Annales Univ. Sci. Budapest., Sect. Comp. 57 (2024) 101–115

GLOBAL SINKHORN AUTOENCODER — OPTIMAL TRANSPORT ON THE LATENT REPRESENTATION OF THE FULL DATASET

Adrián Csiszárik, Melinda F. Kiss, Balázs Maga,
Ákos Matszangosz and Dániel Varga
(Budapest, Hungary)

Communicated by Péter Burcsi
(Received January 5, 2024; accepted June 26, 2024)

Abstract. We propose an Optimal Transport (OT)-based generative model from the Wasserstein Autoencoder (WAE) family of models, with the following innovative property: the optimization of the latent point positions takes place over the full training dataset rather than over a minibatch. Our contributions are the following:

1. We define a new class of *global* Wasserstein Autoencoder models, and implement an Optimal Transport-based incarnation we call the Global Sinkhorn Autoencoder.

2. We implement several metrics for evaluating such models, both in the unsupervised setting, and in a semi-supervised setting, which are the following: the global OT loss, which measures the OT loss on the full test dataset; the reconstruction error on the full test dataset; a so-called covered area which measures how well the latent points are matched; and two types of clustering measures.

3. We demonstrate on specific complex prior distributions that *global* optimal transport improves the performance of generative models compared to minibatch-based baselines when evaluated by the previously listed metrics.

Key words and phrases: Optimal Transport, Wasserstein distance, Sinkhorn Autoencoder, Generative models.

2010 Mathematics Subject Classification: 49Q22, 62M45, 68T07.

This work was supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.3-VEKOP-16-2017-00002), the Hungarian National Excellence Grant 2018-1.2.1-NKP-00008 and by the Ministry of Innovation and Technology NRDI Office within the framework of the Artificial Intelligence National Laboratory (RRF-2.3.1-21-2022-00004). M.F. Kiss and A. Csiszárik was partly supported by the project TKP2021-NKTA-62 financed by the National Research, Development and Innovation Fund of the Ministry for Innovation and Technology, Hungary.

This work is a detailed version of a MaCS 2020 presentation.

1. Introduction

Our objects of interest in this paper are generative autoencoders, that is, autoencoders where the decoder can be used as a generator when samples from some simple prior are fed to it. The beautiful Theorem 2.1 of [23], generalizing Theorem 1 of [29], gives a powerful and general guideline in designing generative autoencoders. Informally, it states that assuming a Lipschitz generator, the Wasserstein distance between the true data distribution and the generated distribution is equal to the Wasserstein distance between the encoded true data distribution and the latent prior, plus the reconstruction error.

In Tolstikhin's original, slightly less general formulation in [29], the pushforward of the true data distribution is assumed to be equal to the prior, and it is deduced that the Wasserstein distance between the true data distribution and the generated distribution is equal to the reconstruction error. Of course the assumption of the theorem is of a "spherical cow" kind: it is impossible to match the pushforward of the true data distribution exactly. But the advantage of this formulation, exploited by the Wasserstein Autoencoder class of algorithms, is that we can apply Lagrangian relaxation, adding an extra penalty term for matching the aggregate posterior to the prior, not necessarily in Wasserstein distance, but in any well-chosen divergence. This penalty term can be seen as a statistical test verifying that the pushforward is indeed close to the prior.

Very often a further simplification is made: the prior is represented by a sample, and the inputs of the statistical test are the two sets of samples. This is the form that the Adversarial Autoencoder [22] and the Sinkhorn Autoencoder [23] take. The obvious disadvantage of this approach is that it introduces a further sampling error and potential bias in the gradients, when compared to treating the prior analytically (which is called the semi-discrete case in the field of Optimal Transport). One advantage is that it can deal with complex artificially constructed priors.

In all incarnations of this idea that we are aware of, such as Adversarial Autoencoders [22], Maximum Mean Discrepancy (MMD) nets [9], sliced Wasserstein Autoencoders [32], or Sinkhorn Autoencoders [23], the input for this statistical test is the latent image of the minibatch (latent minibatch for short). However, as it is hinted by experiments of [26], the size of the minibatch may strongly affect the performance of the WAE. In our work, we go further and argue that in the WAE class of models, the minibatch size is not large enough compared to the latent dimension, which means that the statistical test will be too weak to guarantee a good match between the pushforward and the prior. (The typical minibatch size in WAE-class models is in the range of 50-200, although some MMD-based generative models moved it into the 1000s range exactly with the goal of strengthening the statistical power, as in [20]).

