Parallel Optimal Transport GAN*

Gil Avraham† Yan Zuo†
Tom Drummond
ARC Centre of Excellence for Robotic Vision, Monash University, Australia
{gil.avraham, yan.zuo, tom.drummond}@monash.edu

Abstract

Although Generative Adversarial Networks (GANs) are known for their sharp realism in image generation, they often fail to estimate areas of the data density. This leads to low modal diversity and at times distorted generated samples. These problems essentially arise from poor estimation of the distance metric responsible for training these networks. To address these issues, we introduce an additional regularisation term which performs optimal transport in parallel within a low dimensional representation space. We demonstrate that operating in a low dimension representation of the data distribution benefits from convergence rate gains in estimating the Wasserstein distance, resulting in more stable GAN training. We empirically show that our regulariser achieves a stabilising effect which leads to higher quality of generated samples and increased mode coverage of the given data distribution. Our method achieves significant improvements on the CIFAR-10, Oxford Flowers and CUB Birds datasets over several GAN baselines both qualitatively and quantitatively.

1. Introduction

Optimal transport theory has found widespread applications in numerous fields, including various applications in statistics and machine learning domains. Despite some difficulties accompanying the optimal transport approach, it offers a solution that is both intuitive and numerically well-behaved. This makes it a compelling approach for large scale problems which are unstable in nature; of note, optimal transport has been recently employed in the domain of Generative Adversarial Networks (GANs) [12] with great success.

Generative Adversarial Networks are generative models where a density function is not explicitly approximated but a sampling mechanism is provided. A GAN framework typically has two components: a Discriminator for estimating a family of f-divergences [26] between a model distribution P_g and the data distribution P_r, and a Generator that provides a mapping between a known noise distribution P_z to the model distribution P_g. The framework is optimised through a minimax two-step procedure where each network is responsible for training its counterpart. In practice, the family of f-divergences suffers from numerical instabilities and brittle parameter selection, giving rise to many regularisation methods [22, 23, 28]; ultimately leading to training instability issues such as vanishing gradients and mode collapse [1, 6, 21, 29].

Optimal transport has played a key role in alleviating these instability issues in GANs [2, 30]. Most notably, [2] proposed an alternative to the value function being optimised by GANs. This approach utilised ideas from transportation theory, leveraging the Wasserstein distance to produce a value function which was shown to be less fragile and better fitted as a GAN value function when using gradient-based optimisation methods. Whilst use of the Wasserstein distance was shown to help stabilise training of GANs, its convergence rate was still directly dependent on the dimensionality of the problem [8, 11], making it a less reliable distance when measuring high dimensional data [5, 31] (as is often the case in images). Additionally, the current solution exists in the dual formulation, which imposes the constraint that the Discriminator must lie within the space of 1-Lipschitz functions. Enforcement of this constraint requires regularising divergence or distance used for GAN training which is imposed via gradient penalty [13].

We propose the Parallel Optimal Transport GAN (POT-GAN), an unsupervised approach which regularises the Generator using the Wasserstein distance in a low dimensional latent representation of the data distribution and model distribution P_r, P_g respectively. We show that significant convergence gains are achieved by reducing the dimensionality of the problem when estimating the Wasserstein distance. This allows for the direct computation of the primal form of the optimal transportation problem, avoiding

*This work was supported by the Australian Research Council Centre of Excellence for Robotic Vision (project number CE1401000016).

†Authors contributed equally

978-1-7281-3293-8/19/$31.00 ©2019 IEEE
DOI 10.1109/CVPR.2019.00454

the need to carefully maintain the 1-Lipschitz constraint accompanying the dual form. The latent representation is obtained using a Variational Autoencoder (VAE) [18] and for estimating the Wasserstein distance, we use the Monge formulation [34] where a transport map and a cost function are estimated throughout the optimisation process. This procedure provides better initial estimates of the Wasserstein distance at every iteration leading to more stable training, where continuous matching of intrinsic representations will result in increased mode coverage. We experimentally validate our method and show significant improvements on Inception [29] and FID [15] scores.

The main contributions of this paper are the following:

1. We introduce a novel GAN framework which uses an optimal transport regulariser on the latent representation of the data distribution, aiding the Generator network towards a more stable convergence and increased modal diversity.

2. We demonstrate that both a transportation map and cost function can be effectively estimated within a low dimensional latent space. We employ a novel approach to perform this task by utilising the non-linear mapping capabilities of decision forests.

3. We show our method significantly improves on several GAN benchmarks on Inception and FID scores on CIFAR-10, Oxford Flowers and CUB Birds datasets.

2. Background

The goal of a Generative Adversarial Network (GAN) [12] is to train a Generator which, given an input sample $z \in \mathcal{R}^{d_z}$ drawn from some arbitrary distribution (usually standard Gaussian), is able to produce an output sample $G(z) \in \mathcal{R}^{d_r}$, belonging to some distribution P_g which closely approximates a target distribution P_r. A GAN achieves this by setting up a game between the Generator G and a Discriminator D (typically represented by neural networks), which can be described by the following minimax objective loss function:

$$\begin{aligned}\min_G \max_D V(D,G) = E_{x\sim P_r}[\log D(x)] \\ + E_{z\sim P_z}[\log(1 - D \circ G(z))]\end{aligned} \tag{1}$$

Proper optimisation over Eq. 1 occurs when both D and G each learn at the same rate and are evenly matched, ensuring that both networks continue to improve. [12] showed that training D to convergence estimates Jensen-Shannon (JS) distance between the two distributions, and this signal is provided towards improving G. Further works generalised this approach leading to f-divergences [26] and mutual information minimisation criteria [3].

