Generative Models

Shihua Zhang

December 29, 2021

Outline

- Generative Models
- VAE
- GAN
- Similarities between VAE and GAN
- Drawbacks of VAE and GAN
- **Solutions by Wasserstein Metric**
  - Wasserstein GAN
  - Wasserstein AE
- Unified Theory and Stronger Model
- Further Problems

Wasserstein Distance: An Introduction

- Monge Problem
  Given two mesure spaces $(X, \mathcal{F}_1, P_X)$, $(Y, \mathcal{F}_2, P_G)$ and a cost function $c(x, y)$, find the transport map $T : X \to Y$ which minimizes the total cost:

$$W_c(X, Y) = \inf_{T_*(P_X)=P_G} \mathbb{E}_{P_X}[c(X, Y)]$$

  Here $T_*(P_X) = P_G$ means $P_G(B) = P_X(T^{-1}(B))$ for $B \in \mathcal{F}_2$.

  We call W the Wasserstein Distance.

  Notice: optimal transport map T^{op} may not exist (for example, when P_X is a Dirac measure while P_G is not).

Wasserstein Distance: An Introduction

- Kantorovich Relaxation

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

Γ: coupling, a joint distribution satisfying marginal restriction
Notice: optimal coupling always exists

Wasserstein Distance: An Introduction

- Kantorovich Relaxation

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

  Γ: coupling, a joint distribution satisfying marginal restriction
  Notice: optimal coupling always exists

- Kantorovich duality

$$W_c(X, Y) = \sup_{\psi(y)+\phi(x) \leqslant c(x,y)} [\mathbb{E}_{P_X}\psi(y) + \mathbb{E}_{P_G}\phi(x)]$$

- Define c-transform $\phi^c(y) = \inf_x[c(x, y) - \phi(x)]$, then

$$W_c(X, Y) = \sup_{\phi}[\mathbb{E}_{P_X}\phi^c(y) + \mathbb{E}_{P_G}\phi(x)]$$

- If c is 1-norm and ϕ is 1-Lipchitz, $\phi^c = -\phi$ and then

$$W_c(X, Y) = \sup_{|\phi|_L \leqslant 1} [\mathbb{E}_{P_G}\phi(x) - \mathbb{E}_{P_X}\phi(y)]$$

Wasserstein Distance: A Better Metric

- Wasserstein Metric as a Weaker Metric

  Example 1 (Learning parallel lines). Let $Z \sim U[0,1]$ the uniform distribution on the unit interval. Let $\mathbb{P}_0$ be the distribution of $(0, Z) \in \mathbb{R}^2$ (a 0 on the x-axis and the random variable Z on the y-axis), uniform on a straight vertical line passing through the origin. Now let $g_\theta(z) = (\theta, z)$ with θ a single real parameter. It is easy to see that in this case,

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

- Observation
  - Only W converges at 0.
  - Moreover, only W is continuous (even differentiable) at 0.

Wasserstein Distance: A Better Metric

- Wasserstein Metric as a Weaker Metric

Theorem 1

Let $\mathbb{P}$ be a distribution on a compact space $\mathcal{X}$ and $(\mathbb{P}_n)_{n\in\mathbb{N}}$ be a sequence of distributions on $\mathcal{X}$. Then, considering all limits as $n \to \infty$.

1. *The following statements are equivalent*
   - $\delta(\mathbb{P}_n, \mathbb{P}) \to 0$ *with* δ *the total variation distance.*
   - $JS(\mathbb{P}_n, \mathbb{P}) \to 0$ *with JS the Jensen-Shannon divergence.*
2. *The following statements are equivalent*
   - $W(\mathbb{P}_n, \mathbb{P}) \to 0$
   - $\mathbb{P}_n \xrightarrow{\mathcal{D}} \mathbb{P}$ *where* $\xrightarrow{\mathcal{D}}$ *represents convergence in distribution for random variables.*
3. $KL(\mathbb{P}_n|\mathbb{P}) \to 0$ *or* $KL(\mathbb{P}|\mathbb{P}_n) \to 0$ *imply the statements in 1.*
4. *The statements in 1 imply the statements in 2.*

- Conclusion: Difficulty to convergence: $KL > JS > W$.

