Deep Generative Models 4/3

Part I: Wasserstein distances and the WGAN (+improvements)

Problems with vanilla GANs

The vanilla GAN formulation suffers from:

- Unstable Training
  - Limit cycles and general failure to converge
  - Sensitive to imbalances in generator/discriminator (in architecture and instance performance)

Problems with vanilla GANs

The vanilla GAN formulation suffers from:

- Unstable Training
  - Limit cycles and general failure to converge
  - Sensitive to imbalances in generator/discriminator (in architecture and instance performance)
- Mode Collapse

Problems with vanilla GANs

The vanilla GAN formulation suffers from:

- Unstable Training
  - Limit cycles and general failure to converge
  - Sensitive to imbalances in generator/discriminator (in architecture and instance performance)
- Mode Collapse

Several implementation-level changes improve performance:

- $-\log D$ trick **[Goo14]**
  - $\min_G \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \to \min_G \mathbb{E}_{z\sim p(z)}[-\log(D(G(z)))]$
- DCGAN **[Rad16]**
- Unrolled GANs **[Met16]**

Problems with vanilla GANs

The vanilla GAN formulation suffers from:

- Unstable Training
  - Limit cycles and general failure to converge
  - Sensitive to imbalances in generator/discriminator (in architecture and instance performance)
- Mode Collapse

Several implementation-level changes improve performance:

- $-\log D$ trick **[Goo14]**
  - $\min_G \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))] \rightarrow \min_G \mathbb{E}_{z\sim p(z)}[-\log(D(G(z)))]$
- DCGAN **[Rad16]**
- Unrolled GANs **[Met16]**

Is there a theoretical perspective to address all of these underlying problems (simultaneously)?

Theoretical perspective: Back to GANs

Recall cost $V(D,G)$ (for the vanilla GAN) **[Goo14]**:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{z\sim p(z)}[\log(1-D(G(z)))]$$

Theoretical perspective: Back to GANs

Recall cost $V(D, G)$ (for the vanilla GAN) **[Goo14]**:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_r(x)}[\log D(x)] + \mathbb{E}_{z \sim p(z)}[\log(1 - D(G(z)))]$$

Optimizing the discriminator,

$$\max_D V(D, G) = C(G) = -\log(4) + 2JSD(p_r \| p_\theta)$$

Theoretical perspective: Back to GANs

Recall cost $V(D, G)$ (for the vanilla GAN) **[Goo14]**:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_r(x)}[\log D(x)] + \mathbb{E}_{z \sim p(z)}[\log(1 - D(G(z)))]$$

Optimizing the discriminator,

$$\max_D V(D, G) = C(G) = -\log(4) + 2JSD(p_r \| p_\theta)$$

Where JSD is the Jensen-Shannon Divergence,

$$JSD(p_r \| p_\theta) = \frac{1}{2} KL(p_r \| \frac{p_r + p_g}{2}) + \frac{1}{2} KL(p_\theta \| \frac{p_r + p_\theta}{2})$$

Theoretical perspective: Back to GANs

Recall cost $V(D, G)$ (for the vanilla GAN) **[Goo14]**:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_r(x)}[\log D(x)] + \mathbb{E}_{z \sim p(z)}[\log(1 - D(G(z)))]$$

Optimizing the discriminator,

$$\max_D V(D, G) = C(G) = -\log(4) + 2JSD(p_r \| p_\theta)$$

Where JSD is the Jensen-Shannon Divergence,

$$JSD(p_r \| p_\theta) = \frac{1}{2} KL(p_r \| \frac{p_r + p_g}{2}) + \frac{1}{2} KL(p_\theta \| \frac{p_r + p_\theta}{2})$$

The Jenson-Shannon Divergence is a nice theoretical justification, but is it the right one to evaluate against? To take gradients against?

Theoretical perspective: Supports and Distances

Arjovsky et al. argue no.

Theoretical perspective: Supports and Distances

Arjovsky et al. argue no.

- In cases where p_r lives on a low-d manifold, p_r and p_θ may be supported on very different, even disjoint sets

Theoretical perspective: Supports and Distances

Arjovsky et al. argue no.

- In cases where p_r lives on a low-d manifold, p_r and p_θ may be supported on very different, even disjoint sets
- Many familiar tools for comparing distributions are no longer useful in this setting. (∞'s in KL divergence)

Theoretical perspective: Supports and Distances

Arjovsky et al. argue no.

- In cases where p_r lives on a low-d manifold, p_r and p_θ may be supported on very different, even disjoint sets
- Many familiar tools for comparing distributions are no longer useful in this setting. (∞'s in KL divergence)
- Even more so when using these tools for *learning*, not just comparison (i.e. taking gradients).

Suggest that the right tool for the job is the *Earth Mover's/Wasserstein Distance*.

EM/Wasserstein Distance

Consider a transportation problem **[Vil08]**: given a set of N bakeries and M cafes, what is the optimal way to transport loaves of bread between them?

Define $p_{i \in 1 \ldots N}$ the mass of bread held by each bakery, $q_{j \in 1 \ldots M}$ the mass of bread desired by each cafe. Define x_i, y_j the positions of bakeries and cafes.

We assume that $\sum_i p_i = \sum_j q_j = 1$, and cost is proportional to work (mass×distance).

Find an optimal *coupling* (i.e. plan, transport matrix, joint distribution) $\gamma_{i,j}$ the mass of bread moved from p_i to q_j. This defines the Earth Mover's (EM) distance:

$$EMD = \min_{\gamma} \sum_i \sum_j \|x_i - y_j\| \gamma_{i,j}$$

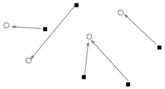
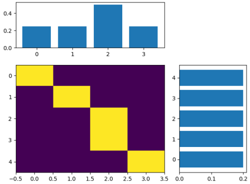


Fig. 3.2. Economic illustration of Monge's problem: squares stand for production units, circles for consumption places.