2.1. GAN Regularisation

Since GANs lack an explicit specification of the density function, variational inference has been combined with GANs to leverage advantages from each method [16]. [20] introduced the VAEGAN which offered a trade-off between the sharpness offered by a GAN in exchange for the stability and diversity of a Variational Autoencoder (VAE). Subsequently, [4] added an additional adversarial loss which aimed to diffuse generated samples towards reconstructed real samples. Both VAEGAN and MDGAN are similar works in the sense that they attempt to add an additional manifold matching penalisation term. A drawback to both of these works is that the matching is being performed in the high-dimensional pixel space, which harms these two methods substantially. ALI by [9] and BiGAN [7] both jointly train a Generator and its inference while maintaining the adversarial loss which mitigated sample quality degradation. Most recently, VEEGAN [32] trained an auxiliary network for inverting the Generator output in a GAN to match the input noise distribution, using the mismatch to provide a training signal for mitigating mode collapse and improve modal diversity.

2.2. Optimal transport in GANs

Optimal transport [35] addresses the problem of matching distributions from a perspective of mass transportation. In this context, the Wasserstein distance is the measure between two distributions. For the set of all joint probability distributions $\Gamma(x_r, x_g)$, we have probability distributions P_r, P_g over $\mathcal{X} \subseteq \mathbb{R}^d$ and the cost function $c(x_r, x_g) : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}^+$. The transport plan $\gamma^* \in \Gamma(x_r, x_g)$ minimises the following:

$$W_c(P_r, P_g) = \inf_{\gamma \in \Gamma(P_r, P_g)} \mathbb{E}_{(x_r, x_g)\sim\gamma}[c(x_r, x_g)] \tag{2}$$

A study by [2] compared different distance metrics for measuring distribution distances and showed the advantages of using the Wasserstein distance over other probability distances including the Jensen-Shannon distance. To address the expensive nature of estimating Eq. 2 for distributions of high dimensions, [2] suggested using the Kantorovich-Rubinstein duality [35]:

$$W_1(P_r, P_g) = \sup_{f \in \mathbb{F}_{Lip}} \mathbb{E}_{x_r\sim P_r}[f(x_r)] - \mathbb{E}_{x_g\sim P_g}[f(x_g)] \tag{3}$$

With the special case of $(\mathcal{X}, c)$ as a metric space and $\mathbb{F}_{Lip}$ are bounded 1-Lipschitz functions. This form of computing the Wasserstein distance replaced the conventional way of estimating the JS distance in Eq. 1, resulting in a new GAN objective:

$$\begin{aligned}\min_G \max_D V(D,G) = \min_{\theta\in\Theta} \max_{\omega\in\Omega} \mathbb{E}_{x_r\sim P_r}[D_\omega(x_r)] \\ - \mathbb{E}_{z\sim P_z}[D_\omega \circ G_\theta(z)]\end{aligned} \tag{4}$$

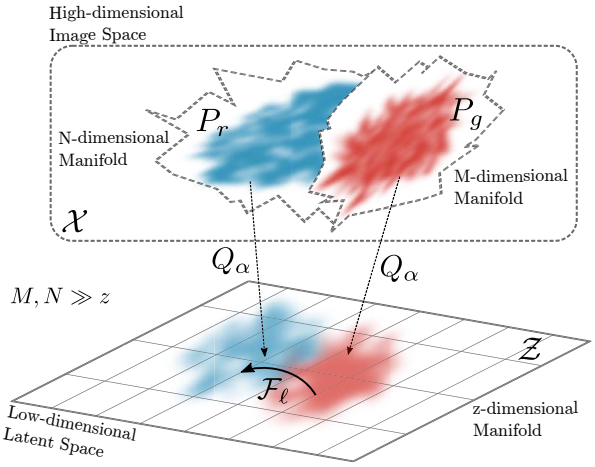


Figure 1: The data distribution P_r and model distribution P_g are both high dimensional and compact in space $\mathcal{X}$. Obtaining the optimal coupling between these distributions in practical settings with a finite sample size suffers from a weak convergence rate. Projecting both distributions (with mapping Q_α) onto a latent space $\mathcal{Z}$ offers an increased convergence rate and better estimate of the Wasserstein distance. Estimation of Wasserstein distance in this space encompasses learning a probability transformation $\mathcal{F}_\ell$ between the latent distributions, followed by computing the transport map (a permutation). The operation of matching the latent distribution serves as a guide for the overall goal of matching distributions P_r, P_g.

With G, D being neural networks parameterised by θ, ω and P_r, P_z being the data distribution and noise distribution respectively. In [2], weight clipping was used to maintain the 1-Lipschitz bound on D; this was recently extended upon by [13] which used gradient penalty instead to enforce the Lipschitz constraint. In this work we will show how the primal form in Eq. 2 computed explicitly on the latent representation of the data is used as a regulariser to guide the Generator component in Eq. 4.

3. On convergence rates of Wasserstein distance

The work of [12] showed that given the data distribution P_r and enough modelling capacity, the Generator in a GAN setup recovers a model distribution P_g matching the data distribution P_r. Complementing this finding, [24] provides strong evidence that P_r lies on a low dimensional manifold and [1] further rigorously proves that both P_r and P_g lie on low dimensional manifolds. Our GAN framework constructs an optimal transport regulariser on the latent representation which aims to help stabilise training, yielding a better estimation of the true distribution (refer to Fig. 1).

