_i}, \alpha_{y_j})$. The matrix M is precomputed before the training, and is frozen during the training.

We pre-train the feature map G to be an identity map before the main adversarial training. We use the Exponential Moving Average[4] of the trained feature maps as the final feature map.

Data processing For all the *NIST datasets, we rescale the images to size 32×32, and repeat their channel 3 times and obtain 3-channel images. We use the default train-test split from `torchvision`. For the VTAB datasets, we use a masked auto-encoder with 196 batches and 1024 embed dimension based on ViT-Large. So the final embedding dimension is $197 \times 1024 = 201728$. We also use the default train-test split from `torchvision`.

Hyperparameters For the experimental results in §5.2, we use the OTDD neural map and train them using Adam optimizer with learning rate 10^{-3} and batch size 64. We train a LeNet for 2000 iterations, and fine-tune for 100 epochs. Regarding the comparison with other baselines in §5.2, for transfer learning methods, we train a SpinalNet for 10^4 iterations, and fine-tune it for 2000 iterations on the test dataset. Training from scratch on the test dataset takes also 2000 iterations. For the results in §5.3, we pre-train the ResNet-18 model for 5 epochs, then fine-tune the model on the few-shot dataset for 10 epochs. During fine-tuning, we still let the whole network tunable. The batch size is 128, and the learning rate is 10^{-3}.

[1] `https://github.com/harryliew/WGAN-QC`
[2] `https://github.com/milesial/Pytorch-UNet`
[3] `https://github.com/dipuk0506/SpinalNet`
[4] `https://github.com/fadel/pytorch_ema`

C DISCUSSIONS OVER COMPLEXITY-ACCURACY TRADE-OFF

We agree that our method is more computationally demanding than Mixup in general. Specifically, we consider Mixup and our methods to occupy different points of a compute-accuracy trade-off characterized by the expressivity of the geodesics between datasets they define. That being said, the trade-off is nevertheless not a prohibitive one, as shown by the fact that we can scale our method to VTAB-sized datasets with a very standard GPU setup.

'Vanilla' mixup with uniform dataset weights is indeed quite cheap (but, as shown in Table 2, considerably worse than alternatives). On the other hand, the version of Mixup that uses the 'optimal' mixture weights (labeled Mixup - optimal in Table 2, and the only Mixup version in Table 1) requires solving Eq. (3), which involves non-trivial computing to obtain OTDD maps. In the context of the trade-off spectrum described above, Mixup with optimal weights is strictly in between vanilla Mixup and OTDD interpolation.

D ADDITIONAL RESULTS

Figure 1: The numbers above images are the labels. In the first labelling method, all 0 MNIST digits are assigned as class "0", and they are labelled as class "7" in the bottom labelling.

D.1 OTDD NEURAL MAP VISUALIZATION

We show the OTDD neural map between 2D Gaussian mixture models with 16 components in Figure 2. This example is very special so that we have the closed-form solution of OTDD map. The feature map is a identity map and the pushforward label is equal to the corresponding class that has the same conditional distribution $p(x|y)$ as source label. For example, the sample from top left corner cluster is still mapped to the top left corner cluster, and the label is changed from blue to orange. This map achieves zero transport cost. Since the transport cost is always non-negative, this map is the optimal OTDD map. However, Asadulaev et al. [2022], Bunne et al. [2022] enforce mapping to preserve the labels, so with their methods, the blue cluster would still map to the blue cluster. Thus their feature map is highly non-convex and more difficult to learn. We refer to Figure 5 in Asadulaev et al. [2022] for their performance on the same example. Compared with them, our pushforward dataset aligns with the target dataset better.

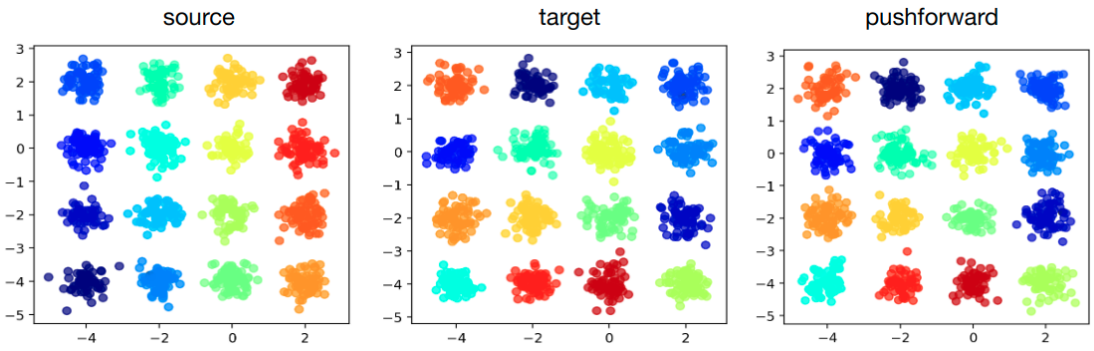

Figure 2: OTDD neural map for 2D Gaussian mixture distributions.