EM/Wasserstein Distance

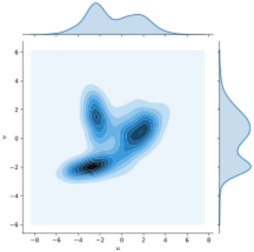
There are a set of Wasserstein distances, with $W_p(p_x, q_y)$ defined with $x \in M$,$y \in M$ and a distance D on x, y:

$$W_p = \inf_{\gamma \in \Pi(x,y)} \int_{M \times M} D(x,y)^p d\gamma(x,y)$$

Here $\Pi(x, y)$ represents the set of all joint distributions having p_x, q_y as their marginals. We will consider W_1 with $D(x, y)$ the Euclidean distance:

$$W_1 = \inf_{\gamma \in \Pi(x,y)} \int_{M \times M} \|x - y\| d\gamma(x,y) = \inf_{\gamma \in \Pi(x,y)} \mathbb{E}[\|x - y\|]$$

This identifies the EMD and W_1 under a common interpretation.



[image from https://en.wikipedia.org/wiki/Wasserstein$_m etric$]

An Illustrative Example

Take the mapping $\theta \to p_\theta$. Given a sequence of distributions p_{θ_t} parametrized by a sequence θ_t, we would like convergence in θ_t to imply convergence in p_{θ_t} ($\theta \to p_\theta$ is continuous).

An Illustrative Example

Take the mapping $\theta \to p_\theta$. Given a sequence of distributions p_{θ_t} parametrized by a sequence θ_t, we would like convergence in θ_t to imply convergence in p_{θ_t} ($\theta \to p_\theta$ is continuous).

Example: Let $Z \sim U[0,1]$ be the uniform distribution on the unit interval. Let $\mathbb{P}_0$ be the distribution of $(0, Z) \in R^2$. uniform on a straight line centered at the origin. Now let $\mathbb{P}_\theta$ be the distribution of (θ, Z) on R^2.

$$KL(\mathbb{P}_\theta \| \mathbb{P}_0)) = KL(\mathbb{P}_0 \| \mathbb{P}_\theta)) = \begin{cases} \infty, & \text{if } \theta \neq 0 \\ 0, & \text{if } \theta = 0 \end{cases}$$

Take the mapping $\theta \to p_\theta$. Given a sequence of distributions p_{θ_t} parametrized by a sequence θ_t, we would like convergence in θ_t to imply convergence in p_{θ_t} ($\theta \to p_\theta$ is continuous).

Example: Let $Z \sim U[0,1]$ be the uniform distribution on the unit interval. Let $\mathbb{P}_0$ be the distribution of $(0, Z) \in R^2$. uniform on a straight line centered at the origin. Now let $\mathbb{P}_\theta$ be the distribution of (θ, Z) on R^2.

$$KL(\mathbb{P}_\theta \| \mathbb{P}_0)) = KL(\mathbb{P}_0 \| \mathbb{P}_\theta)) = \begin{cases} \infty, & \text{if } \theta \neq 0 \\ 0, & \text{if } \theta = 0 \end{cases}$$

$$JS(\mathbb{P}_\theta \| \mathbb{P}_0)) = \begin{cases} \log 2, & \text{if } \theta \neq 0 \\ 0, & \text{if } \theta = 0 \end{cases}$$

An Illustrative Example

Take the mapping $\theta \to p_\theta$. Given a sequence of distributions p_{θ_t} parametrized by a sequence θ_t, we would like convergence in θ_t to imply convergence in p_{θ_t} ($\theta \to p_\theta$ is continuous).

Example: Let $Z \sim U[0,1]$ be the uniform distribution on the unit interval. Let $\mathbb{P}_0$ be the distribution of $(0, Z) \in R^2$. uniform on a straight line centered at the origin. Now let $\mathbb{P}_\theta$ be the distribution of (θ, Z) on R^2.

$$KL(\mathbb{P}_\theta \| \mathbb{P}_0)) = KL(\mathbb{P}_0 \| \mathbb{P}_\theta)) = \begin{cases} \infty, & \text{if } \theta \neq 0 \\ 0, & \text{if } \theta = 0 \end{cases}$$

$$JS(\mathbb{P}_\theta \| \mathbb{P}_0)) = \begin{cases} \log 2, & \text{if } \theta \neq 0 \\ 0, & \text{if } \theta = 0 \end{cases}$$

$$W_1(\mathbb{P}_\theta \| \mathbb{P}_0)) = |\theta|$$

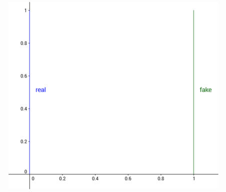
An Illustrative Example

Example: Let $Z \sim U[0,1]$ be the uniform distribution on the unit interval. Let $\mathbb{P}_0$ be the distribution of $(0, Z) \in R^2$. uniform on a straight line centered at the origin. Now let $\mathbb{P}_\theta$ be the distribution of (θ, Z) on R^2.

$$KL(\mathbb{P}_\theta \| \mathbb{P}_0)) = KL(\mathbb{P}_0 \| \mathbb{P}_\theta)) = \begin{cases} \infty, & \text{if } \theta \neq 0 \\ 0, & \text{if } \theta = 0 \end{cases}$$

$$JS(\mathbb{P}_\theta, \mathbb{P}_0)) = \begin{cases} \log 2, & \text{if } \theta \neq 0 \\ 0, & \text{if } \theta = 0 \end{cases}$$

$$W_1(\mathbb{P}_\theta, \mathbb{P}_0)) = |\theta|$$



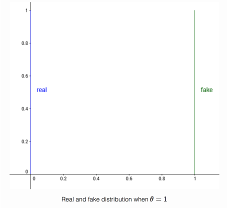
Real and fake distribution when $\theta = 1$

Example: Let $Z \sim U[0,1]$ be the uniform distribution on the unit interval. Let $\mathbb{P}_0$ be the distribution of $(0, Z) \in R^2$. uniform on a straight line centered at the origin. Now let $\mathbb{P}_\theta$ be the distribution of (θ, Z) on R^2.

$$KL(\mathbb{P}_\theta || \mathbb{P}_0)) = KL(\mathbb{P}_0 || \mathbb{P}_\theta)) = \begin{cases} \infty, & \text{if } \theta \neq 0 \\ 0, & \text{if } \theta = 0 \end{cases}$$

$$JS(\mathbb{P}_\theta, \mathbb{P}_0)) = \begin{cases} \log 2, & \text{if } \theta \neq 0 \\ 0, & \text{if } \theta = 0 \end{cases}$$

$$W_1(\mathbb{P}_\theta, \mathbb{P}_0)) = |\theta|$$



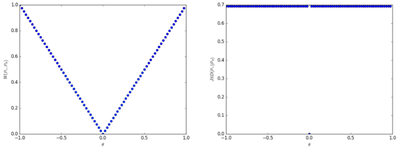


Figure 1: These plots show $\rho(\mathbb{P}_\theta, \mathbb{P}_0)$ as a function of θ when ρ is the EM distance (left plot) or the JS divergence (right plot). The EM plot is continuous and provides a usable gradient everywhere. The JS plot is not continuous and does not provide a usable gradient.