This is motivated from a convergence stand point of the Wasserstein distance and is discussed as follows.

We denote a probability distribution by P and the empirical distribution by $\hat{P}_n$. In the limit of large n, the Wasserstein distance of order k [35] approaches zero almost surely:

$$W_k(P, \hat{P}_n) \to 0 \quad a.s. \tag{5}$$

In practice, having access to a limited number of samples n drawn from P raises the question of how to quantify the rate at which $\hat{P}_n$ converges to P. Unfortunately, as shown in [8], the rate of convergence suffers from the *curse of dimensionality* [10]:

$$\mathbb{E}[W_1(P, \hat{P}_n)] \lesssim \left(\frac{1}{n}\right)^{\frac{1}{d}} \tag{6}$$

and in effect, for probability measures over $\mathcal{R}^d$ as the dimensionality of the space d grows larger, the representative power of $\hat{P}_n$ towards P shrinks, needing more samples to yield applicable convergence rates. Recently [37] generalised the original asymptotic bound by [8]. Here, we show a simplified version of [37].

Theorem 1. *Let $k \in [1, \infty)$. The convergence rate of the empirical distribution towards the $k-$order Wasserstein distance is given by:*

$$\mathbb{E}[W_k(P, \hat{P}_n)] \lesssim \left(\frac{1}{n}\right)^{\frac{1}{d}} \tag{7}$$

full proof given in [37].

We define the convergence rate product as Ω_b^a for $d_a \leq d_b$.

Proposition 1. *Let us define the distribution P_r with the random variable $X \in \mathcal{X} \subseteq \mathcal{R}^{d_r}$ and P_z as the distribution of its latent encoding with the latent random variable $Z \in \mathcal{Z} \subseteq \mathcal{R}^{d_z}$, with $\{\forall d_r, d_z \in \mathbb{Z}^+ : d_z \leq d_r\}$. Given the corresponding empirical distributions P'_r, P'_z along with their associated convergence rates then:*

$$\Omega_r^z \lesssim \left(\frac{1}{n}\right)^{\frac{d_r + d_z}{d_r d_z}} \tag{8}$$

where Ω_r^z is the convergence rate product for n number of samples.

Proof. Obtain convergence rates for empirical distributions P'_r, P'_z by applying **Theorem 1**. □

By inspecting Eq. 8, one can immediately observe that for $d_r \gg d_z$, the convergence product is dominated by the term d_z, *i.e.*:

$$\Omega_r^z \lesssim \left(\frac{1}{n}\right)^{\frac{1}{d_z}} \tag{9}$$

For probability distributions of high dimensional data such as images (*i.e.* when d_r is high), computing the Wasserstein distance in Eq. 2 is not feasible. Although the Kantorovich-Rubinstein duality can be used under the assumption of a compact metric space, maintaining the Lipschitz-1 constraint is non-trivial and does not scale well. The correct choice of latent representation size d_z in Eq. 9, in conjunction with a given dataset size can allow for quicker convergence and obtaining an estimate of the Wasserstein distance for the latent representation.

4. Parallel optimal transport in GANs

We define P_r, p_r, P_g and p_g to be the data and model probability distributions and densities respectively. The latent densities $z_r \sim p_r(z), z_g \sim p_g(z)$ are defined as:

$$p_r(z) = \int_{\mathcal{X}} p_r(z|x)dp_r(x), \quad p_g(z) = \int_{\mathcal{X}} p_g(z|x)dp_g(x) \tag{10}$$

For estimating the Wasserstein distance between latent distributions, we write out Eq. 2 and reformulate according to [34]:

$$W(P_r, P_g) = \inf_{\gamma \in \Gamma(P_r, P_g)} \mathbb{E}_{(z_r, z_g)\sim\gamma}[c(z_r, z_g)] = \inf_{\gamma \in \Gamma(P_r, P_g)} \int_{\mathcal{Z}\times\mathcal{Z}} c(z_r, z_g)d\gamma(p_r, p_g) \leq \inf_{\gamma_\sigma \in \Gamma_\sigma(P_r, P_g)} \int_{\mathcal{Z}} c(z_r, z_g)dp_r(z_r)\delta[z_g - \sigma(z_r)] \tag{11}$$

with the final inequality emerging from the Monge formulation of the optimal transport problem [34]. σ is the transport map and γ_σ is the transport plan where $\gamma_\sigma(z_r, z_g) = p_r\delta[z_g - \sigma(z_r)]$ is limited to distributions that a mass point cannot split (*i.e.* there must be a one-to-one mapping between P_r and P_g).

The bottom term in Eq. 11 recovers the Wasserstein distance under the conditions that the cost function in Eq. 2 is convex and distributions P_r, P_g are continuous [34]. Minimising Eq. 11 requires estimating the transport map σ by finding one-to-one corresponding pairs $\{z_{r,i}, z_{g,i}\}_{i=1,\dots,N}$ under a dynamically changing cost function.