D.2 MCCANN'S INTERPOLATION BETWEEN DATASETS

Our OTDD map can be extended to generate McCann's interpolation between datasets. We propose an anolog of McCann's interpolation (3.1) in the dataset space. We define McCann's interpolation between datasets P_0 and P_1 as

$$P_t^M := ((1-t)\mathrm{Id} + t\mathcal{T}^*)\sharp P_0, \quad t \in [0,1],$$

where $\mathcal{T}^*$ is the optimal OTDD map from P_0 to P_1 and t is the interpolation parameter. The superscript M of P_t^M means McCann. We use the same convex combination method in §4.2 to obtain samples from P_t^M. Assume $(x_0, y_0) \sim P_0$, $(x_1, y_1) = \mathcal{T}^*(x_0, y_0)$ and P_0, P_1 contain 7, 3 classes respectively, i.e. $y_0 \in \{0,1\}^7, y_1 \in \{0,1\}^3$. Then the combination of features is $x_t = (1-t)x_0 + tx_1$, and the combination of labels is

$$y_t = (1-t)\begin{bmatrix} y_0 \\ \mathbf{0}_3 \end{bmatrix} + t\begin{bmatrix} \mathbf{0}_7 \\ y_1 \end{bmatrix}.$$

Thus (x_t, y_t) is a sample from $((1-t)\mathrm{Id} + t\mathcal{T}^*)\sharp P_0$. We visualize McCann's interpolation between two Gaussian mixture distributions in Figure 3. This method can map the labeled data from one dataset to another, and do the interpolation between them. Thus we can use it to map abundant data from an external dataset, to a scarce dataset for data augmentation. For example, in Figure 4, the target dataset only has 30 samples, but the source dataset has 60000 samples. We learn the OTDD neural map between them and solve their interpolation. We find that P_1^M creates new data out of the domain of the original target distribution, which Mixup [Zhang et al., 2018] can not achieve. Thus, the data from P_t^M for t close to 1.0 can enrich the target dataset, and be potentially used in data augmentation for classification tasks.

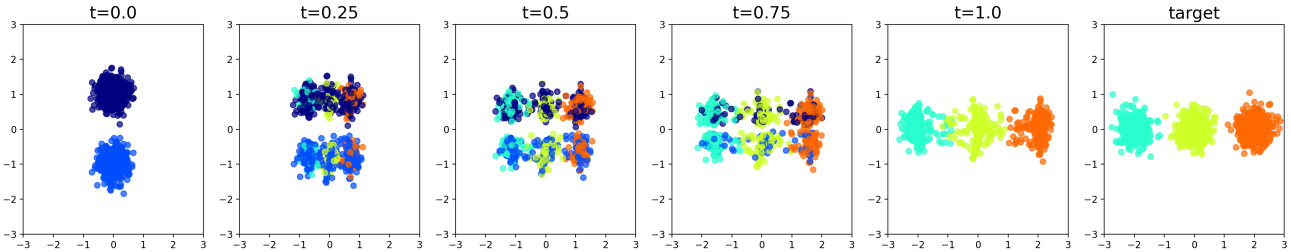

Figure 3: McCann's interpolation for 2D labelled datasets. Each color represents a class. When $t \to 1.0$, the samples within blue classes become less and less, and finally disappear when $t = 1.0$.

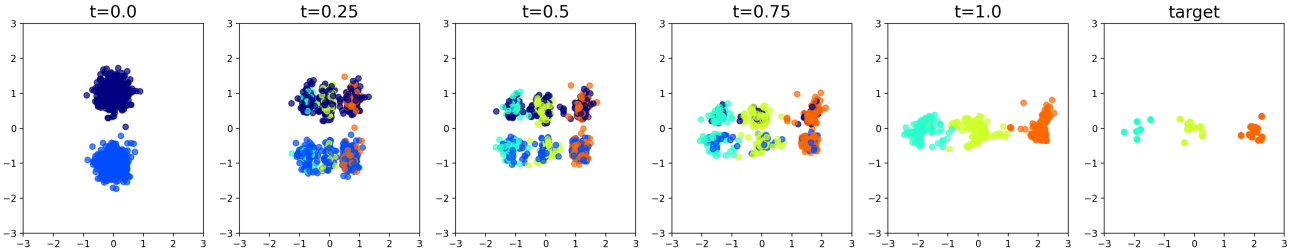

Figure 4: Data augmentation by mapping an external dataset to a few-shot dataset.