It is not very surprising that a sample size that is approximately in the same range as the data dimension can lead to a too weak statistical test: the field of sampling is a natural habitat of the curse of dimensionality (see [8]). But this intuition leads us toward an uncomfortable position: maybe even the latent image of the full dataset might not be large enough to statistically verify the closeness of the pushforward and the prior.

In this paper, we present tasks where a *global* model optimizing the regularization loss term on the full dataset offers improvements over minibatch-based loss, according to various evaluation metrics.

2. Theoretical framework for Wasserstein distance

Following [29], the sample spaces are denoted by $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$, while X, Y, Z and P_X, P_Y, P_Z denote the corresponding random variables and distributions. Now let $\Pi(P_X, P_Y)$ be the set of all joint distributions with marginals P_X, P_Y, that is, the set of couplings from P_X to P_Y. For a measurable, non-negative cost function $c : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^+ \cup \{\infty\}$ the optimal transport cost between distributions P_X and P_Y is defined by

$$W_c(P_X, P_Y) = \inf_{\Gamma \in \Pi(P_X, P_Y)} \mathbb{E}_{(X,Y)\sim\Gamma}[c(X, Y)]. \tag{2.1}$$

If $c(x, y) = d(x, y)^p$ for a metric d and $p \geq 1$, then $W_p := \sqrt[p]{W_c}$ is called the p-Wasserstein distance. (Also known as the Kantorovich–Rubinstein distance, or in the case $p = 1$, earth mover's distance.) In this special case, this concept provides a way to measure distance between probability distributions such that it reflects the geometry of their domain.

To use this machinery as the basis of a generative model, denote by P_X the true data distribution on $\mathcal{X}$. To define the latent variable model, one can fix a latent space $\mathcal{Z}$ with a prior distribution P_Z on $\mathcal{Z}$, and consider the conditional distribution $G(X|Z)$ parametrized by a neural network G. The generative model is specified as $G(X|Z)P_Z$, while the induced marginal is P_G. The aim of the neural network is to learn P_G such that it approximates P_X, that is

$$\min_G W_c(P_X, P_G). \tag{2.2}$$

Theorem 2.1 of [23] states the following:

Theorem 2.1. *Let $\mathcal{X}$, $\mathcal{Z}$ be endowed with any metrics and $p \geq 1$. Let P_X be a non-atomic distribution and $G(X|Z)$ be a deterministic generator/decoder that*

is λ-Lipschitz. Then we have the equality:

$$W_p(P_X, P_G) = \inf_{Q \in \mathcal{F}} \sqrt[p]{\mathbb{E}_{X \sim P_X} \mathbb{E}_{Z \sim Q(Z|X)} \left[d(X, G(Z))^p \right]} + \\ + \lambda \cdot W_p(Q_Z, P_Z)\,, \tag{2.3}$$

where $\mathcal{F}$ is any class of probabilistic encoders that at least contains a class of universal approximators.

If $\mathcal{X}$, $\mathcal{Z}$ are Euclidean spaces endowed with the L_p-norms $\|\cdot\|_p$ then a valid minimal choice for $\mathcal{F}$ is the class of all deterministic neural network encoders Q (here written as a function), for which the objective reduces to:

$$W_p(P_X, P_G) = \inf_{Q \in \mathcal{F}} \sqrt[p]{\mathbb{E}_{X \sim P_X} \left[\|X - G(Q(X))\|_p^p \right]} + \\ + \lambda \cdot W_p(Q_Z, P_Z).$$

This theorem leads to the following unconstrained min-min-optimization objective over deterministic decoder and encoder neural networks:

$$\min_G \min_Q \sqrt[p]{\mathbb{E}_{X \sim P_X} \left[\|X - G(Q(X))\|_p^p \right]} + \tag{2.4}$$

$$+ \lambda \cdot W_p(Q_Z, P_Z), \tag{2.5}$$

with $\lambda \geq \|G\|_{\text{Lip}}$ for all occuring G.