The Discriminator value function is expressed using the WGAN-GP [13] form:

$$V_{D_\omega} = \mathbb{E}_{z\sim P_z}[D_\omega \circ G_\theta(z)] - \mathbb{E}_{x_r\sim P_r}[D_\omega(x_r)] + \lambda_{GP}\mathbb{E}_{\hat{x}\sim P_{\hat{x}}}[(\|\nabla_{\hat{x}} D_\omega(\hat{x})\|^2 - 1)^2] \tag{12}$$

we obtain the Generator value function by adding our regularisation term:

$$V_{G_\theta} = \mathbb{E}_{z\sim P_z}[-D_\omega \circ G_\theta(z)] + \lambda_{POT}\mathbb{E}_{(z_r, z_g)\sim P_g\delta[z_g - \sigma^*_\rho(z_r)]}[\left\|\sigma^*_\rho(z_r) - \mathcal{F}^*_\ell(z_g)\right\|^2] \tag{13}$$

with λ_{POT} being a hyperparameter specifying the weighting applied to the regularisation term. Permutation operator σ^*_ρ is the transportation map and $\mathcal{F}^*_\ell$ is a component of the cost function found by minimising:

$$\mathcal{F}^*_\ell, \sigma^*_\rho = \min_{\sigma_\rho \in \mathcal{S}_\rho, \ell} \Sigma_{i=1}^N \left\|\sigma_{\rho_i}(z_{r,i}) - \mathcal{F}_\ell(z_{g,i})\right\|^2 \tag{14}$$

where $\mathcal{S}_\rho$ is the set of all possible permutations. We note that $\mathcal{F}^*_\ell, \sigma^*_\rho$ need not be unique. The latent representations z_r, z_g are obtained by using a pre-trained fixed Variational Auto-Encoder [18] and applying the Encoder on $x_r \sim P_r$, $x_g \sim P_g$. The functions D and G are parameterised using neural networks with parameters ω and θ respectively. $\mathcal{F}$ is a learnable function parameterised by ℓ which is discussed in Section 4.2. Eq. 12 and Eq. 14 each evaluate the different metrics on their respective domains and are computed in parallel; where their results are then used to update the Generator in Eq. 13.

4.1. Latent Transform Map

The Wasserstein distance as formulated in Eq. 11 is the mean cost between a one-to-one correspondence of matching pairs after applying a probability transformation $\mathcal{F}$ (discussed in Section 4.2) on latent samples z_g. The lower dimensions of the latent representation allows a quick convergence to regularise Generator G_θ. The encoding model for inferring the latent representation is denoted by Q and is a neural network parameterised by α.

Estimating the cost function and the transportation map for each batch is described in Algorithm 1. This procedure finds the correspondences between the data and generated pairs such that a sample z_r, from the data distribution will be covered by a generated sample z_g, from the model distribution (*i.e.*$\{\forall z_{i,r} : \min_j c(z_{i,r}, z_{j,g})\}$). In the context of neural networks, this means constructing a differentiable operation that minimises the distances associated with corresponding pairs of real and generated samples. The transport map is a permutation over a learned cost function, we note that rearranging the data samples does not interfere with the gradient-descent operation for minimising the loss in Eq. 13. The cost function c is convex and also differentiable as discussed in detail in the following section.

4.2. Learning a cost function

The transformation map σ in Eq. 11 requires a one-to-one correspondence between latent samples z_r to z_g. Learning the cost function in the form of: $c(a, b) = \|a - \mathcal{F}_\ell(b)\|^2$ satisfies the convexity requirement and also estimates a probability transformation $\mathcal{F}_\ell$ to every state in which distribution P_g evolves throughout GAN training so that a transformation map can be found. Omitting $\mathcal{F}$ from the cost function in favour of a standard $L2$ cost function will re-

Algorithm 1 Obtain transport map σ_ρ and cost function transformation $\mathcal{F}_\ell$ in minibatch of size N

$\boldsymbol{\rho} \leftarrow$ initialise permutation container
$\boldsymbol{z}_r \in Q_\alpha(X_r) \leftarrow$ sample data batch
$\boldsymbol{z}_g \in Q_\alpha(X_g) \leftarrow$ sample generated batch
for $itr = 1$ to $iters$ **do**
  for $i = 1$ to N **do**
    $\rho_i \leftarrow \min_j \|(z_{i,r} - \mathcal{F}_\ell(z_{j,g})\|^2$
  end for
  $\boldsymbol{z}_r^{'} \leftarrow \text{Permute}\{\boldsymbol{z}_r, \rho\}$
  $\ell \leftarrow \text{Adam}(\nabla_\ell(\sum_{i=1}^{N} \left\|(z_{i,r}^{'} - \mathcal{F}_\ell(z_{j,g})\right\|^2), \beta_1, \beta_2)$
end for
return $(\boldsymbol{\rho}, \boldsymbol{\ell})$

sult in an inaccurate transformation map σ, leading to less accurate Wasserstein distance estimated by the regulariser.

Latent Transformation Forest For the choice of $\mathcal{F}$, [39, 40] showed that the non-linear discriminating power of decision forests were able to disentangle complex, jointly distributed data in the compact latent representation space of real images. In addition, [38] demonstrated that the use of decision forests offers a smooth manifold on the low dimension space of image distributions. Given these insights, we adopt the approach in [38] to employ a soft decision forest as a learnable component of the cost function c. Since the role of $\mathcal{F}$ is to operate as a probability transformation on the latent representation of our data, we refer to it as a Latent Transformation Forest (LTF). LTF is applied to the latent representation of samples from the generated distribution *i.e.*:

$$c(z_r, z_g) = \|z_r - \mathcal{F}_\ell(z_g)\|^2 \tag{15}$$

where ℓ are the parameters of $\mathcal{F}$, which hold the values stored in the leaf nodes of the forest.