Wasserstein Distance: A Better Metric

- Wasserstein Metric is indeed continuous and differentiable[1]

Theorem 2

Let $\mathbb{P}_r$ be a fixed distribution over $\mathcal{X}$. Let $\mathcal{Z}$ be a random variable (e.g. Gaussian) over another space Z. Let $g : \mathcal{Z} \times \mathbb{R}^d \to \mathcal{X}$ be a function, which will be denoted as $g_\theta(z)$ with z the first coordinate and θ the second. Let $\mathbb{P}_\theta$ denote the distribution of $g_\theta(z)$. Then,
1. If g is continuous in θ, so is $W(\mathbb{P}_r, \mathbb{P}_\theta)$.
2. If g is locally Lipschitz and satisfies a regularity assumption, then $W(\mathbb{P}_r, \mathbb{P}_\theta)$ is continuous everywhere, and differentiable almost everywhere.
3. Statements 1-2 are false for the Jensen-Shannon divergence $JS(\mathbb{P}_r, \mathbb{P}_\theta)$ and all the KLs.

[1] Arjovsky M , Chintala S , Bottou L . Wasserstein GAN[J]. 2017.

Wasserstein Distance: A Better Metric

- Wasserstein Metric is indeed continuous and differentiable

Theorem 3

Let g_θ be any feedforward neural network parameterized by θ, and p(z) a prior over z such that $\mathbb{E}_{z\sim p(z)}[|z|] \leqslant \infty$ (e.g. Gaussian, uniform, etc.). Then the regularity assumption mentioned above is satisfied and therefore $W(\mathbb{P}_r, \mathbb{P}_\theta)$ is continuous everywhere and differentiable almost everywhere.

Wasserstein metric is more suitable than JS and KL when measuring the distance between the generated and real data distributions.

Outline

- Generative Models
- VAE
- GAN
- Similarities between VAE and GAN
- Drawbacks of VAE and GAN
- Solutions by Wasserstein Metric
  - **Wasserstein GAN**
  - Wasserstein AE
- Unified Theory and Stronger Model
- Further Problems

Wasserstein GAN: Theory

- Different with the original GAN metric: $JS(\mathbb{P}_g, \mathbb{P}_r) \longrightarrow W(\mathbb{P}_g, \mathbb{P}_r)$
- Optimization objective

$$W(\mathbb{P}_X, \mathbb{P}_G) = \inf_{\gamma \in P(P_X, P_G)} \mathbb{E}_{\Gamma(X,Y)}[|x - y|]$$

Wasserstein GAN: Theory

- Different with the original GAN metric: $JS(\mathbb{P}_g, \mathbb{P}_r) \longrightarrow W(\mathbb{P}_g, \mathbb{P}_r)$
- Optimization objective

$$W(\mathbb{P}_X, \mathbb{P}_G) = \inf_{\gamma \in P(P_X, P_G)} \mathbb{E}_{\Gamma(X,Y)}[|x - y|]$$

Since the formula is intractable, we choose the dual fomulation:

$$W(\mathbb{P}_X, \mathbb{P}_G) = \sup_{|\phi|_L \leqslant 1} [\mathbb{E}_{P_G}\phi(x) - \mathbb{E}_{P_X}\phi(y)]$$

Wasserstein GAN: Theory

- Different with the original GAN metric: $JS(\mathbb{P}_g, \mathbb{P}_r) \longrightarrow W(\mathbb{P}_g, \mathbb{P}_r)$
- Optimization objective

$$W(\mathbb{P}_X, \mathbb{P}_G) = \inf_{\gamma \in P(P_X, P_G)} \mathbb{E}_{\Gamma(X,Y)}[|x - y|]$$

Since the formula is intractable, we choose the dual fomulation:

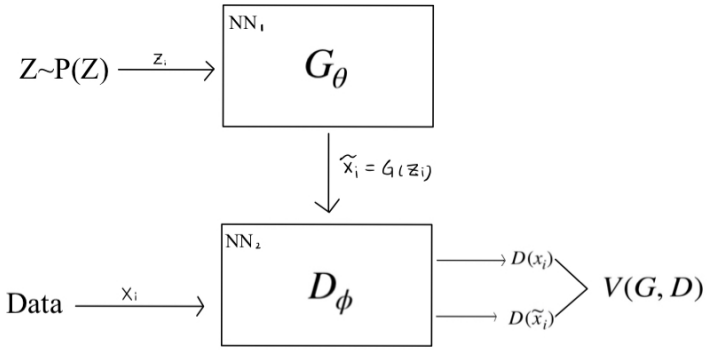
$$W(\mathbb{P}_X, \mathbb{P}_G) = \sup_{|\phi|_L \leqslant 1} [\mathbb{E}_{P_G}\phi(x) - \mathbb{E}_{P_X}\phi(y)]$$