D.3 CORRELATION STUDY OF *NIST EXPERIMENTS

A more concrete visualization of the correlation between $\mathcal{W}^2(P_a, Q)$ and *NIST transfer learning test accuracy is shown in Figure 5. Among all datasets, USPS and KMNIST lack correlation. We believe it's caused by (i) small variance in the distances from pretraining dataset to target dataset, implying a limited relative diversity of datasets on which to draw on and (ii) (in the case of USPS) a very simple task where baseline accuracy is already very high and hard to improve upon via transfer.

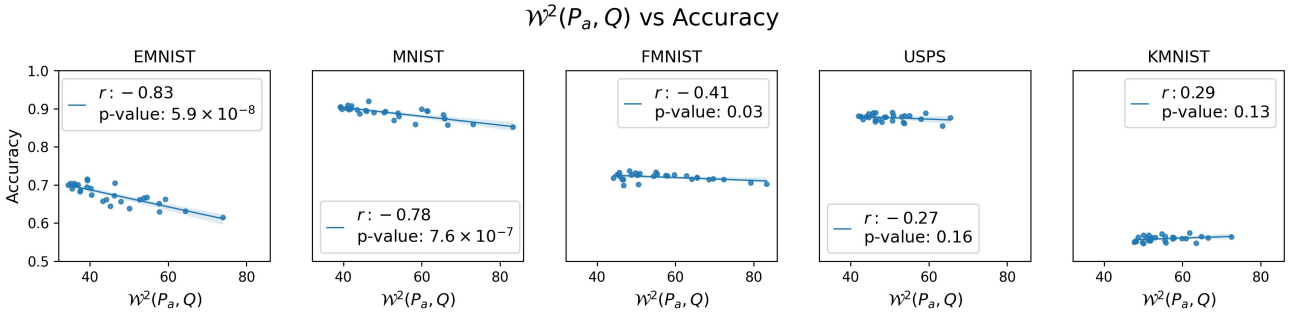

Figure 5: Pearson correlation between the (averaged) function $\mathcal{W}^2(P_a, Q)$ and the test accuracy of the fine-tuned model. Most datasets present a negative correlation between $\mathcal{W}^2(P_a, Q)$ and the accuracy. When test dataset is USPS or KMNIST (rightmost two), all three training datasets are similarly distant to the test dataset; thus, the range of $\mathcal{W}^2(P_a, Q)$ is not wide enough to show an obvious negative correlation. This explains the nearly zero slope and relatively large p-value for those two datasets. Similar pattern has been observed in Yeaton et al. [2022, Figure 5(a)].

D.4 FINE-GRAINED ANALYSIS OVER $\mathcal{W}^2(P_a, W)$ IN *NIST EXPERIMENTS

In Table 1, we provide a more fine-grained analysis for different aspects of $\mathcal{W}(P_a, Q)$ and their effect on transfer accuracy. To do so, we provide the min, median, range, and standard deviation of $\mathcal{W}(P_a, Q)$ in the table below. In addition, as a proxy for the hardness / best possible gain from transfer learning, we show in the last column *OTDD accuracy* minus *few shot accuracy*, where *OTDD accuracy* and *few shot accuracy* are the mean accuracies in Rows 1 and 4, respectively, in Table 1.

Based on these statistics, we make the following observations on the relation between $\mathcal{W}(P_a, Q)$ and transfer accuracy:

- The accuracy improvement is strongly driven by $\min_a \mathcal{W}(P_a, Q)$. EMNIST and MNIST are with relatively smaller $\min_a \mathcal{W}(P_a, Q)$ and share the largest improvement margin. On the other hand, FMNIST and KMNIST as Q have the largest $\mathcal{W}(P_a, Q)$ to the other pre-training datasets, and have relatively smaller accuracy gain. In other words, the correlation between distance and accuracy is stronger in the part of the convex dataset polytope that is closest to the target dataset.

- The strength of the correlation between $\mathcal{W}(P_a, Q)$ and accuracy seems to depend on the **range** and **standard deviation** of he former. On the one hand, **settings with low dynamic range in $\mathcal{W}(P_a, Q)$ (like USPS and EMNIST) make it harder to observe meaningful differences in accuracy**. On the other hand, this indicates that those datasets are roughly (or at least **more**) equidistant from all pretraining datasets, and therefore any convex combination of them will also be close to equidistant from the target, yielding no visible improvement.