[left image from https://www.alexirpan.com/2017/02/22/wasserstein-gan.html]

How is Wasserstein Different?

- Can be accessed only as the result of an optimization
- Defines a space of joint distributions
- Induces a weaker topology (rabbit hole)

How is Wasserstein Different?

- Can be accessed only as the result of an optimization
- Defines a space of joint distributions
- Induces a weaker topology (rabbit hole)

Formally:
Theorem 1. Let $\mathbb{P}_r$ be a fixed distribution over $\mathcal{X}$. Let Z be a random variable (e.g. Gaussian) over another space $\mathcal{Z}$. Let $g : \mathcal{Z} \times \mathbb{R}^d \to \mathcal{X}$ be a function that will be denoted $g_\theta(z)$ with z the first coordinate and θ the second. Let $\mathbb{P}_\theta$ denote the distribution of $g_\theta(Z)$. Then,

- If g is continuous in θ, so is $W_1(\mathbb{P}_r, \mathbb{P}_\theta)$.
- If g is locally Lipschitz and satisfies regularity assumption 1, then $W_1(\mathbb{P}_r, \mathbb{P}_\theta)$ is continuous everywhere, and differentiable almost everywhere.
- Statements 1-2 are false for the Jensen-Shannon divergence $JSD(\mathbb{P}_r, \mathbb{P}_\theta)$ and all the KLs.

How is Wasserstein Different?

- Can be accessed only as the result of an optimization
- Defines a space of joint distributions
- Induces a weaker topology (rabbit hole)

Formally:

Theorem 1. Let $\mathbb{P}_r$ be a fixed distribution over $\mathcal{X}$. Let Z be a random variable (e.g. Gaussian) over another space $\mathcal{Z}$. Let $g : \mathcal{Z} \times \mathbb{R}^d \to \mathcal{X}$ be a function that will be denoted $g_\theta(z)$ with z the first coordinate and θ the second. Let $\mathbb{P}_\theta$ denote the distribution of $g_\theta(Z)$. Then,

- If g is continuous in θ, so is $W_1(\mathbb{P}_r, \mathbb{P}_\theta)$.
- If g is locally Lipschitz and satisfies regularity assumption 1, then $W_1(\mathbb{P}_r, \mathbb{P}_\theta)$ is continuous everywhere, and differentiable almost everywhere.
- Statements 1-2 are false for the Jensen-Shannon divergence $JSD(\mathbb{P}_r, \mathbb{P}_\theta)$ and all the KLs.

Corollary 1. Let g_θ be any feedforward neural network parametrized by θ, and $p(z)$ a prior over z such that $\mathbb{E}_{z\sim p(z)}[||z||] < \infty$. Then assumption 1 is satisfied and therefore $W(\mathbb{P}_r, \mathbb{P}_\theta)$ is continuous everywhere and differentiable almost everywhere.

Implementation: tractable costs

- We may now agree that $\min_\theta W_1(\mathbb{P}_r, \mathbb{P}_\theta)$ is a good thing to do.

Implementation: tractable costs

- We may now agree that $\min_\theta W_1(\mathbb{P}_r, \mathbb{P}_\theta)$ is a good thing to do.
    - However, $\min_\theta \inf \mathbb{E}_\gamma(\|x-y\|)$ is intractable for our cases of interest!

- We may now agree that $\min_\theta W_1(\mathbb{P}_r, \mathbb{P}_\theta)$ is a good thing to do.
  - However, $\min_\theta \inf \mathbb{E}_\gamma(\|x - y\|)$ is intractable for our cases of interest!

Kantorovich-Rubenstein Duality:

$$W_1(\mathbb{P}_r, \mathbb{P}_\theta) = \inf_\gamma \mathbb{E}_\gamma(\|x - y\|) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim \mathbb{P}_r}[f(x)] - \mathbb{E}_{x \sim \mathbb{P}_\theta}[f(x)]$$

Implementation: Tractable costs

Go back to the EMD Picture:

$$EMD = \min_{\gamma} \sum_i \sum_j \|x_i - y_j\| \gamma_{i,j}$$

Take $x = \text{vec}(\gamma), c = \text{vec}(\|x - y\|)$, $b = [p_i, q_j]^T$, A gives correct marginalization $b = Ax$. Then EMD calculation becomes an LP problem:

$$EMD = \min_{x} c^T x \qquad \text{(s.t. } Ax = b, x \geq 0\text{)}$$

We can then solve the dual problem (strong duality holds):

$$EMD = \max_{\phi} b^T \phi \qquad \text{(s.t. } A^T \phi \leq c\text{)}$$

Recall that $b = [p_i, q_j]^T$. Divide ϕ into f_1, f_2. By constraint arguments, we can show that optimally, $f_2 = -f_1 = f$, and that changes in f should be bounded by the distance between points. This gives:

$$EMD = \sup_{\|f\|_L \leq 1} \sum_j f_j q_j - \sum_i f_i p_i$$

Interpretation [Vil08]: Here, f is the *price* of buying/selling loaves of bread at bakeries/cafes at x_i/y_j. [derivation from https://vincentherrmann.github.io/blog/wasserstein/]

Implementation: Approximations

Approximation 1: Restriction to a parametric family of functions.
We will optimize over a family $\{f_w\}_{w \in \mathcal{W}}$ that are all K-Lipschitz for some K:

$$\min_{\theta} W_1(\mathbb{P}_r, \mathbb{P}_\theta) \approx \min_{\theta} \max_{w \in \mathcal{W}} \mathbb{E}_{x \sim \mathbb{P}_r}[f_w(x)] - \mathbb{E}_{z \sim \mathbb{P}_z}[f_w(g_\theta(z))]$$

- Evaluation gives $W_1(\mathbb{P}_r, \mathbb{P}_\theta)$ up to a multiplicative constant.
- Differentiation w.r.t θ gives $\frac{d}{d\theta} W_1(\mathbb{P}_r, \mathbb{P}_\theta)$ up to a multiplicative constant.