3. Entropy regularization and Sinkhorn autoencoders

We follow [23] to summarize the direct influences of our work in the field of Sinkhorn autoencoders. By the previous section, using p-Wasserstein distance to the prior to train a generative model is theoretically established. However, even estimating the Wasserstein distance is incredibly difficult computationally. Algorithms based on traditional combinatorial results as the Hungarian method [17] have unacceptable time and sample complexity. This obstacle is overcome via entropy regularizaton ([7], [13], [12] and [10]). Notably, for nonnegative ε, the entropy regularized OT cost is defined as

$$\tilde{S}_{c,\varepsilon}(P_X, P_Y) := \inf_{\Gamma \in \Pi(P_X, P_Y)} \mathbb{E}_{(X,Y) \sim \Gamma}[c(X, Y)] + \\ + \varepsilon \cdot \mathrm{KL}(\Gamma, P_X \otimes P_Y), \tag{3.1}$$

(where KL is the Kullback-Leibler divergence) which can be made a divergence by removing the entropic bias:

$$S_{c,\varepsilon}(P_X,P_Y) := \tilde{S}_{c,\varepsilon}(P_X,P_Y) - \\ - \frac{1}{2}\left(\tilde{S}_{c,\varepsilon}(P_X,P_X) + \tilde{S}_{c,\varepsilon}(P_Y,P_Y)\right). \tag{3.2}$$

We call $S_{c,\varepsilon,L}$ the approximate Sinkhorn divergence, which can be computed in $O \sim M^2 \cdot L$ time, where L is the number if Sinkhorn iterations ([7], [1]). Moreover, as $\varepsilon \to 0$, $S_{c,\varepsilon}$ converges to $W_c(P_X,P_Y)$, while as $\varepsilon \to \infty$, it converges to $\mathrm{MMD}_{-c}(P_X,P_Y)$, which has favorable sample complexity compared to $W_c(P_X,P_Y)$. More explicitly, the sample complexity of $\mathrm{MMD}_{-c}(P_X,P_Y)$ is independent of the dimension, and scales as $\frac{1}{\sqrt{M}}$ ([15]). Consequently, by a clever balancing of ε we can get close to our original objective while maintaining preferable computational, and statistical properties.

If we replace Wasserstein distance by Sinkhorn divergence in (2.4), we arrive at the Sinkhorn AutoEncoder (SAE) objective introduced by [23]:

$$\min_G \min_Q \sqrt[p]{\mathbb{E}_{X\sim P_X}\left[\|X - G(Q(X))\|_p^p\right]} + \\ + \lambda \cdot S_{\|\cdot\|_p^p,\varepsilon,L}(Q_Z,P_Z),$$

supposed to approximate our original objective.

4. Our method

Core idea: utilize the global scope. To optimize the Wasserstein distance (specifically, the Wasserstein-2 distance) between the aggregate posterior and the prior, our models employ optimal transport, specifically, it utilizes the unbiased Sinkhorn divergence as a loss function, similarly to the Sinkhorn Autoencoder models of [23]. The crucial difference that distinguishes our models from Sinkhorn Autoencoders is that our model we call **Global-SAE** optimizes the transport between the latent image of the full dataset and a similarly-sized sample from the prior, as opposed to working on the minibatch level.

Detailed description of the Global-SAE algorithm. Our main proposal for new methods capable to work with point cloud sizes strongly exceeding the ones encountered in the minibatch regime is summarized in Algorithm 1, in which we proceed as follows. In lines 2-3 we resample the target latent point cloud, and calculate the latent images $\hat{z}$ of the dataset x. Note, that instead of considering data and target samples of size M of a minibatch, we examine the complete dataset $x = \{x_i\}_{i=1}^n$ and a correspondingly sized target set $z = \{z_i\}_{i=1}^n$ sampled

from the prior distribution P_Z, thus, we operate in the global scope of the dataset. In lines 4-6 we calculate the reconstruction loss for a minibatch. (For this regular autoencoder loss term we remain with the minibatch scope.) In line 7 we calculate the error terms resulting from the optimal transport cost between $\hat{z}$ and z. Then in line 8 we update the model parameters by taking a gradient step with the above global optimal transport loss function and also with the reconstruction error.