Lipchitz can be obtained by setting weight space $\mathcal{W}$ as a compact space:

$$K \cdot W(\mathbb{P}_X, \mathbb{P}_G) = \max_{w \in \mathcal{W}} [\mathbb{E}_{P_G}\phi_w(x) - \mathbb{E}_{P_X}\phi_w(y)]$$

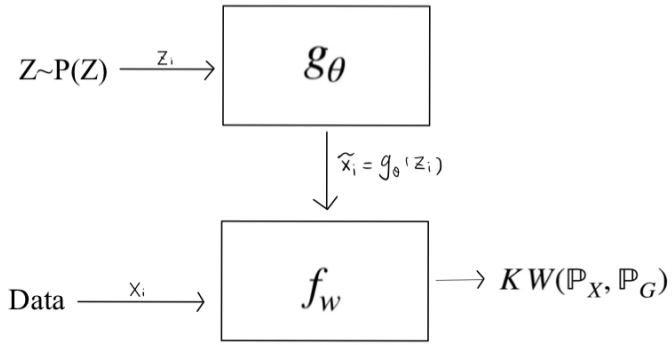
Wasserstein GAN: Neural Network Structure

GAN:



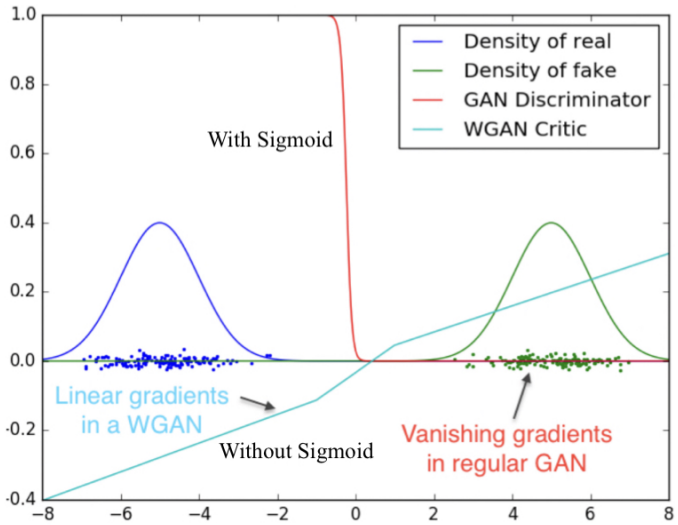
$$\min_\theta \max_w [\mathbb{E}_{P_{data}} \log D_w(x) - \mathbb{E}_{P_z} \log D_w(G_\theta(z)))]$$

WGAN:



$$\min_\theta \max_w [\mathbb{E}_{P_{data}} f_w(x) - \mathbb{E}_{P_z} f_w(G_\theta(z))]$$

Difference:
1. No sigmoid in output layer.
2. No log term in loss function.
3. Weight Clipping



Wasserstein GAN: Algorithm

Algorithm 1 WGAN, our proposed algorithm. All experiments in the paper used the default values $\alpha = 0.00005$, $c = 0.01$, $m = 64$, $n_{\text{critic}} = 5$.

Require: : α, the learning rate. c, the clipping parameter. m, the batch size. n_{critic}, the number of iterations of the critic per generator iteration.

Require: : w_0, initial critic parameters. θ_0, initial generator's parameters.