Implementation: Approximations

Approximation 1: Restriction to a parametric family of functions.
We will optimize over a family $\{f_w\}_{w \in \mathcal{W}}$ that are all K-Lipschitz for some K:

$$\min_{\theta} W_1(\mathbb{P}_r, \mathbb{P}_\theta) \approx \min_{\theta} \max_{w \in \mathcal{W}} \mathbb{E}_{x \sim \mathbb{P}_r}[f_w(x)] - \mathbb{E}_{z \sim \mathbb{P}_z}[f_w(g_\theta(z))]$$

- Evaluation gives $W_1(\mathbb{P}_r, \mathbb{P}_\theta)$ up to a multiplicative constant.
- Differentiation w.r.t θ gives $\frac{d}{d\theta} W_1(\mathbb{P}_r, \mathbb{P}_\theta)$ up to a multiplicative constant.

Approximation 2: Implementation of $\mathcal{W}$ via clipping.
If the weights of the network are in a compact space, the network will be K-Lipschitz for some K.

- Clip the weights of the network to a fixed box after each gradient update
- Not the same as updating within the constraints.

Implementation: Approximations

Approximation 1: Restriction to a parametric family of functions.
We will optimize over a family $\{f_w\}_{w \in \mathcal{W}}$ that are all K-Lipschitz for some K:

$$\min_\theta W_1(\mathbb{P}_r, \mathbb{P}_\theta) \approx \min_\theta \max_{w \in \mathcal{W}} \mathbb{E}_{x \sim \mathbb{P}_r}[f_w(x)] - \mathbb{E}_{z \sim \mathbb{P}_z}[f_w(g_\theta(z))]$$

- Evaluation gives $W_1(\mathbb{P}_r, \mathbb{P}_\theta)$ up to a multiplicative constant.
- Differentiation w.r.t θ gives $\frac{d}{d\theta} W_1(\mathbb{P}_r, \mathbb{P}_\theta)$ up to a multiplicative constant.

Approximation 2: Implementation of $\mathcal{W}$ via clipping.
If the weights of the network are in a compact space, the network will be K-Lipschitz for some K.

- Clip the weights of the network to a fixed box after each gradient update
- Not the same as updating within the constraints.

Approximation 3: Monte Carlo estimates

- As is standard practice, expectations are approximated via Monte Carlo sampling.
- How does this interact with W_1 as opposed to JS distance?
- *Restrictions inherit optimality

What has actually changed?

GAN Cost:

$$\min_{\theta}\max_{w} V(D,g)\begin{cases} &= \max_w \mathbb{E}_{x\sim\mathbb{P}_r}[\log(D_w(x)] + \mathbb{E}_{z\sim\mathbb{P}_z}[\log(1-D_w(g_\theta(z)))] \\ &= \min_\theta -\mathbb{E}_{z\sim\mathbb{P}_z}[D_w(g_\theta(z))] \end{cases}$$

WGAN Cost:

$$\min_{\theta}\max_{w\in\mathcal{W}} \tilde{V}(f,g) = \mathbb{E}_{x\sim\mathbb{P}_r}[f_w(x)] - \mathbb{E}_{z\sim\mathbb{P}_z}[f_w(g_\theta(z))]$$

- Unified cost
- Weights w are now clipped to a restricted range $\mathcal{W}$
- Remove log sigmoid nonlinearity from output of D_w to recover f_w

What has actually changed?

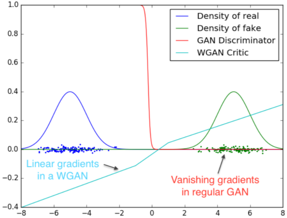
GAN Cost:

$$\min_\theta \max_w V(D,g) \begin{cases} = \max_w \mathbb{E}_{x\sim\mathbb{P}_r}[\log(D_w(x)] + \mathbb{E}_{z\sim\mathbb{P}_z}[\log(1 - D_w(g_\theta(z)))] \\ = \min_\theta -\mathbb{E}_{z\sim\mathbb{P}_z}[D_w(g_\theta(z))] \end{cases}$$

WGAN Cost:

$$\min_\theta \max_{w\in\mathcal{W}} \tilde{V}(f,g) = \mathbb{E}_{x\sim\mathbb{P}_r}[f_w(x)] - \mathbb{E}_{z\sim\mathbb{P}_z}[f_w(g_\theta(z))]$$

- Unified cost
- Weights w are now clipped to a restricted range $\mathcal{W}$
- Remove log sigmoid nonlinearity from output of D_w to recover f_w



Practical Implications

- We can now train the critic to optimality, no concerns about saturation and loss of the gradient if the critic becomes too good.
- Avoids mode collapse (?)

Empirical Results

Meaningful loss metric:

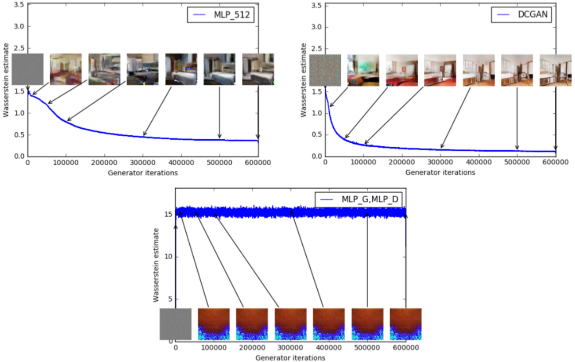


Figure 3: Training curves and samples at different stages of training. We can see a clear correlation between lower error and better sample quality. Upper left: the generator is an MLP with 4 hidden layers and 512 units at each layer. The loss decreases constistently as training progresses and sample quality increases. Upper right: the generator is a standard DCGAN. The loss decreases quickly and sample quality increases as well. In both upper plots the critic is a DCGAN without the sigmoid so losses can be subjected to comparison. Lower half: both the generator and the discriminator are MLPs with substantially high learning rates (so training failed). Loss is constant and samples are constant as well. The training curves were passed through a median filter for visualization purposes.