Constructing the Forest For a given latent encoding of a generated sample, z_g, we can construct the internal decision nodes of the LTF through a reshape of z_g. The internal decision nodes utilise the values of the generated latent vector to determine routing portions to the terminal leaf nodes of the LTF. This is achieved by comparing the values of the generated latent vector z_g with a designated threshold value t_n and then passing this value through a sigmoid function (*i.e.* $\sigma(z_n - t_n)$) which converts them into a value between the range of $[0, 1]$. This outputs the routing portion to the left child or leaf node of the current decision node. The right portion is computed as $1 - \sigma(z_n - t_n)$ where t_n is a threshold value z_n is compared against. This is illustrated in Fig. 2.

The leaf nodes ℓ of the LTF hold values which represent the learned values of the transformed generated latent vector z_g' before Eq. 15 is applied. These values are blended according to weights dictated by the portions computed by the internal decision nodes (referring to Fig. 2, the portion allocated to the values held in leaf node ℓ_0 would be $p_0 \times p_1 \times p_3$). This represents a non-linear transformation on the generated samples' from its original representation to a transformed one where the $L2$ norm between latent distributions of the data and generated distributions are more evenly distributed (for additional details refer to supplementary material).

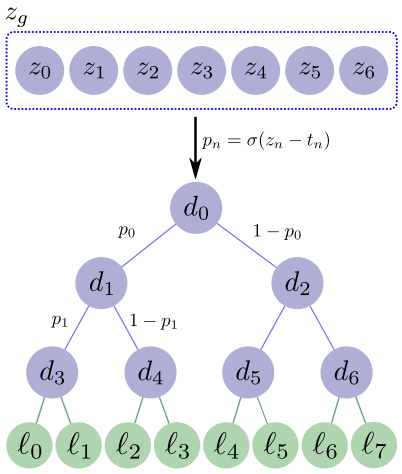


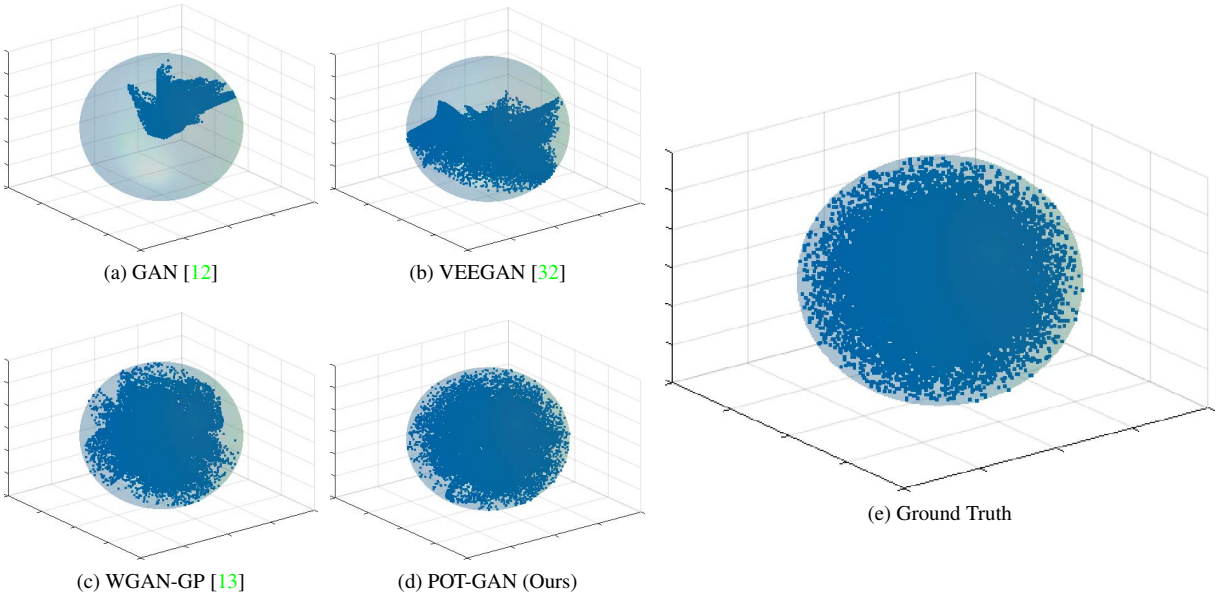
Figure 2: The latent vector of generated samples z_g is remapped as soft decision nodes in our LTF. The weighted sum of leaves of the LTF output the transformed generated latent vector z_g'.