Algorithm 1 Global Sinkhorn Autoencoder (Global-SAE)

Input: Dataset $x = \{x_i\}_{i=1}^n$, prior distribution P_Z,
encoder weights Ψ, decoder weights Φ, learning rate μ, traning iters $iter$, minibatch size M
parameters for the Sinkhorn algorithm: $\varepsilon \in \mathbb{R}$, $L \in \mathbb{N}$, $c = \|.\|_2^2$, sinkhorn weight λ

Output: Trained model with encoder weights Ψ, decoder weights Φ

1: **for** $i = 1..iters$ **do**
2: $z = \{z_i\}_{i=1}^n \sim P_Z$ {Sample target point set from P_Z — global data}
3: $\hat{z} = \{\hat{z}_i\}_{i=1}^n \leftarrow Q_\Psi(x)$ {Encoded image of the entire dataset x — global data}
4: $\widetilde{x} = \{x_i\}_{i \in I}$ where $I \subseteq \{1, 2, ..., n\}$ random subset, $|I| = M$ {Minibatch sample from x}
5: $\widetilde{x}' \leftarrow G_\Phi(Q_\Psi(\widetilde{x}))$
6: $D \leftarrow \frac{1}{M}\|\widetilde{x} - \widetilde{x}'\|_2^2$ {Calculate reconstruction loss for minibatch}
7: $S = S_{c,\varepsilon,L}(\hat{z}, z)$ {Calculate Sinkhorn loss for global point clouds $\hat{z}$ and z — global data}
8: $(\Psi, \Phi) \leftarrow (\Psi, \Phi) - \mu \cdot \nabla_{(\Psi,\Phi)}(D + \lambda \cdot S)$ {Update model parameters}
9: **end for**

4.1. Algorithm variants with different tradeoffs

Taking calculations in the global scope can be costly. To improve the efficiency of our model for larger datasets, in this subsection we discuss possible modifications to Algorithm 1 with different type of speed vs. accuracy vs. memory footprint trade-offs. These approaches consider the three most expensive steps of Algorithm 1. Here we list these three expensive operations with a brief summary of the type of cost they introduce.

- Calculating the pushforward of the entire dataset (line 3 in Algorithm 1): Considering that the memory footprint of the latent embeddings is fairly low, the main cost here is the computational cost of the forward passes with the encoder model $Q_\Psi(x)$ for the entire dataset.

- Calculating the Sinkhorn divergence between large point clouds (line 7 in Algorithm 1): The iterative projections of the Sinkhorn algorithm has memory complexity $O(n)$, where n is the size of the point cloud. The main cost here thus is the computational cost of the Sinkhorn algorithm which is quadratic in the point cloud size n.

- Computing the gradients for the full latent point cloud (line 8 in Algorithm 1): considering the backward pass also (instead of just the forward pass in line 3) roughly only doubles the computational cost, however, to backpropagate the gradients through the encoder, we must either store all the activations for the entire point cloud (which we consider costly in terms of memory) or introduce another doubling factor of necessary computations (by making the global backward pass also with minibatches).

Frequency of global point cloud updates. One possibility is to update the push-forward less frequently. For this we introduce a hyperparameter *recalculate frequency*, which controls how often do we encode the entire dataset (if this hyperparameter is greater than 1, then we use the last seen positions of the points, except for the current minibatch).

Calculate in the global scope, but backpropagate only on a minibatch. In the Global-SAE algorithm all latent positions are calculated by the encoder in each iteration, thus the gradient signal coming from the Sinkhorn loss can be back-propagated for all datapoints to the encoder weights. In contrast to this, we introduce a new variant: Minibatch-Global-SAE, which takes the latent positions from a "cache" for all points except for the current minibatch. This means that for this variant, the gradient signal coming from the Sinkhorn loss can only propatage to the encoder for the elements of the current minibatch, and the gradient signal is "thrown away" for all datapoints outside the minibatch. The only difference between the two algorithms is that in case of Minibatch-Global-SAE we take the gradient on the minibatch, while in Global-SAE we take the gradient on the entire dataset. The runtime and the memory requirements for the decoder forward-backward pass and the Sinkhorn loss calculation are identical between the two variants. The significant differences between the resource requirements of the two variants lie in the way encoder gradient updates are treated. The Full Global variant requires a forward-backward pass of the encoder on the full dataset for each minibatch calculation.

4.2. Global model variants

Our general approach can be applied to different WAE variants resulting in different global analogs. For example the Global-WAE-MMD abbreviation could stand for a variant with the MMD loss, which is a particularly meaningful choice. The necessary requirement is that the utilized WAE loss must

be computable for the full dataset, hence the WAE-GAN has no direct analog. Here we leave the exploration of the efficiency of these method for future work.

