Permutation invariant networks to learn Wasserstein metrics

Arijit Sehanobish
Internal Medicine (Cardiology) and Computer Science
Yale University
arijit.sehanobish@yale.edu

Neal G. Ravindra
Internal Medicine (Cardiology) and Computer Science
Yale University
neal.ravindra@yale.edu

David van Dijk
Internal Medicine (Cardiology) and Computer Science
Yale University
david.vandijk@yale.edu

Abstract

Understanding the space of probability measures on a metric space equipped with a Wasserstein distance is one of the fundamental questions in mathematical analysis. The Wasserstein metric has received a lot of attention in the machine learning community especially for its principled way of comparing distributions. In this work, we use a permutation invariant network to map samples from probability measures into a low-dimensional space such that the Euclidean distance between the encoded samples reflects the Wasserstein distance between probability measures. We show that our network can generalize to correctly compute distances between unseen densities. We also show that these networks can learn the first and the second moments of probability distributions.

1 Introduction

The Wasserstein distance is a distance function between probability measures on a metric space $\mathcal{X}$. It is a natural way to compare the probability distributions of two variables $X$ and $Y$, where one variable is derived from the other by small, non-uniform perturbations, while strongly reflecting the metric of the underlying space $\mathcal{X}$. It can also be used to compare discrete distributions. The Wasserstein distance enjoys a number of useful properties, which likely contributes to its wide-spread interest amongst mathematicians and computer scientists [1, 2, 3, 4, 5, 6, 7, 8, 9]. However, despite it’s broad use, the Wasserstein distance has several problems. For one, it is computationally expensive. Second, the Wasserstein distance is not Hadamard differentiable, which can present serious challenges when trying to use it in machine learning. Third, the distance is not robust. To alleviate these problems, one can use various regularized entropies to compute an approximation of this Wasserstein distance. Such an approach is more tractable and also enjoys several nice properties [10, 11, 12].

In this short article, we are interested in learning about the Wasserstein space of order $p$, i.e. an infinite dimensional space of all probability measures with up to $p$-th order finite moments on a complete and separable metric space $\mathcal{X}$. More specifically, we asked, (I) can we propose a neural network that correctly computes the Wasserstein distance between 2 measures, even if both of them...
are not in our training examples? (2) What properties of the measures does such a network learn? For example, does it learn something about the moments of these distributions? (3) What properties of the original Wasserstein space can we preserve in our encoded space?

There has been a lot of work in understanding the space of Gaussian processes [13, 14] but our work is more similar to, which attempts to understand Wasserstein spaces with neural networks [15] [16]. Like [15], we use a Siamese network to compare and contrast various densities but the questions we address in this article are different than that of [15]. Furthermore, we try to approximate the Wasserstein space by learning a mapping from the space to a low dimensional Euclidean space, unlike [16], where they learn a mapping from an Euclidean space to the Wasserstein space.

2 Theory

Let \( \mathcal{X} \) be a complete and separable metric space. For simplicity, we take \( \mathcal{X} \) to be \( \mathbb{R}^n \) or a closed and bounded subset of \( \mathbb{R}^n \). Let \( \mathbb{P}(\mathcal{X}) \) be the space of all probability measures on \( \mathcal{X} \). One can endow the space \( \mathbb{P}(\mathcal{X}) \) with a family of metrics called the Wasserstein metrics \( W_p \).

\[
W_p(\mu, \nu) = \inf_{\gamma \sim \mu \nu} \mathbb{E}(|X - Y|^p)^{1/p}, \quad p \geq 1
\]

We use the notations \( W_p(X, Y) \) and \( W_p(\mu, \nu) \) interchangeably whenever \( X \sim \mu \) and \( Y \sim \nu \). We also assume that \( \mathbb{E}(|X|^p) \) and \( \mathbb{E}(|Y|^p) \) is finite. Most of the following properties regarding the space \( \mathbb{P}(\mathcal{X}) \) and \( W_p \) are well-known but we summarize them for the convenience of the reader [17, 18].

**Theorem 2.1.** (i) \( \mathbb{P}(\mathcal{X}) \) equipped with \( W_p \) is a complete and separable metric space. 
(ii) If \( X \) and \( Y \) are degenerate at \( x, y \in \mathcal{X} \), then \( W_p(X, Y) = |x - y| \).
(iii) (Scaling law) For any \( a \in \mathbb{R} \), \( W_p(aX, aY) = |a| W_p(X, Y) \).
(iv) (Translation invariance) For any \( x \in \mathcal{X} \), \( W_p(X + x, Y + x) = W_p(X, Y) \).
(v) \( \mathbb{P}(\mathcal{X}) \) is flat metric space under \( W_1 \) but the sectional curvature is non-negative under \( W_2 \).