Algorithm 2 A full outline of the training procedure for the POT-GAN model. For the complete formulation of $\mathcal{L}_{GAN}$ and the gradient penalisation term refer to [13]

$\boldsymbol{\alpha} \leftarrow$ Pre-train VAE, discard Decoder and retain Encoder (Q) parameters
$\boldsymbol{\theta}, \boldsymbol{\omega}, \boldsymbol{\ell}, \leftarrow$ initialise Generator (G), Discriminator (D), LTF $(\mathcal{F})$ parameters respectively
for $itr = 1$ to $iters$ **do**
  $\boldsymbol{X_r} \leftarrow$ random mini-batch from dataset
  $\boldsymbol{Z}_g \leftarrow$ sample noise $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$
  $\boldsymbol{X_g} \leftarrow$ generate data $G_\theta(\boldsymbol{Z}_g)$
  $(\boldsymbol{\rho}, \boldsymbol{\ell}) \leftarrow \boldsymbol{Algorithm1}(\boldsymbol{X_r}, \boldsymbol{X_g}, \boldsymbol{\ell})$
  $\mathcal{L}_{GAN} \leftarrow D_\omega \circ G_\theta(\boldsymbol{Z_g}) - D_\omega(\boldsymbol{X_r}) +$
    $\lambda_{GP} \mathbb{E}_{\hat{x} \sim P_{\hat{x}}}[(\|\nabla_{\hat{x}} D_\omega(\hat{x})\|^2 - 1)^2]$
  $\mathcal{L}_{POT} \leftarrow \|\boldsymbol{\sigma_\rho} \circ Q_{\boldsymbol{\alpha}}(\boldsymbol{X_r}) - \mathcal{F}_{\boldsymbol{\ell}} \circ Q_{\boldsymbol{\alpha}} \circ G_{\boldsymbol{\theta}}(\boldsymbol{Z_g})\|^2$
  $\boldsymbol{\omega} \overset{+}{\leftarrow} \text{Adam}(\nabla_{\boldsymbol{\omega}} \mathcal{L}_{GAN}, \beta_1, \beta_2)$
  $\boldsymbol{\theta} \overset{+}{\leftarrow} \text{Adam}(\nabla_{\boldsymbol{\theta}}(-\mathcal{L}_{GAN} + \lambda_{POT} * \mathcal{L}_{POT}), \beta_1, \beta_2)$
end for

Reverse-KL Divergence : $KL(P_g(x)\|\|P_r(x))$				
	GAN [12]	VEEGAN [32]	WGAN-GP [13]	POT-GAN (Ours)
8 Gaussian Ring (2D)	0.2417±0.0113	0.1540±0.0127	0.0046±0.0004	**0.0020±0.0002**
Gaussian Ball (3D)	3.0772±0.1014	2.5153±0.0708	0.9438±0.0044	**0.7428±0.0038**

Table 1: The reverse-KL divergence for learning Gaussian distributions. These values correlate with the visual results seen in Fig. 3.



(a) GAN [12] (b) VEEGAN [32] (e) Ground Truth (c) WGAN-GP [13] (d) POT-GAN (Ours)

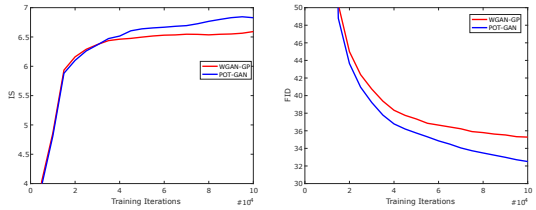
Figure 3: Results of learning a 3D Gaussian $x_r \sim \mathcal{N}_3(0, I)$ distribution from a 2D Gaussian input $z \sim \mathcal{N}_2(0, I)$. The 2-Dimensional latent space, cannot fully cover the 3-Dimensional data distribution. The transparent light blue sphere in Fig. 3a-3e has a radius of 3 standard deviations. The generated samples are the points inside the sphere. Points generated outside the sphere were clipped for visualisation purposes. In this case, we treat the problem as covering the single mode as best as possible. Aside from the quantitative results shown in Table 1, it is observable between Fig. 3a-3d, our method 3d covers the sphere in the most convincing manner in relation to ground truth.

5. Experiments

For qualitative and quantitative evaluations, we perform experiments on simulated data as well as three real datasets. Empirically, we found that concurrently training of the VAE and GAN does not lead to improved performance. As such, we pre-train and subsequently fix our VAE to reduce computation costs and reduce training time. In Algorithm 2, we detail the full procedure for training the POT-GAN model.

We compare POT-GAN to four GAN benchmarks, DCGAN [27], WGAN-GP [13], VEEGAN [32] and WAE-GAN [33]. To our knowledge, VEEGAN represents the most recent GAN framework which explicitly aims at stabilising GAN training by Generator regularisation. POT-GAN uses the WGAN-GP model in [13] as a baseline GAN. For the latent representation, we used a pretrained Variational Autoencoder [18], trained with settings specified in its respective paper (for more details on network architectures refer to the supplementary material).

Our POT-GAN model is trained with a batch size of 64, with network weights using the initialisation scheme detailed in [14]. Similar to [27], we use the ADAM optimiser [17], specifying a learning rate of 0.0002, first and second moment terms of 0.5 and 0.9 respectively, minimising the GAN loss and regularisation term defined in Section 4. For size of latent vector z, we use the commonly chosen 128 dimensions. For choice of λ_{POT}, we experimented with values of 1.0, 0.1 and 0.01. We found empirically that a value of $\lambda_{POT} = 0.01$ performed best and use this value for all our experiments.



(a) IS (b) FID

Figure 4: CIFAR-10 (a) Inception Score [29] and (b) FID Score [15] performance plots over Generator iterations. Both WGAN-GP [13] and POT-GAN were trained up to 100K iterations. The regularisation component in POT-GAN restricts the optimisation space in a way that leads to significant improvement on both measures that correlate well with quality and mode diversity of the model distribution.