5. Experiments

In this section we present experiments that demonstrate the effectiveness of our method.

5.1. MNIST with a two dimensional Gaussian mixture prior

In this experiment we followed [22], by choosing the dataset to be the standard benchmark MNIST dataset [18], and the prior to be a mixture of 10 2D Gaussians, shaping a flower, see Figure 1. Specifically, our prior point cloud is a uniform sample from a Gaussian mixture with the following means: $\Big(3\cos(\alpha), 3\sin(\alpha)\Big)$, and variances: $\Big(\big(\cos^2(\alpha)+\frac{\sin^2(\alpha)}{100}, \cos(\alpha)\sin(\alpha)(1-1/100)\big), \big(\cos(\alpha)\sin(\alpha)(1-1/100), \sin^2(\alpha)+\frac{\cos^2(\alpha)}{100}\big)\Big)$, where $\alpha = \frac{2\pi\cdot i}{10}$ $i = 0, \ldots, 9$.

We conduct experiments with two neural network architectures: an MLP network with 3 hidden layers of 256 neurons in each layer and ReLU activations, both in the encoder and the decoder; and a modern incarnation of the LeNet-5 architecture [19] with maxpooling, ReLU activations, and transposed convolutions in the decoder.

We used the SamplesLoss function from the geomloss [11] library to compute the Sinkhorn loss. The geomloss library abstracts the underlying iterations of the Sinkhorn-Knopp algorithm and instead allows users to control the regularization and convergence behavior through the parameters blur and scaling. We set $blur = 0.05$ and $scaling = 0.7$ in our experiments.

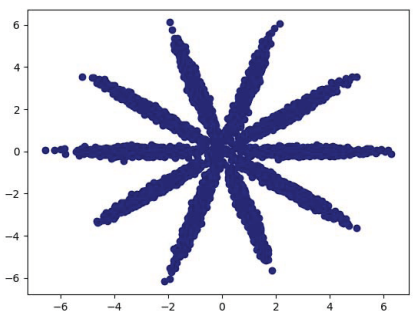


Figure 1: The "flower" prior. A mixture of 10 Gaussians in 2 dimensions.

5.1.1. Evaluation metrics

To obtain a detailed view on the quality of the models, we examine five different quantitative evaluation metrics, each of which shed light from a different viewpoint on the quality of the learned model. Besides reporting the standard *test reconstruction loss*, and the Sinkhorn loss on the entire test set (which we call *global OT loss* for brevity), we also introduce three metrics that measure the quality of the latent space formed by the trained model.

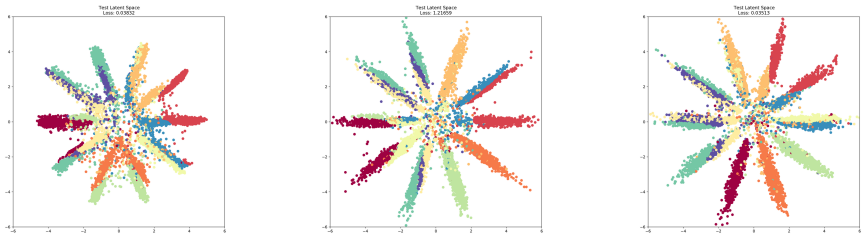
Local clustering To measure how well the same labeled points coalesce we utilize the *local clustering* metric which is a standard evaluation metric in semisupervised models. Here, we call an encoded point good, if the 10 nearest encoded points to it in the latent space have the same label as this point. The ratio of the good points in the test set is what we call *local clustering*.

Cluster matching. One might wish that working with such a prior (the "flower" prior), as there are 10 labels, all encoded images with the same label should belong to the same petal. We are interested in how much the actual embedding approximates this ideal. This consideration leads to a metric we call *cluster matching*. Informally, the metric is the classification performance of the clustering algorithm assigning the points to petals by maximum likelihood, assuming that the unknown assignment between clusters and labels is chosen optimally. First, we split the plane by 5 lines passing through the origin, such that each angle between adjacent lines is the same, and for each angular domain, the angular bisector passes through the mean of one of the Gaussian distribution in the mixture. We partition the plane into 10 domains, and we can assign each encoded point to the petal which has its mean in that specific domain. Then we create a complete bipartite graph where the two sets of nodes represent the labels and the petals respectively, and the weight of the edge between the ith petal and jth label is equal to the number of test points assigned to the ith petal and has label j. The value *cluster matching* is the weight of the maximal weight perfect matching divided by the size of the test set.