**Proof.** See Section 2 in [18].

**Theorem 2.2.** (Topology generated by \( W_p \)) (i) If \( \mathcal{X} \subset \mathbb{R}^n \) is compact and \( p \in [1, \infty) \), in the space \( \mathbb{P}(\mathcal{X}) \), we have \( \mu_k \to \mu \) iff \( W_p(\mu_k, \mu) \to 0 \).
(ii) If \( \mathcal{X} = \mathbb{R}^n \), then \( W_p(\mu_k, \mu) \to 0 \) iff \( \mu_k \to \mu \) and \( \int |x|^p d\mu_k \to \int |x|^p d\mu \).

**Proof.** See proofs associated with Theorem 5.10 and Theorem 5.11 in [17].

The measures \( \mu \) and \( \nu \) are rarely known in practice. Instead, one has access to finite samples \( \{x_i\} = X \sim \mu \) and \( \{y_j\} = Y \sim \nu \). We then construct discrete measures \( \mu := \sum_{i=1}^n a_i \delta_{x_i} \) and \( \nu := \sum_{j=1}^m b_j \delta_{y_j} \), where \( a, b \) are vectors in the probability simplex, and the pairwise costs can be compactly represented as an \( n \times m \) matrix \( C \), i. e., \( c_{ij} := c(x_i, y_j) \) where \( c \) is the metric of the underlying space \( \mathcal{X} \). Since the marginals here are fixed to be the laws of \( X \) and \( Y \), the problem is to find a copula [19] that couples \( X \) and \( Y \) together as “tightly” as possible in an \( L^p \)-sense, on average; if \( p = 2 \) then the copula one seeks is the one that maximizes the correlation (or covariance) between \( X \) and \( Y \), i.e., the copula inducing maximal linear dependence. Solving the above problem scales cubically on the sample sizes and is extremely difficult in practice. Adding an entropy regularization, leads to a problem that can be solved much more efficiently [10] [11] [12]. In this article, we use the Sinkhorn distance \( SD^\lambda \) and their computation, as in [11]. For more details about the entropic regularization, please see Appendix C. The Sinkhorn distance however is not a true metric [14] and fails to satisfy \( SD^\lambda(\mu, \nu) = 0 \). The technical workaround this issue is explained in Appendix D. Moreover, the Sinkhorn distance requires discretizing the space, which alters the metric. The goal of this paper is to see how well a neural network, trained using the Sinkhorn distance, can capture the topology of the \( \mathbb{P}(\mathcal{X}) \) under \( W_p \).

3 Neural Networks to understand \( \mathbb{P}(\mathcal{X}) \)

We draw random samples with replacement of size \( N \) from various distributions in \( \mathbb{P}(\mathcal{X}) \). For technical reasons (described in Appendix D), we only use continuous distributions during the training process. We use the DeepSets architecture [20] to encode this set of \( N \) elements as we want an
encoding that is invariant of the permutations of the samples. More precisely, if \( X \sim \mu \) and \( Y \sim \nu \), \((\mu = \nu \) is allowed, but \( X \) and \( Y \) are drawn independently) and we denote the set of samples drawn from \( \mu \) as \( S_X \) (similarly \( S_\nu \)), we train the encoder \( H_\theta \) such that,

\[
||H_\theta(S_X) - H_\theta(S_Y)|| = SD_\lambda^\mu(\mu, \nu)
\]

Thus, the loss function becomes,

\[
L_{wass} = \frac{1}{2} \sum \left( ||H_\theta(S_X) - H_\theta(S_Y)|| - SD_\lambda^\mu(\mu, \nu) \right)^2
\]

where \( m \) is the size of the mini-batch and we pick 2 sets at random from the mini-batch to compare distances. One can think of our network as a Siamese Network [21] with a DeepSet backbone which allows us to compare and contrast samples drawn from same or different distributions. Our work can be thought as next-generation functional data analysis [22] (Section 6). More details about the network architecture can be found in Appendix B. The code is available at [https://github.com/arijitthegame/encoding_wasserstein_metrics](https://github.com/arijitthegame/encoding_wasserstein_metrics).