5.1. Learning Gaussians

2D Gaussian Mixture Model We train a GAN [12], VEEGAN [32] and WGAN-GP [13] to learn a 2-Dimensional 8 Gaussian mixture model. We compute the reverse-KL divergence, $KL(P_g||P_r)$, using a Parzen window and the results of 50 runs are shown in Table 1. As observed, our method significantly outperforms competing methods, offering more than a relative 50% improvement in reverse-KL Divergence compared to WGAN-GP.

3D Gaussian Ball Additionally, we set up a traditional GAN, VEEGAN and WGAN-GP to learn a 3D Gaussian ball ($x_r \sim \mathcal{N}_3(0, I)$) from a 2D Gaussian input ($z \sim \mathcal{N}_2(0, I)$). This example is meant to train a Generator G_θ to map between a space which does not have the capacity to fully model the true distribution. In Fig. 3, we plot the results. The traditional GAN and VEEGAN fail to cover large parts of the Gaussian ball, exhibiting partial mode coverage behaviour. WGAN-GP is able to cover the sphere better, but there are clearly noticeably large holes within the sphere, representing parts of the distribution not covered. In contrast, our method covers the true distribution significantly better than WGAN-GP and there are no noticeable dead-zones within the sphere. We compute the reverse-KL divergence similarly to the 8-Gaussian mixture model example and list the results of 50 runs in Table 1. These results correlate well with the visual results in Fig. 3. Our method outperforms GAN [12], VEEGAN [32] and WGAN-GP [13], under this example where the Generator is made to learn the best mapping it can for covering a single mode while lacking the latent space capacity to fully represent the true distribution.

5.2. Real Datasets

For real data, we experimented on three commonly used datasets for obtaining comparative benchmarks:

CIFAR-10 The CIFAR-10 dataset [19] is a dataset consisting of 50,000 32×32 training images and 10,000 32×32 testing images evenly distributed across 10 broad class categories.

Oxford Flowers The Oxford Flowers dataset [25] consists of 8,189 images in 102 separate intra-class flower categories. Following common practice, images were downsampled to 64×64 and samples were generated at the same resolution.

CUB Birds The CUB Birds dataset [36] consists of 11,788 images in 200 separate intra-class bird categories. Following common practice, images were downsampled to 64×64 and samples were generated at the same resolution.

5.3. Qualitative Results

In Fig. 5 we show qualitative results of generated samples comparing POT-GAN to the benchmark GANs. We observe a noticeable improvement in both sample quality and diversity of POT-GAN when compared to the benchmark GAN baselines.

5.4. Quantitative Results

For assessing the improvement in quality and modal diversity of the POT-GAN model, we use Inception Score [29] and FID Score [15]. Methods such as VEEGAN [32] and WAE-GAN [33] are constructed to prevent mode collapse and increase modal diversity of the recovered distribution, but in doing so sacrifice sample quality; they perform poorly on both Inception and FID scores. In Table 2 we show Inception and FID scores of POT-GAN along with benchmark models on the CIFAR-10, Oxford Flowers and CUB Birds datasets. These results correlate with the sample quality presented in Fig. 5 where POT-GAN achieves a considerable improvement over the other models. For an additional ablation study on the choice of $\mathcal{F}$, please refer to the supplementary material.

5.5. Effects of regularisation

Finally, we study the effects of adding our optimal transport regularisation term to the Generator used in the WGAN-GP [13] framework over the entire GAN training process. We show the added value the regularisation term provides over this baseline along the entire optimisation process. The Inception and FID scores for CIFAR-10 are plotted over Generator iterations shown in Fig. 4. In both plots we see POT-GAN converging to a better state than

Inception Score : $\exp(\mathbb{E}_{\boldsymbol{x}} KL(p(y\|\boldsymbol{x})\|\|p(y)))$					
	DCGAN [27]	VEEGAN [32]	WAE-GAN [33]	WGAN-GP [13]	POT-GAN (Ours)
CIFAR-10 [19]	6.16±0.07	6.25±0.05	4.18±0.04	6.58±0.06	**6.87±0.04**
Oxford Flowers [25]	2.33±0.04	2.11±0.02	2.30±0.01	3.42±0.04	**3.53±0.03**
CUB Birds [36]	3.93±0.03	3.74±0.02	3.42±0.04	4.51±0.04	**4.78±0.04**
FID Score : $\|\boldsymbol{m} - \boldsymbol{m}_w\|_2^2 + Tr(\boldsymbol{C} + \boldsymbol{C}_w - 2(\boldsymbol{C}\boldsymbol{C}_w)^{1/2}$					
CIFAR-10 [19]	37.7	35.6	87.7	34.4	**32.5**
Oxford Flowers [25]	88.2	299.0	145.9	98.7	**65.7**
CUB Birds [36]	76.3	173.9	143.3	70.4	**58.6**

Table 2: Inception and FID scores for DCGAN, VEEGAN, WAE-GAN, WGAN-GP and POT-GAN on CIFAR-10, Oxford Flowers and CUB Birds datasets.

(a) DCGAN [27] (b) VEEGAN [32] (c) WGAN-GP [13] (d) POT-GAN (Ours)

Figure 5: Qualitative results on Oxford Flowers and CUB Birds datasets. Visually, when compared to the benchmark GAN models, POT-GAN offers a significant improvement in sample quality and diversity. Additional samples for CIFAR-10 can be found in the supplementary material.