Covered area. As the latent space is two dimensional, we can check visually how well the models are able to match the encoded points to the prior point cloud. In addition we worked out a measure which we call *covered area* which checks how well the encoded points are matched to the prior distribution.

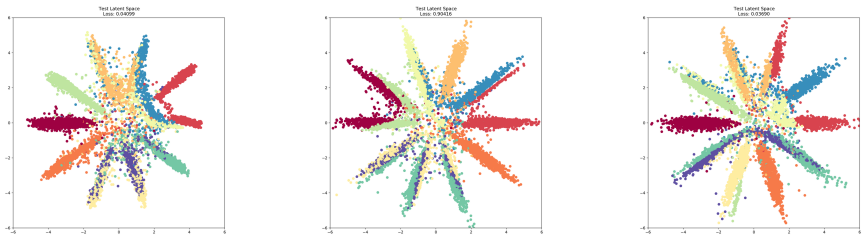
For Gaussian priors, let T be a transformation that transforms the prior to the uniform distribution on the unit square in the plane. We transform each encoded point by T, thus each encoded point is transformed to the unit square. We create a very dense grid on the square, look at the small neighborhood of the transformed encoded points and compute how many little squares in the grid intersect with the neighborhoods. The ratio of such squares to the number of squares in the grid is what we call *covered area*.

For Gaussian mixtures, we assign each encoded point to one of the mixture components by maximal likelihood. Assuming large divergence between our mixture components, every such "petal" is a Gaussian distribution with good approximation, and we can calculate the covered area for each of the petals separately, then take the average of these values.



(a) Local (b) Minibatch global (c) Full global

Figure 2: The encoded test set in the latent space, using the MLP net. "Local" refers to the baseline SAE method. The points match the prior better and they have a more orderly arrangement in the global versions.



(a) Local (b) Minibatch global (c) Full global

Figure 3: The encoded test set in the latent space, using the LeNet-5 architecture. "Local" refers to the baseline SAE method. The points match the prior better and they have a more orderly arrangement in the global versions.

5.1.2. Results

Hyperparameter selection. We did extensive grid search for the λ parameter in all three models, altogether in the set $\{0.001, 0.01, 0.1, 1, 10, 100, 1000, 10000\}$. The metric which we primarily took into consideration in choosing the best lambda was the global OT loss, and secondarily the cluster matching. From this we found that the best λ value for the baseline and full global models was 0.1 and it was 100 for the minibatch global model. To further validate these choices, as the latent space is two dimensional, we can track visually how well the models are able to match the prior point cloud. We found that the visually chosen λ parameter concides with the λ parameter chosen according to the numerical results.

We report the metrics introduced in the previous section with each of the three models. The results can be seen in Tables 1 and 2. For the MLP net the experiments ran for 50 epochs, and 30 epochs for LeNet-5. The highlighted number in each row is the best result in that metric. We also present generated images (Figures 4 and 5) for both nets.

MLP. In the case of the MLP net, the differences between the Sinkhorn loss and test reconstruction were small, although both global models performed better than the baseline model. The significant differences came in the other three metrics, where again both global models performed better, with an almost 0.1 difference in favour of the global models for the cluster matching and 0.05 for the local clustering.

LeNet-5. For LeNet-5, in cluster matching all three models performed better than for the MLP. There was a significant jump in the performance of the baseline model in this measure, although one of the global models was still better. As for the local clustering, the baseline model offered the same result, but the global models performed significantly better.

MLP	Local	Minibatch global	Full global
Sinkhorn loss ↓	0.018 ±0.28e−2	**0.013** ±0.33e−2	0.014 ±0.38e−2
Reconstruction ↓	0.036 ±0.19e−3	0.034 ±0.33e−3	**0.034** ±0.1e−3
Local clustering ↑	0.456 ±0.012	0.505 ±0.019	**0.516** ±0.017
Cluster matching ↑	0.55 ±0.067	0.64 ±0.033	**0.64** ±0.01
Covered area ↑	0.83 ±0.82e−2	**0.89** ±0.007	0.87 ±0.008

Table 1: Results for the MLP net after 50 epochs. Averages of 5 runs with different random seeds. An arrow indicates if lower (↓) or higher (↑) is better.