Compare JS:

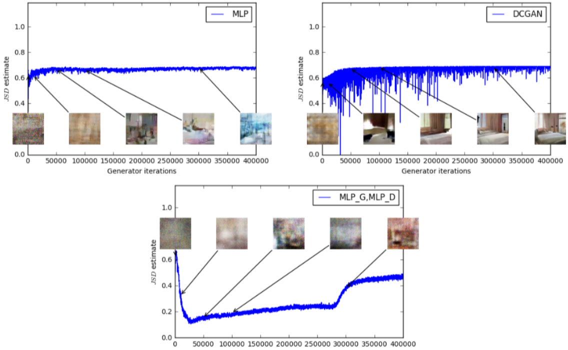


Figure 4: JS estimates for an MLP generator (upper left) and a DCGAN generator (upper right) trained with the standard GAN procedure. Both had a DCGAN discriminator. Both curves have increasing error. *Samples get better for the DCGAN but the JS estimate increases or stays constant, pointing towards no significant correlation between sample quality and loss. Bottom: MLP with both generator and discriminator. The curve goes up and down regardless of sample quality. All training curves were passed through the same median filter as in Figure 3.*

Empirical Results

Stability w.r.t. architecture:

Figure 5: Algorithms trained with a DCGAN generator. Left: WGAN algorithm. Right: standard GAN formulation. Both algorithms produce high quality samples.

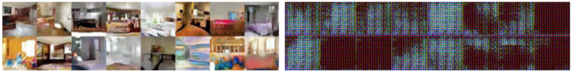


Figure 6: Algorithms trained with a generator without batch normalization and constant number of filters at every layer (as opposed to duplicating them every time as in [18]). Aside from taking out batch normalization, the number of parameters is therefore reduced by a bit more than an order of magnitude. Left: WGAN algorithm. Right: standard GAN formulation. As we can see the standard GAN failed to learn while the WGAN still was able to produce samples.

Figure 7: Algorithms trained with an MLP generator with 4 layers and 512 units with ReLU nonlinearities. The number of parameters is similar to that of a DCGAN, but it lacks a strong inductive bias for image generation. Left: WGAN algorithm. Right: standard GAN formulation. The WGAN method still was able to produce samples, lower quality than the DCGAN, and of higher quality than the MLP of the standard GAN. Note the significant degree of mode collapse in the GAN MLP.

How relevant is the theory?

Potential Problems

- How are distances implemented when estimating expectations via monte carlo?
- If the compact space $\mathcal{W}$ is very large (i.e. K-Lipschitz for K large), will we ever reach a limit?
- Simpler explanations (matching capacity, etc.)

On the other hand..

- Correlates well with objective
- Simplifies architectures
- Lends basis for sanity checks, improvements (performance on toy problems, estimating K)

Theory-Driven Improvement: WGAN-GP

How else can we enforce Lipschitz continuity?

Theory-Driven Improvement: WGAN-GP

How else can we enforce Lipschitz continuity?
If f^*, the optimal critic is differentiable and $\mathbb{P}_r$ and $\mathbb{P}_\theta$ have support intersecting in a set of measure 0, f^* has gradient norm 1 almost everywhere under $\mathbb{P}_r$ and $\mathbb{P}_\theta$.

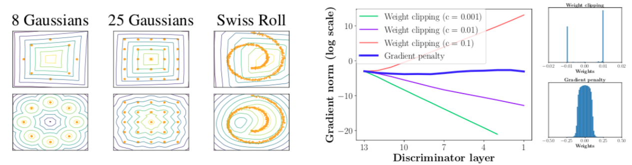
Theory-Driven Improvement: WGAN-GP

How else can we enforce Lipschitz continuity?
If f^*, the optimal critic is differentiable and $\mathbb{P}_r$ and $\mathbb{P}_\theta$ have support intersecting in a set of measure 0, f^* has gradient norm 1 almost everywhere under $\mathbb{P}_r$ and $\mathbb{P}_\theta$.
WGAN-GP cost:

$$\min_{\theta} \max_{w \in \mathcal{W}} \bar{V}(f, g) = \mathbb{E}_{x \sim \mathbb{P}_r}[f_w(x)] - \mathbb{E}_{z \sim \mathbb{P}_z}[f_w(g_\theta(z))] + \lambda \mathbb{E}_{\hat{x} \sim \mathbb{P}_{\hat{x}}}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2]$$

How else can we enforce Lipschitz continuity?
If f^*, the optimal critic is differentiable and $\mathbb{P}_r$ and $\mathbb{P}_\theta$ have support intersecting in a set of measure 0, f^* has gradient norm 1 almost everywhere under $\mathbb{P}_r$ and $\mathbb{P}_\theta$.
WGAN-GP cost:

$$\min_{\theta} \max_{w \in \mathcal{W}} \bar{V}(f, g) = \mathbb{E}_{x \sim \mathbb{P}_r}[f_w(x)] - \mathbb{E}_{z \sim \mathbb{P}_z}[f_w(g_\theta(z))] + \lambda \mathbb{E}_{\hat{x} \sim \mathbb{P}_{\hat{x}}}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2]$$



(a) Value surfaces of WGAN critics trained to optimality on toy datasets using (top) weight clipping and (bottom) gradient penalty. Critics trained with weight clipping fail to capture higher moments of the data distribution. The 'generator' is held fixed at the real data plus Gaussian noise.

(b) (left) Gradient norms of deep WGAN critics during training on the Swiss Roll dataset either explode or vanish when using weight clipping, but not when using a gradient penalty. (right) Weight clipping (top) pushes weights towards two values (the extremes of the clipping range), unlike gradient penalty (bottom).

Figure 1: Gradient penalty in WGANs does not exhibit undesired behavior like weight clipping.