WGAN-GP [13]. The additional regularisation term continuously estimates a more accurate Wasserstein distance and the Generator is constantly penalised for deviating from the lower dimension representation. In the supplementary material, we also show the critic loss curves for the CIFAR-10 dataset, which provides a good indication of the increased stability and improvement in convergence of POT-GAN.

6. Conclusion

In this paper, we have presented POT-GAN, an unsupervised learning approach for GANs which estimates the Wasserstein distance on the latent representation of the data, using this to regularise training of a GAN. We provide convergence rate guarantees when working in a lower dimension and show that by applying our latent space regularisation term to the Generator, we can yield significant improvements in sample quality and diversity when sampling from the recovered model distribution. Using our approach, we demonstrate significant improvements on the Inception and FID scores over several GAN baselines.

References

[1] M. Arjovsky and L. Bottou. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017. 1, 3

[2] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017. 1, 2, 3

[3] I. Belghazi, S. Rajeswar, A. Baratin, R. D. Hjelm, and A. Courville. Mine: mutual information neural estimation. *arXiv preprint arXiv:1801.04062*, 2018. 2

[4] T. Che, Y. Li, A. P. Jacob, Y. Bengio, and W. Li. Mode regularized generative adversarial networks. *arXiv preprint arXiv:1612.02136*, 2016. 2

[5] N. Courty, R. Flamary, and M. Ducoffe. Learning wasserstein embeddings. *arXiv preprint arXiv:1710.07457*, 2017. 1

[6] T. Doan, J. Monteiro, I. Albuquerque, B. Mazoure, A. Durand, J. Pineau, and R. D. Hjelm. Online adaptative curriculum learning for gans. *arXiv preprint arXiv:1808.00020*, 2018. 1

[7] J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016. 2

[8] R. Dudley. The speed of mean glivenko-cantelli convergence. *The Annals of Mathematical Statistics*, 40(1):40–50, 1969. 1, 3

[9] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016. 2

[10] J. H. Friedman. On bias, variance, 0/1loss, and the curse-of-dimensionality. *Data mining and knowledge discovery*, 1(1):55–77, 1997. 3

[11] A. Genevay, G. Peyré, and M. Cuturi. Learning generative models with sinkhorn divergences. *arXiv preprint arXiv:1706.00292*, 2017. 1

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 1, 2, 3, 6, 7

[13] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*, 2017. 1, 3, 4, 5, 6, 7, 8

[14] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015. 6

[15] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017. 2, 7

[16] Z. Hu, Z. Yang, R. Salakhutdinov, and E. P. Xing. On unifying deep generative models. *arXiv preprint arXiv:1706.00550*, 2017. 2

[17] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[18] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2, 4, 6

[19] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009. 7, 8

[20] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*, 2015. 2

[21] Z. Lin, A. Khetan, G. Fanti, and S. Oh. Pacgan: The power of two samples in generative adversarial networks. *arXiv preprint arXiv:1712.04086*, 2017. 1

[22] L. Mescheder, A. Geiger, and S. Nowozin. Which training methods for gans do actually converge? In *International Conference on Machine Learning*, pages 3478–3487, 2018. 1

[23] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018. 1

[24] H. Narayanan and S. Mitter. Sample complexity of testing the manifold hypothesis. In *Advances in Neural Information Processing Systems*, pages 1786–1794, 2010. 3

[25] M.-E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1447–1454. IEEE, 2006. 7, 8

[26] S. Nowozin, B. Cseke, and R. Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems*, pages 271–279, 2016. 1, 2

[27] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 6, 8

[28] K. Roth, A. Lucchi, S. Nowozin, and T. Hofmann. Stabilizing training of generative adversarial networks through regularization. *arXiv preprint arXiv:1705.09367*, 2017. 1

[29] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016. 1, 2, 7

[30] T. Salimans, H. Zhang, A. Radford, and D. Metaxas. Improving gans using optimal transport. *arXiv preprint arXiv:1803.05573*, 2018. 1

[31] V. Seguy, B. B. Damodaran, R. Flamary, N. Courty, A. Rolet, and M. Blondel. Large-scale optimal transport and mapping estimation. *arXiv preprint arXiv:1711.02283*, 2017. 1

[32] A. Srivastava, L. Valkoz, C. Russell, M. U. Gutmann, and C. Sutton. Veegan: Reducing mode collapse in gans using implicit variational learning. In *Advances in Neural Information Processing Systems*, pages 3310–3320, 2017. 2, 6, 7, 8

[33] I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schoelkopf. Wasserstein auto-encoders. *arXiv preprint arXiv:1711.01558*, 2017. 6, 7, 8

[34] C. Villani. Topics in optimal transportation (graduate studies in mathematics, vol. 58). 2003. 2, 4

[35] C. Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008. 2, 3

[36] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 7, 8

[37] J. Weed and F. Bach. Sharp asymptotic and finite-sample rates of convergence of empirical measures in wasserstein distance. *arXiv preprint arXiv:1707.00087*, 2017. 3

[38] Y. Zuo, G. Avraham, and T. Drummond. Generative adversarial forests for better conditioned adversarial learning. *arXiv preprint arXiv:1805.05185*, 2018. 5

[39] Y. Zuo, G. Avraham, and T. Drummond. Traversing latent space using decision ferns. *arXiv preprint arXiv:1812.02636*, 2018. 5

[40] Y. Zuo and T. Drummond. Fast residual forests: Rapid ensemble learning for semantic segmentation. In *Conference on Robot Learning*, pages 27–36, 2017. 5