3.1 Regularizers for ensuring better properties

If \( X' = X + x \), then \( S_{X'} \) is a set of samples \( X \) after translation \( x \) (this similarly applies for \( Y' \) and \( S_{Y'} \)). To ensure the properties of \( W_p \) are reflected in our computed Euclidean distance, we demand that,

1. \( ||H_\theta(S_{X'}) - H_\theta(S_Y)|| = ||H_\theta(S_X) - H_\theta(S_Y)|| \)
2. \( ||H_\theta(S_{aX}) - H_\theta(S_{aY})|| = |a||H_\theta(S_X) - H_\theta(S_Y)||. \)

These constraints comprise the loss function

\[
\mathcal{L} := L_{wass} + \frac{1}{2} \sum \left( ||H_\theta(S_{X'}) - H_\theta(S_Y)|| - ||H_\theta(S_X) - H_\theta(S_Y)|| \right)^2
\]

4 Experiments

In this section we will describe our toy examples and show the discriminative behavior of our Neural Networks and the interesting properties of the space it can uncover. Our datasets are the following:

1. Random samples of size 500 drawn independently about \( 50 \) times from uniform, Normal, Beta, Gamma, Exponential, Laplace, Log Normal and mixtures of Gaussian distributions with varying parameters.
2. Random samples of size 300 drawn independently about \( 100 \) times from \( 2D \) Normal distributions with various \( \mu, \Sigma \). Fig 1 (B) shows the embedding our datasets by our model. In Fig 2 we show how well the neural network approximates the Sinkhorn distances from samples drawn from our test densities. All the results shown here are with the \( W_1 \) metric. Other plots showing how well our network respects the scaling law (iii) in Theorem 2.1 and the results with the \( W_2 \) metric are shown.
in the Appendix A. Detailed quantitative results can be found in Appendix F. Some of our results with the $W_2$ metric are weaker than the ones with the $W_1$ metric. This may be due to the following reasons: $\mathbb{P}(\mathcal{X})$ under $W_2$ is no longer flat and the Sinkhorn distance changes the metric differently than it changes the space under $W_1$; secondly, since our target is an Euclidean space which is a flat space, we are losing more structural information when mapping from the 2-Wasserstein space.

**Generalizing to out-of-sample-densities** : We also show that our model can generalize well to densities that are out of our training set. These densities are primarily constructed from the training densities but by changing the parameters (Fig 2B,C). But even more interestingly, we found that our model can correctly measure the distance between 2 Dirac measures and distance between 2 Binomial densities, even though they are not a part of the training densities.

**Translating samples** : Given 2 samples $X \sim \mu, Y \sim \nu$, we can translate them around by a random vector $a$, to create new samples $X' := X + a, Y' := Y + a$, under property 4 in Theorem 3.1, $H_\theta(X), H_\theta(Y), H_\theta(X'), H_\theta(Y')$ would form a parallelogram. Fig 2D shows the exact relationship between the distances of encoded translated samples and the encoded samples. Furthermore, we took samples from a 2D Normal Distribution $N(\mu, \Sigma)$ and rotated it around by using a circle, i.e. created new samples via $X' := X + (\cos(\phi), \sin(\phi))$ and we found that the encoded translated samples also formed a circular pattern around the original encoded sample. Thus our simple examples show that our metric preserves the translation invariance property and some geometry of the space (Figure 2E).

**Learning statistical properties of the measures** : Surprisingly for encoded 1D-distributions, we found the strong correlation between means (and variances) of the distributions and the $x$-coordinate (and $y$-coordinate) of the encoded point (Fig 3A,B). That explains why the encoded Dirac distribution at $0$ and Normal distribution with mean $0$ and standard deviation $\sigma$ has $x$-coordinates close to each other. An open question and an interesting future work will to be understand if we can capture higher moments as we increase the output dimension.

**Respecting the topology of the space** : We know that the Dirac delta measure is the limit of Gaussian measures under the weak convergence of measures. Choosing samples drawn from $N(0, 1/n)$ we can see that our encoded points converge to the point encoded by the Dirac measure (Fig 3C). This gives us an empirical evidence that our neural network may be continuous with respect to the Wasserstein metric.

**Wasserstein barycenters** : Given two densities $\mu_1, \mu_2$, if $\hat{\mu}$ is their Wasserstein barycenter [23, 24, 25, 26, 27], our aim is to show that $H_\theta(\hat{\mu})$ can be approximated by the midpoint of the line joining...

---

*Figure 2: (A–C) Pearson’s $r$ correlation coefficient for association between embedded and Sinkhorn distances (color code in Appendix A). (D) Correlation after translations. (E) Samples from a multivariate normal distribution translated around a circular path.*
$H_0(\mu_1)$ and $H_0(\mu_2)$. Fig 3D shows the following examples of this claim: **1)** Samples drawn from $N(0, 1)$ and $N(1, 1)$. **2)** Dirac at 0 and 1. **3)** Uniform distribution in [0, 1] and in [0.8, 1].