LeNet-5	Local	Minibatch global	Full global
Sinkhorn loss ↓	0.016 ±0.21e−2	**0.011** ±0.13e−2	0.015 ±0.13e−2
Reconstruction ↓	0.0362 ±0.12e−3	0.0360 ±0.16e−3	**0.0356** ±0.4e−3
Local clustering ↑	0.5339 ±0.019	0.5305 ±0.033	**0.5752** ±0.031
Cluster matching ↑	0.6374 ±0.018	**0.6406** ±0.073	0.6317 ±0.048
Covered area ↑	0.862 ±0.85e−2	**0.893** ±0.007	0.862 ±0.009

Table 2: Results for the LeNet net after 30 epochs. Averages of 5 runs with different random seeds. An arrow indicates if lower (↓) or higher (↑) is better.

6. Conclusion

We introduced a new algorithm in the Wasserstein Autoencoder class of algorithms, a global version of the Sinkhorn Autoencoder. For a Global Wasser-

(a) Local (b) Minibatch global (c) Full global

Figure 4: Generated images using the MLP net.

(a) Local (b) Minibatch global (c) Full global

Figure 5: Generated images using LeNet-5.

stein Autoencoder, the optimization of the latent point positions takes place over the full training dataset rather than over a minibatch.

We presented two variants of this global model, a full global variant, where the gradient signal coming from the global regularization loss acts on all the latent points, and a minibatch global variant, where it only acts on the current minibatch. As our experiments demonstrate, our global models consistently improve on the local baselines for complex priors in low latent dimensions. The minibatch global model's performance was consistently on par with the full global model's performance, despite the less robust theoretical guarantees. This makes the minibatch global model a promising alternative to the full global model, thanks to its significantly lower memory- and runtime complexity.

References

[1] **Altschuler, J., J. Weed P. Rigollet,** Near-linear time approximation algorithms for optimal transport via Sinkhorn iteration, In *NIPS* (2017).

[2] **Bellemare, G., I. Danihelka, W. Dabney, S. Mohamed, B. Lakshminarayanan, S. Hoyer and R. Munos,** The cramer distance as a solution to biased Wasserstein gradients, arXiv preprint (2017) https://arxiv.org/pdf/1705.10743

[3] **Bojanowski, P. and A. Joulin,** Unsupervised learning by predicting noise, In *ICML* (2017).

[4] **Bousquet, O., S. Gelly, I. Tolstikhin, C.-J. Simon-Gabriel and B. Schoelkopf,** From optimal transport to generative modeling: the VEGAN cookbook, arXiv preprint (2017) https://arxiv.org/pdf/1705.07642

[5] **Chizat, L., G. Peyre, B. Schmitzer and F.-X. Vialard,** Scaling algorithms for unbalanced transport problems, arXiv preprint (2016) https://arxiv.org/pdf/1607.05816

[6] **Cominetti, R. and J. San Martín,** Asymptotic analysis of the exponential penalty trajectory in linear programming, *Math. Programming*, **67(2, Ser. A)** (1994), 169–187.

[7] **Cuturi, M.,** Sinkhorn distances: Lightspeed computation of optimal transport, In *NIPS* (2013).

[8] **Donoho, D.,** High-dimensional data analysis: The curses and blessings of dimensionality, AMS Math Challenges Lecture (2000).

[9] **Dziugaite, G.K., D.M. Roy and Z. Ghahramani,** Training generative neural networks via Maximum Mean Discrepancy optimization, In *AUAI* (2015).

[10] **Feydy, J., T. Séjourné, F.-X. Vialard, S. Amari, A. Trouvé and G. Peyré,** Interpolating between optimal transport and MMD using Sinkhorn divergences, arXiv preprint (2018) https://arxiv.org/pdf/1810.08278

[11] **Feydy, J., T. Séjourné, F.-X. Vialard, S. Amari, A. Trouve and G. Peyré,** Interpolating between optimal transport and MMD using Sinkhorn divergences, In: *22nd International Conference on Artificial Intelligence and Statistics*, (2019), 2681–2690.