- Intrinsic task hardness matters. Consider USPS: all pretraining datasets, regardless of distance, seem to yield very similar accuracy on it, and it has the lowest accuracy gain (only $\sim$5%) among 5 tasks. But considering that the no-transfer (i.e. 5-shot) accuracy is already almost 81%, it is clear that the benefit from transfer learning is "a priori" limited, and therefore all pretraining datasets yield a similar minor improvement.

Table 1: Statistics of $\mathcal{W}(P_a, Q)$ and transfer accuracy in *NIST experiments (§5.2).

Test dataset	Mean of $\mathcal{W}(P_a, Q)$	Median of $\mathcal{W}(P_a, Q)$	Range of $\mathcal{W}(P_a, Q)$	Standard deviation of $\mathcal{W}(P_a, Q)$	Mean of accuracy improvement
EMNIST	34.41	43.71	39.58	9.94	13.46
MNIST	39.13	49.04	44.17	11.35	20.94
FMNIST	44.19	54.75	39.11	10.64	10.62
USPS	42.04	48.32	**23.49**	**6.13**	5.28
KMNIST	47.65	53.92	**24.83**	**6.19**	10.88

D.5 FULL RESULTS OF VTAB EXPERIMENTS

In Section 5.3, we only showed the relative improvement of the test accuracy compared to non-pretraining. Here we will show the full test accuracy results. We keep the hyper-parameters consistent through all pre-training datasets. Table 2 clearly shows that the interpolation dataset with optimal weight assigned by our method can have a better performance than a naïve uniform weight. And with the same weight, our OTDD map will give a higher accuracy than Mixup because Mixup does not use the information from the reference dataset (see Figure 4).

Poor sub-pooling performance We show the sub-pooling baseline as a non-trivial method to combine datasets. However, it performs poorly, and we believe there are two main reasons for this. First, this baseline wastes relevant label data, by discarding the original labels of the pretraining dataset and replacing them with the inputted nearest-neighbor label from the target examples. Secondly, it only uses the neighbors of the pet dataset, leaving all other datapoints unused.

Table 2: Test accuracy (mean $\pm$ std over 5 runs in percent) of 1000-shot learning on Oxford-IIIT Pet test dataset. Non-transfer learning skips the pre-training step.

Transfer learning	OTDD map (optimal weight)	**22.60 $\pm$ 1.01**
	OTDD map (uniform weight)	21.06 $\pm$ 0.45
	Mixup (optimal weight)	17.45 $\pm$ 2.2
	Mixup (uniform weight)	15.4 $\pm$ 1.56
	CALTECH101	18.24 $\pm$ 3.42
	DTD	11.46 $\pm$ 0.68
	FLOWERS102	11.11 $\pm$ 1.92
	POOLING	14.88 $\pm$ 0.57
	SUB-POOLING	14.88 $\pm$ 0.57
Non-transfer learning		11.71 $\pm$ 1.65

References

Martial Agueh and Guillaume Carlier. Barycenters in the wasserstein space. *SIAM Journal on Mathematical Analysis*, 43 (2):904–924, 2011.

Arip Asadulaev, Alexander Korotin, Vage Egiazarian, and Evgeny Burnaev. Neural optimal transport with general cost functionals. *arXiv preprint arXiv:2205.15403*, 2022.

Charlotte Bunne, Andreas Krause, and Marco Cuturi. Supervised training of conditional monge maps. *arXiv preprint arXiv:2206.14262*, 2022.

HM Dipu Kabir, Moloud Abdar, Abbas Khosravi, Seyed Mohammad Jafar Jalali, Amir F Atiya, Saeid Nahavandi, and Dipti Srinivasan. Spinalnet: Deep neural network with gradual input. *IEEE Transactions on Artificial Intelligence*, 2022.

Huidong Liu, Xianfeng Gu, and Dimitris Samaras. Wasserstein gan with quadratic transport cost. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4832–4841, 2019.

Robert J McCann. Existence and uniqueness of monotone measure-preserving maps. *Duke Mathematical Journal*, 80(2): 309–323, 1995.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

Filippo Santambrogio. {Euclidean, metric, and Wasserstein} gradient flows: an overview. *Bulletin of Mathematical Sciences*, 7(1):87–154, 2017.

Anna Yeaton, Rahul G Krishnan, Rebecca Mieloszyk, David Alvarez-Melis, and Grace Huynh. Hierarchical optimal transport for comparing histopathology datasets. *arXiv preprint arXiv:2204.08324*, 2022.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.