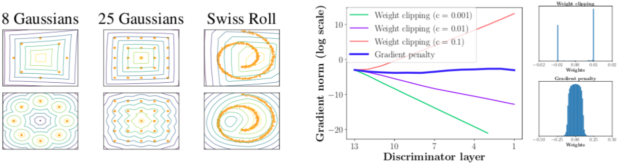
Theory-Driven Improvement: WGAN-GP

How else can we enforce Lipschitz continuity?

If f^*, the optimal critic is differentiable and $\mathbb{P}_r$ and $\mathbb{P}_\theta$ have support intersecting in a set of measure 0, f^* has gradient norm 1 almost everywhere under $\mathbb{P}_r$ and $\mathbb{P}_\theta$.

WGAN-GP cost:

$$\min_{\theta} \max_{w \in \mathcal{W}} \bar{V}(f, g) = \mathbb{E}_{x \sim \mathbb{P}_r}[f_w(x)] - \mathbb{E}_{z \sim \mathbb{P}_z}[f_w(g_\theta(z))] + \lambda \mathbb{E}_{\hat{x} \sim \mathbb{P}_{\hat{x}}}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2]$$



(a) Value surfaces of WGAN critics trained to optimality on toy datasets using (top) weight clipping and (bottom) gradient penalty. Critics trained with weight clipping fail to capture higher moments of the data distribution. The ‘generator’ is held fixed at the real data plus Gaussian noise.

(b) (left) Gradient norms of deep WGAN critics during training on the Swiss Roll dataset either explode or vanish when using weight clipping, but not when using a gradient penalty. (right) Weight clipping (top) pushes weights towards two values (the extremes of the clipping range), unlike gradient penalty (bottom).

Figure 1: Gradient penalty in WGANs does not exhibit undesired behavior like weight clipping.

Inspired by, but within the theory?

Table 2: Outcomes of training 200 random architectures, for different success thresholds. For comparison, our standard DCGAN scored 7.24.

Min. score	Only GAN	Only WGAN-GP	Both succeeded	Both failed
1.0	0	8	192	0
3.0	1	88	110	1
5.0	0	147	42	11
7.0	1	104	5	90
9.0	0	0	0	200

Figure 2: Different GAN architectures trained with different methods. We only succeeded in training every architecture with a shared set of hyperparameters using WGAN-GP.

Connecting weight constraints to theory

Integral probability metrics:

$$d_{\mathcal{F}}(\mathbb{P}_r, \mathbb{P}_\theta) = \sup_{f \in \mathcal{F}} \mathbb{E}_{x \sim \mathbb{P}_r}[f(x)] - \mathbb{E}_{x \sim \mathbb{P}_\theta [f(x)]}$$

- $\mathcal{F} =$ 1-Lipschitz $\rightarrow d_{\mathcal{F}} = W_1$
- $\mathcal{F} =$ 1-Bounded $\rightarrow d_{\mathcal{F}} = \delta$ (TV)
- $\mathcal{F} = \{f \in \mathcal{H} : \|f\|_\infty \leq 1\} =$ MMD

Suggests potentially rich theoretical framework for understanding architecture-level changes.

Part II: Wasserstein Auto-Encoders

Motivation

VAE ELBO maximization:

$$\max_{\phi,\theta} \mathbb{E}_{Q_\phi(Z|X)} \log P_\theta(X|Z) - D_{KL}(Q_\phi(Z|X), P_\theta(Z)) \tag{1}$$

1. not guarantee that the aggregated posterior $\mathbb{E}_{P(X)} Q_\phi(Z|X)$ matches P_Z
2. require non-deterministic (always gaussian) encoder and random decoder to compute gradients

GAN objective function:

$$\min_G \max_D \mathbb{E}_{P(X)} \log(D(X)) + \mathbb{E}_{P(Z)} \log(1 - D(G(Z))) \tag{2}$$

1. sometimes maxout and provide no gradients when training

Formulation

Optimal transport (OT) problem:

$$W_c(P_X, P_G) = \inf_{\Gamma \in \mathcal{P}(X \in P_X, Y \in P_G)} \mathbb{E}_{(X,Y)\in\Gamma}[c(X,Y)], \tag{3}$$

when $c(x,y) = d^p(x,y), p \geq 1, W_c$, is p-Wasserstein distance.

Theorem:

$$\inf_{\Gamma \in \mathcal{P}(X \in P_X, Y \in P_G)} \mathbb{E}_{(X,Y)\in\Gamma}[c(X,Y)] = \inf_{Q:Q_Z = P_Z} \mathbb{E}_{P_X}\mathbb{E}_{Q(Z|X)}[c(X, G(Z))], \tag{4}$$

where $Q_Z(Z) = \mathbb{E}_{X\in P_X}[Q(Z|X)], P_G(X) = \int_{\mathcal{Z}} p_G(x|z)p_z(z)dz, p_G(x|z)$ is deterministic with any function $G : \mathcal{Z} \to \mathcal{X}$.

WAE objective function:

$$D_{\mathrm{WAE}}(P_X, P_G) = \inf_{Q(Z|X)\in\mathcal{Q}} \mathbb{E}_{P_X}\mathbb{E}_{Q(Z|X)}[c(X, G(Z))] + \lambda \cdot \mathcal{D}_Z(Q_Z, P_Z), \tag{5}$$

where $\mathcal{D}_Z$ can be arbitrary divergence between P_Z and Q_Z.

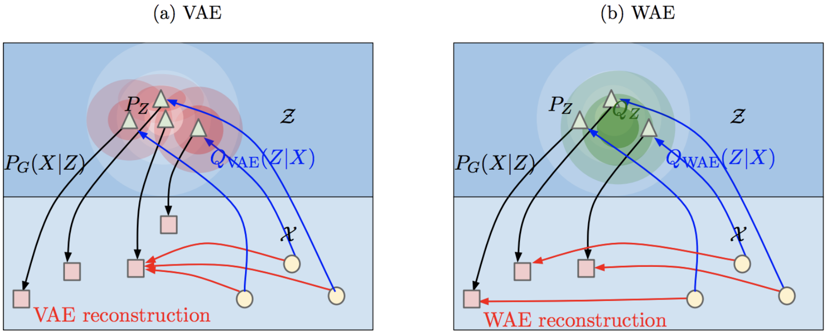


Figure 1: Both VAE and WAE minimize two terms: the reconstruction cost and the regularizer penalizing discrepancy between P_Z and distribution induced by the encoder Q. VAE forces $Q(Z|X=x)$ to match P_Z for all the different input examples x drawn from P_X. This is illustrated on picture (a), where every single red ball is forced to match P_Z depicted as the white shape. Red balls start intersecting, which leads to problems with reconstruction. In contrast, WAE forces the continuous mixture $Q_Z := \int Q(Z|X)dP_X$ to match P_Z, as depicted with the green ball in picture (b). As a result latent codes of different examples get a chance to stay far away from each other, promoting a better reconstruction.