[12] **Genevay, A., L. Chizat, F. Bach, M. Cuturi, G. Peyré and others,** Sample complexity of Sinkhorn divergences, In *AISTATS* (2019).

[13] **Genevay, A., G. Peyré, M. Cuturi and others,** Learning generative models with Sinkhorn divergences, In *AISTATS* (2019).

[14] **Goodfellow, I.J., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville Y. Bengio,** Generative adversarial networks, In *NIPS* (2014).

[15] **Gretton, A., K.M. Borgwardt, M.J. Rasch, B. Schölkopf and A. Smola,** A kernel two-sample test, *Journal of Machine Learning Research*, **13(Mar)** (2018), 723–773.

[16] **Heusel, M., H. Ramsauer, T. Unterthiner B. Nessler,** GANs trained by a two time-scale update rule converge to a local nash equilibrium, In *NIPS* (2017).

[17] **Kuhn, H.W.,** The Hungarian method for the assignment problem, *Naval Research Logistics Quarterly*, **2(1-2)** (1955), 83–97.

[18] **LeCun, Y., C. Cortes and C.J. Burges,** MNIST handwritten digit database, *ATT Labs [Online]* (2010)
http://yann.lecun.com/exdb/mnist/

[19] **LeCun, Y., L. Bottou, Y. Bengio and P. Haffner,** Gradient-based learning applied to document recognition, *Proceedings of the IEEE*, **86(11)** (1998), 2278–2324.

[20] **Li, Y., K. Swersky and R. Zemel,** Generative moment matching networks, arXiv preprint (2015)
https://arxiv.org/pdf/1502.02761

[21] **Luise, G., A. Rudi, M. Pontil and C. Ciliberto,** Differential properties of Sinkhorn approximation for learning with Wasserstein distance, In *NIPS* (2018).

[22] **Makhzani, A., J. Shlens, N. Jaitly and I. Goodfellow,** Adversarial autoencoders, In *ICLR* (2016).

[23] **Patrini, G., M. Carioni, P. Forre, S. Bhargav, M. Welling, R. den Berg, T. Genewein and F. Nielsen,** Sinkhorn autoencoders, arXiv preprint (2018)
https://arxiv.org/pdf/1810.01118

[24] **Peyré, G., and M. Cuturi,** Computational optimal transport, arXiv preprint (2018)
https://arxiv.org/pdf/1803.00567

[25] **Rubenstein, P.K., B. Schoelkopf and I. Tolstikhin,** Wasserstein auto-encoders: Latent dimensionality and random encoders, In *ICLR* workshop (2018).

[26] **Rubenstein, P.K., B. Schoelkopf and I. Tolstikhin,** On the latent space of Wasserstein auto-encoders, arXiv preprint (2018)
https://arxiv.org/pdf/1802.03761

[27] **Schmitzer, B.,** Stabilized sparse scaling algorithms for entropy regularized transport problems, *SIAM Journal on Scientific Computing*, **41(3)** (2016).

[28] **Sinkhorn, R.,** A relationship between arbitrary positive matrices and doubly stochastic matrices, *Ann. Math. Statist.*, **35** (1964).

[29] **Tolstikhin, I., O. Bousquet, S. Gelly and B. Schoelkopf,** Wasserstein auto-encoders, In *ICLR* (2018).

[30] **Villani, C.,** Optimal transport: Old and new, *Grundlehren der mathematischen Wissenschaften*, Springer Berlin Heidelberg (2008).

[31] **Weed, J.,** An explicit analysis of the entropic penalty in linear programming, arXiv preprint (2018)
https://arxiv.org/pdf/1806.01879

[32] **Wu, J., Z. Huang, D. Acharya, W. Li, J. Thoma, D.P. Paudel and L. Van Gool,** Sliced Wasserstein generative models, In *CVPR* (2019).

A. Csiszárik, M.F. Kiss and B. Maga
ELTE Eötvös Loránd University, Faculty of Science
Alfréd Rényi Institute of Mathematics
Budapest
Hungary
csadrian@renyi.hu
mfkiss@renyi.hu
mbalazs0701@gmail.com

Á. Matszangosz and D. Varga
Alfréd Rényi Institute of Mathematics
Budapest
Hungary
matszangosz.akos@renyi.hu
varga.daniel@renyi.hu