![Figure 3: Person’s r comparing embedding axes to means (A) and standard deviations (B). (C) Convergence of samples from Normal distributions with various standard deviations to the Dirac distribution encoding. (D) Barycenters of distributions (left) and midpoints drawn between lines connecting the encoded samples (right).](image)

We also note that none of the measures used above are in the set of our training measures. And finally observe that the figure also shows the correlation between $x$-coordinates and means of the chosen measures. Finally, the experiment also show that we can approximate the Wasserstein geodesic by straight lines in our encoded space.

### 5 Effect of the regularizers

In this section we show the effect of the regularizers in generalizing to unseen distributions and also how well our model learns the Translation law. We show that by just adding the regularizer that enforces the Scaling law, we get better performance in learning the Translation property and generalization to out of sample distributions than the vanilla model which has no regularizers.

![Figure 4: Pearson’s r correlation to compare distances using model trained with (A) scaling regularizer, (B) no regularizer. Out of sample distribution generalization plots for models trained with (C) scaling regularizer, (D) no regularizer.](image)

### 6 Conclusion and Future Work

In this work we showed that we learned a metric by approximating the Wasserstein distance by Sinkhorn distance that obeys the translation invariance and also generalizes to some unseen measures. For 1D measures, we found strong correlation between the encoded vectors $x$-coordinates (resp. $y$-coordinates) with means (variance) of the samples. We are excited by these toy results and would like to prove continuity properties of our neural network. We have also shown our model in general performs better when trained with $W_1$ metric and we would like to understand what role does the topology of the Wasserstein space play in the difference in performances of our model. Finally we would like to investigate if our model can learn higher moments as we increase the output dimension and finally to quantify the distortion of the original Wasserstein metric by our embedding.
Acknowledgements

The first author wants to thank Alexander Cloninger for helpful suggestions and for suggesting to study the geometry of the Wasserstein space by simple translations and scalings. The authors would also like to thank the anonymous reviewers for helpful comments and suggestions.

References


[27] Hermann Karcher. Riemannian Center of Mass and so called karcher mean, 2014.


In this section we show some additional figures: The correlation plots colored by the densities (similar to Fig 2, in the main text, but with full legend). We also show our results for the same experiments discussed earlier with the $W_2$ metric (Fig 6). We found a strong correlation between the variances of the densities and the $y$-coordinates of the encoded points. However unlike in the $W_1$ case, we found no such relations between the means and the $x$-coordinates. Finally we show in Figure 7 how our network has learnt to respect the scaling law ((iii) in Theorem 2.1).

**A Additional Figures**

**B Details about Network Architecture**

We use the DeepSets architecture which basically consists of two blocks of linear layers (with non-linearities between them) $\phi$ and $\rho$. $\phi$ consists of 3 linear layers with hidden sizes 50, 100, 36 with non-linearity Elu. We sum over the outputs of $\phi$ as in to ensure permutation invariance before passing it to the next network $\rho$. $\rho$ consists of 3 linear layers of hidden sizes 30, 30, 10 with an output layer of dimension 2. Elu activation is added to each hidden layer. We trained the model for 500 epochs with Adam optimizer and learning rate $10^{-3}$. 

Figure 5: Pearson’s $r$ correlation coefficient for association between embedded and Sinkhorn distances
C Optimal Transport

The optimization problem defining the distance (Equation 1) is popularly known as optimal transport or the Monge–Kantorovich problem. The Kantorovich formulation \[33\] of the transportation problem...
We use the algorithm in [38], as well as the Geomloss library to compute the barycenters. Another way to view our experiments with the barycenters is to measure how closely we can discretize the space, it can still approximate Wasserstein geodesics via straight lines in our encoded Euclidean space. The results show a better approximation under Wasserstein Barycenters and can be efficiently solved by the Sinkhorn algorithm. In our work λ = 10.

### D Some technical considerations

Note that $SD_p^λ$ is not a true metric as it do not satisfy $SD_p^λ(X, X) = 0$, for all sets of samples. However it is symmetric and satisfies the triangle inequalities. We circumvent this issue by only using continuous measures during training time. This ensures that any of two sets of samples drawn a given measure are distinct with probability 1. Thus, during training we never encounter the set $X$ twice, so a case where $||H_δ(X) − H_δ(X)|| = SD_p^λ(X, X)$ never arises. Thus we end up learning a metric space where the distances between different samples are approximately equal to the Sinkhorn distance.