Algorithm WAE-GAN

Option 1: $\mathcal{D}_Z = D_{\mathrm{JS}}(Q_Z, P_Z)$ and use adversarial discriminator D_γ to estimate it:

$$\inf_{Q(Z|X)\in\mathcal{Q}} \max_{D_\gamma} \mathbb{E}_{P_X}\mathbb{E}_{Q(Z|X)}[c(X, G(Z))] + \lambda\cdot\left(\mathbb{E}_{P_Z}\log D_\gamma(P_Z(Z)) + \mathbb{E}_{Q_Z}\log(1 - D_\gamma(Q_Z(Z)))\right) \tag{6}$$

Note:

1. Though it's min-max again, here we match the nice shape single mode (if gaussion prior) P_Z rather than unknown, complex, possibly multimodal P_X as in GAN.
2. $Q(Z|x) = \delta_{\mu_\phi(x)}, \mu_\phi(x) : \mathcal{X} \to \mathcal{Z}$.
3. When $c(x, y) = \|x - y\|_2^2$, WAE-GAN is equivalent to AAE.
4. The dual algorithm in WGAN does not apply to other cost W_c and does not have encoder.

Algorithm 1 Wasserstein Auto-Encoder with GAN-based penalty (WAE-GAN).

Require: Regularization coefficient $\lambda > 0$.

Initialize the parameters of the encoder Q_ϕ, decoder G_θ, and latent discriminator D_γ.

while (ϕ, θ) not converged **do**

  Sample $\{x_1, \ldots, x_n\}$ from the training set

  Sample $\{z_1, \ldots, z_n\}$ from the prior P_Z

  Sample $\tilde{z}_i$ from $Q_\phi(Z|x_i)$ for $i = 1, \ldots, n$

  Update D_γ by ascending:

$$\frac{\lambda}{n}\sum_{i=1}^{n}\log D_\gamma(z_i) + \log\left(1 - D_\gamma(\tilde{z}_i)\right)$$

  Update Q_ϕ and G_θ by descending:

$$\frac{1}{n}\sum_{i=1}^{n} c\left(x_i, G_\theta(\tilde{z}_i)\right) - \lambda\cdot\log D_\gamma(\tilde{z}_i)$$

end while

Algorithm WAE-MMD

Option 2:

$$\mathcal{D}_Z = \mathrm{MMD}_k(Q_Z, P_Z) = \| \int_{\mathcal{Z}} k(z, \cdot) dP_Z(z) - \int_{\mathcal{Z}} k(z, \cdot) dQ_Z(z) \|_{\mathcal{H}_k}, \tag{7}$$

where $k : \mathcal{Z} \times \mathcal{Z} \to \mathcal{R}$ is a positive-definite reproducing kernel, and $\mathcal{H}_k$ is the corresponding RKHS.

Note:

1. This is not a min-max game.
2. Use the unbiased U-statistic estimator in SGD.
3. Use $k(x, y) = C/(C + \|x - y\|_2^2), C = 2d_z\sigma_z^2$ as it has heavy tails than RBF kernels.
4. Papers [LSZ15, DRG15] estimate $\mathrm{MMD}_k(P_X, P_G)$, which requires number of samples roughly proportional to the dimensionality of the input space $\mathcal{X}$ for each mini-batch.

Algorithm 2 Wasserstein Auto-Encoder with MMD-based penalty (WAE-MMD).

Require: Regularization coefficient $\lambda > 0$, characteristic positive-definite kernel k.

Initialize the parameters of the encoder Q_ϕ, decoder G_θ, and latent discriminator D_γ.

while (ϕ, θ) not converged **do**

Sample $\{x_1, \ldots, x_n\}$ from the training set

Sample $\{z_1, \ldots, z_n\}$ from the prior P_Z

Sample $\tilde{z}_i$ from $Q_\phi(Z|x_i)$ for $i = 1, \ldots, n$

Update Q_ϕ and G_θ by descending:

$$\frac{1}{n}\sum_{i=1}^{n} c(x_i, G_\theta(\tilde{z}_i)) + \frac{\lambda}{n(n-1)}\sum_{\ell \neq j} k(z_\ell, z_j) + \frac{\lambda}{n(n-1)}\sum_{\ell \neq j} k(\tilde{z}_\ell, \tilde{z}_j) - \frac{2\lambda}{n^2}\sum_{\ell, j} k(z_\ell, \tilde{z}_j)$$

end while

Experiments

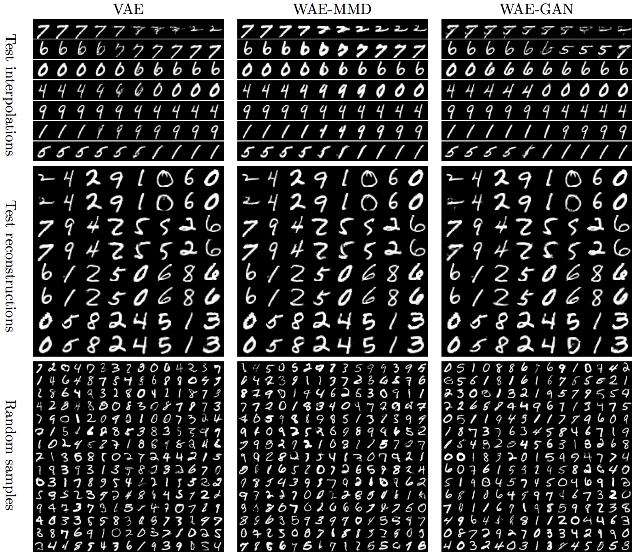


Figure 2: VAE (left column), WAE-MMD (middle column), and WAE-GAN (right column) trained on MNIST dataset. In "test reconstructions" odd rows correspond to the real test points.

Figure 3: VAE (left column), WAE-MMD (middle column), and WAE-GAN (right column) trained on CelebA dataset. In "test reconstructions" odd rows correspond to the real test points.

Two metrics:

1. Frechet Inception Distance (FID) [HRU+17]: smaller means the generated images are more similar to real ones.
2. sharpness: larger means less blurry of the image.

Algorithm	**FID**	**Sharpness**
VAE	63	3×10^{-3}
WAE-MMD	55	6×10^{-3}
WAE-GAN	42	6×10^{-3}
True data	2	2×10^{-2}

Table 1: FID (smaller is better) and sharpness (larger is better) scores for samples of various models for CelebA.

Conclusions: The images sampled from the trained WAE models are of better quality, without compromising the stability of training and the quality of reconstruction compared with VAE.

References

References

[DRG15] Gintare Karolina Dziugaite, Daniel M Roy, and Zoubin Ghahramani.
Training generative neural networks via maximum mean discrepancy optimization.
arXiv preprint arXiv:1505.03906, 2015.

[Goo14] Jean; Mirza Medhi; Xu Bing; Warde-Farley David; Ozair Sherjil; Courville Aaron; Bengio Yoshua Goodfellow, Ian; Pouget-Abadie.
Generative adversarial networks.
NIPS, 2014.

[HRU+17] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp 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.

[LSZ15] Yujia Li, Kevin Swersky, and Rich Zemel.
Generative moment matching networks.
In *International Conference on Machine Learning*, pages 1718–1727, 2015.

[Met16] Ben; Pfau David; Sohl-Dickstein Jascha Metz, Luke; Poole.
Unrolled generative adversarial networks.
arXiv, 2016.

[Rad16] Luke; Chintala Soumith Radford, Alec; Metz.
Unsupervised representation learning with deep convolutional generative adversarial networks.
arXiv, 2016.

[Vil08] Cedric Villani.
Optimal transport, old and new.
Springer, 2008.

Extra

E Generator's cost during normal GAN training

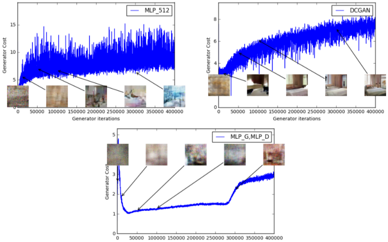


Figure 8: Cost of the generator during normal GAN training, for an MLP generator (upper left) and a DCGAN generator (upper right). Both had a DCGAN discriminator. ***Both curves have increasing error****. Samples get better for the DCGAN but the cost of the generator increases, pointing towards no significant correlation between sample quality and loss. Bottom: MLP with both generator and discriminator. The curve goes up and down regardless of sample quality. All training curves were passed through the same median filter as in Figure 3.*

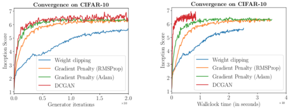


Figure 3: CIFAR-10 Inception score over generator iterations (left) or wall-clock time (right) for four models: WGAN with weight clipping, WGAN-GP with RMSProp and Adam (to control for the optimizer), and DCGAN. WGAN-GP significantly outperforms weight clipping and performs comparably to DCGAN.

Table 3: Inception scores on CIFAR-10. Our unsupervised model achieves state-of-the-art performance, and our conditional model outperforms all others except SGAN.

Unsupervised

Method	Score
ALI [8] (in [27])	$5.34 \pm .05$
BEGAN [4]	5.62
DCGAN [22] (in [11])	$6.16 \pm .07$
Improved GAN (-L+HA) [23]	$6.86 \pm .06$
EGAN-Ent-VI [7]	$7.07 \pm .10$
DFM [27]	$7.72 \pm .13$
WGAN-GP ResNet (ours)	$7.86 \pm .07$

Supervised

Method	Score
SteinGAN [26]	6.35
DCGAN (with labels, in [26])	6.58
Improved GAN [23]	$8.09 \pm .07$
AC-GAN [20]	$8.25 \pm .07$
SGAN-no-joint [11]	$8.37 \pm .08$
WGAN-GP ResNet (ours)	$8.42 \pm .10$
SGAN [11]	$8.59 \pm .12$

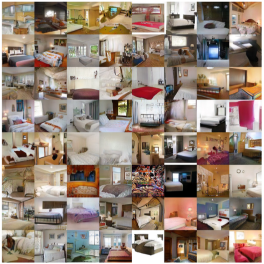


Figure 4: Samples of 128×128 LSUN bedrooms. We believe these samples are at least comparable to the best published results so far.

Table 4: Samples from a WGAN-GP character-level language model trained on sentences from the Billion Word dataset, truncated to 32 characters. The model learns to directly output one-hot character embeddings from a latent vector without any discrete sampling step. We were unable to achieve comparable results with the standard GAN objective and a continuous generator.

Busino game camperate spent odea	Solice Norkedin pring in since
In the bankaway of smarling the	ThiS record (31.) UBS) and Ch
SingersMay , who kill that imvic	It was not the annuas were plogr
Keray Pents of the same Reagun D	This will be us , the ect of DAN
Manging include a tudancs shat "	These leaded as most-worsd p2 a0
His Zuith Dudget , the Denmbern	The time I paid0a South Cubry i
In during the Uitational questio	Dour Fraps higs it was these del
Divos from The ’ noth ronkies of	This year out howneed allowed lo
She like Monday , of macunsuer S	Kaulna Seto consficutes to repor

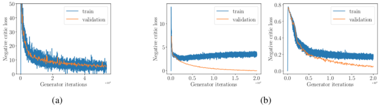


Figure 5: (a) The negative critic loss of our model on LSUN bedrooms converges toward a minimum as the network trains. (b) WGAN training and validation losses on a random 1000-digit subset of MNIST show overfitting when using either our method (left) or weight clipping (right). In particular, with our method, the critic overfits faster than the generator, causing the training loss to increase gradually over time even as the validation loss drops.