### E Wasserstein Barycenters

Given measures $μ_1, \cdots, μ_N$, we define the Wasserstein barycenter as the minimizer of the functional

$$F[ν] = \sum_{i=1}^N w_i W_p^\lambda((ν, μ_i))$$

where $w_i$ are some fixed weights and $\sum_i w_i = 1$. For simplicity, we will take the weights to be $1/N$. We use the algorithm in [38], as well as the Geomloss library to compute the barycenters. Another way to view our experiments with the barycenters is to measure how closely we can approximate Wasserstein geodesics via straight lines in our encoded Euclidean space. The results show a better approximation under $W_1$ metric than the $W_2$ metric. We hypothesize since $SD_p^λ$ discretize the space, it can still approximate $W_1$ metric since $P(X)$ is flat. However a straight line is poor approximation for geodesics in a non-flat space, which explains our poor results for the $W_2$ metric.

Finally our experiments with the barycenters suggest a natural way to embed measures in our 2-dimensional encoded space. Take random samples of size $N$ and repeat this process $M$ times. Our encoder will take in these $M \times N$ samples and produce $M$ points. We can take the centroid of these points and use it to get a representation of our measure.
Quantitative Results

In this section we report our quantitative results of our models in various tasks as well as performance of our models without the regularizers. The numbers reported below are the RMSE errors between true distances and the our calculated distances. All the experiments were run 5 times.

### Table 1: Results of our experiments on 1D measures

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Model trained with $W_1$ (no regs)</th>
<th>Model trained with $W_1$ (scaling reg)</th>
<th>Model trained with $W_2$ (no regs)</th>
<th>Model trained with $W_2$ (scaling reg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>In sample densities</td>
<td>.01 ± .007</td>
<td>.04 ± .005</td>
<td>.06 ± .008</td>
<td>.1 ± .003</td>
</tr>
<tr>
<td>Out-of-sample densities</td>
<td>.02 ± .009</td>
<td>.1 ± .004</td>
<td>.1 ± .001</td>
<td>.19 ± .03</td>
</tr>
<tr>
<td>Translation property</td>
<td>.01 ± .004</td>
<td>.07 ± .006</td>
<td>.04 ± .008</td>
<td>.09 ± .004</td>
</tr>
<tr>
<td>Scaling property</td>
<td>.01 ± .003</td>
<td>.08 ± .006</td>
<td>.05 ± .002</td>
<td>.12 ± .001</td>
</tr>
<tr>
<td>Barycenter Accuracy</td>
<td>.05 ± .008</td>
<td>.09 ± .007</td>
<td>.08 ± .009</td>
<td>.13 ± .005</td>
</tr>
</tbody>
</table>

The results on translation and scaling properties are calculated on both in sample densities and out of sample densities. The performance on both the datasets are similar so we pooled all the results and reported the average scores and the standard deviations (10 trials, 5 with in-training densities and 5 with out-of-sample densities).

### Table 2: Results of our experiments on 2D measures

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Model trained with $W_1$ (no regs)</th>
<th>Model trained with $W_1$ (scaling reg)</th>
<th>Model trained with $W_2$ (no regs)</th>
<th>Model trained with $W_2$ (scaling reg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>In sample densities</td>
<td>.02 ± .005</td>
<td>.06 ± .003</td>
<td>.03 ± .008</td>
<td>.15 ± .001</td>
</tr>
<tr>
<td>Out-of-sample densities</td>
<td>.05 ± .004</td>
<td>.13 ± .004</td>
<td>.08 ± .001</td>
<td>.2 ± .001</td>
</tr>
<tr>
<td>Translation property</td>
<td>.09 ± .004</td>
<td>.15 ± .002</td>
<td>.04 ± .008</td>
<td>.19 ± .004</td>
</tr>
<tr>
<td>Scaling property</td>
<td>.07 ± .01</td>
<td>.14 ± .002</td>
<td>.08 ± .009</td>
<td>.21 ± .001</td>
</tr>
</tbody>
</table>

The size of the sample plays a crucial role here. What is the right size of samples to pick? If the size of the samples $X \sim \mu$, $Y \sim \nu$ are large then our method works well. But picking a large sample size is computationally very expensive. We found a sample size of 500 yields good results while a sample size of below 100 yields inconsistent results (variance is high).

### Sampling sizes

The size of the sample plays a crucial role here. What is the right size of samples to pick? If the size of the samples $X \sim \mu$, $Y \sim \nu$ are large then our method works well. But picking a large sample size is computationally very expensive. We found a sample size of 500 yields good results while a sample size of below 100 yields inconsistent results (variance is high).