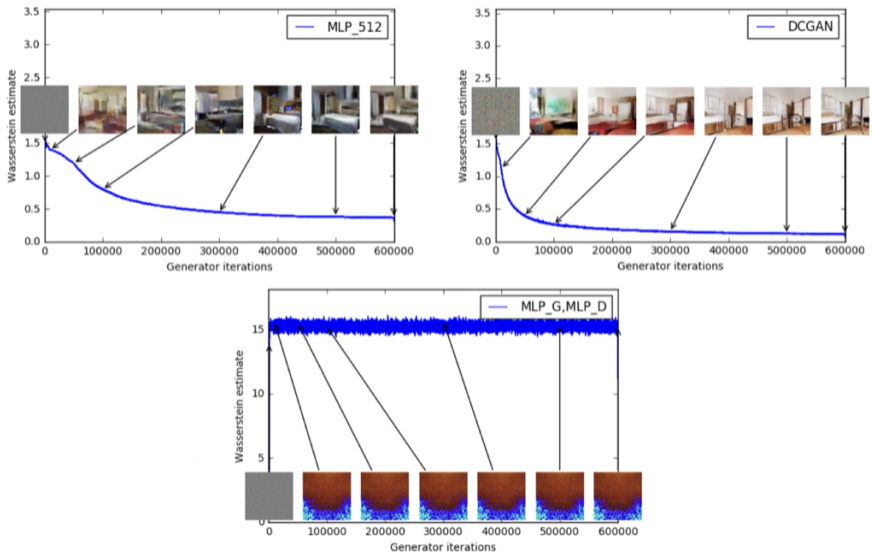
while θ has not converged **do**
  for $t = 0, ..., n_{\text{critic}}$ **do**
    Sample $\{x^{(i)}\}_{i=1}^{m} \sim \mathbb{P}_r$ a batch from the real data.
    Sample $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$ a batch of prior samples.
    $g_w \leftarrow \nabla_w \left[\frac{1}{m}\sum_{i=1}^{m} f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))\right]$
    $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$
    $w \leftarrow \text{clip}(w, -c, c)$
  end for
  Sample $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$ a batch of prior samples.
  $g_\theta \leftarrow -\nabla_\theta \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))$
  $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$
end while

Wasserstein GAN: Experimental Result

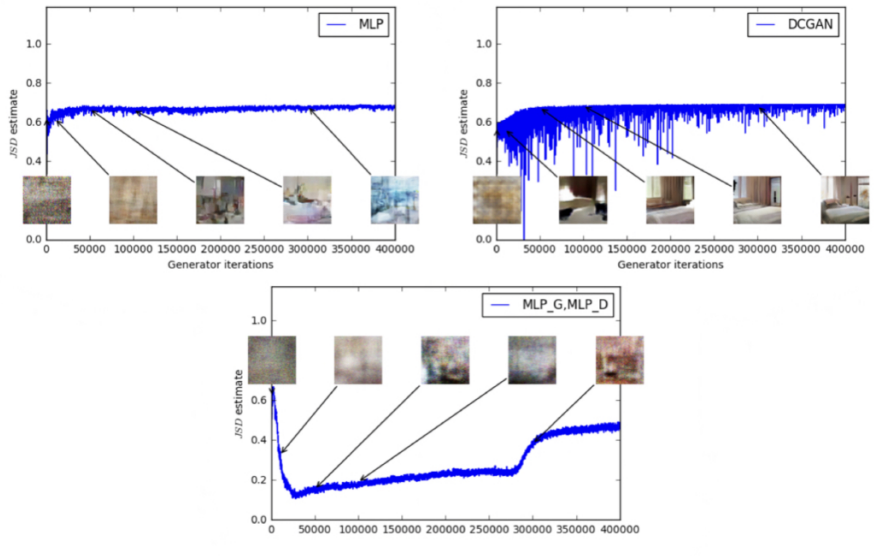
- Meaningful Loss Metric
  Notice: we can (and should) train the critic till optimality, which actually equals to $K \cdot W$

We have a explicit loss metric!

W:



JS:

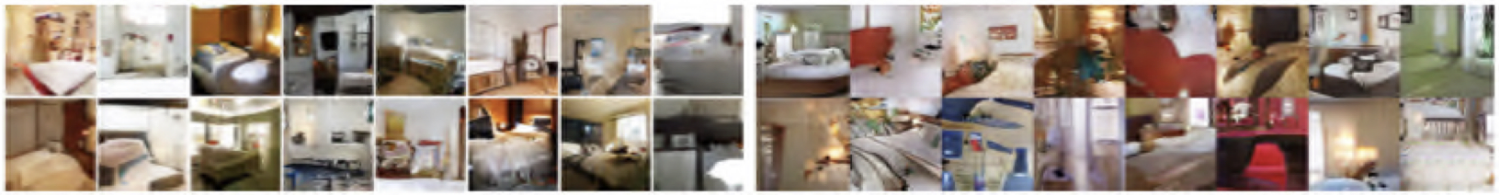


Wasserstein GAN: Experimental Result

- Robustness

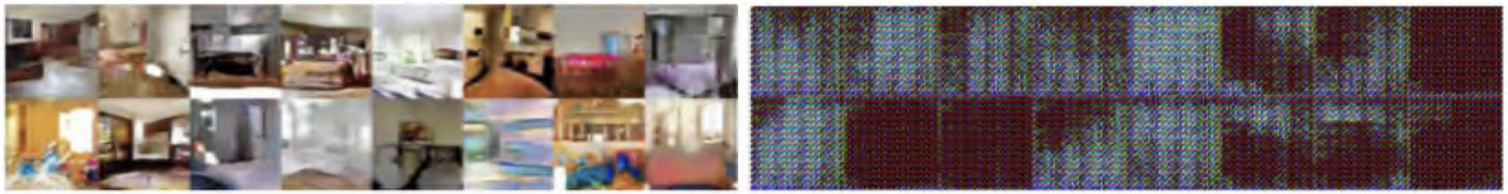
DCGAN Generator:

W: JS:



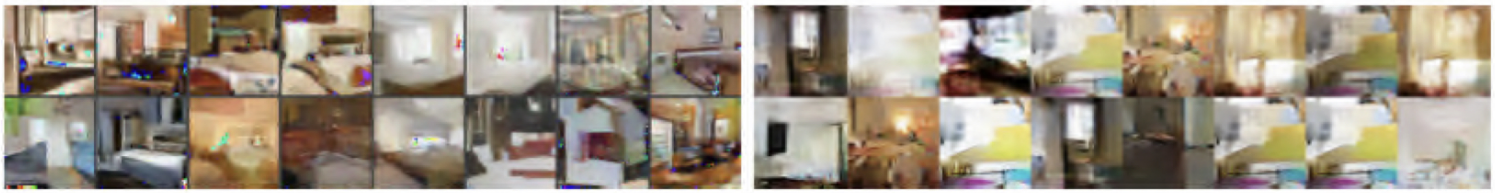
DCGAN Generator without Batch Normalization:

W: JS:



MLP Generator:

W: JS: (Mode Collapse)



Outline

- Generative Models
- VAE
- GAN
- Similarities between VAE and GAN
- Drawbacks of VAE and GAN
- Solutions by Wasserstein Metric
  - Wasserstein GAN
  - **Wasserstein AE**[2]
- Unified Theory and Stronger Model
- Further Problems

[2] Tolstikhin I., et al. Wasserstein Auto-Encoders. 2017.

Wasserstein Distance: Latent Variable Version

- What we need
  1. Deterministic decoder & Proper metric
  2. Stronger regularizer
- Recall Kantorovich's formula:
  $W_c(X, Y) = \sup_{\psi(y)+\phi(x)\leqslant c(x,y)} [\mathbb{E}_{P_X}\psi(y) + \mathbb{E}_{P_G}\phi(x)]$
- Formula with latent variable

Theorem 4

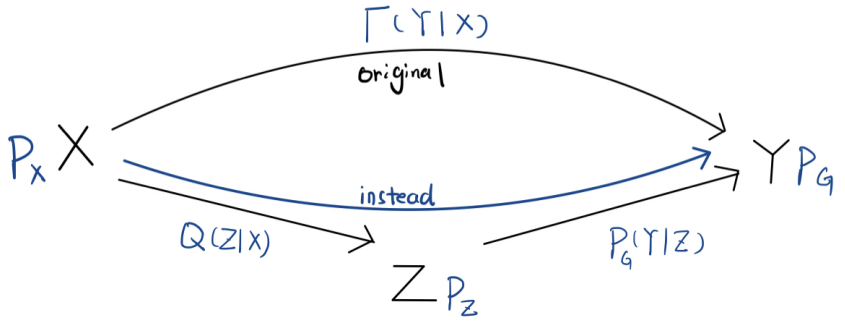
For $\mathbb{P}_G$ as defined above with deterministic $P_G(X|Z)$ and any function $G : \mathcal{Z} \to \mathcal{X}$

$$\inf_{\gamma \in P(P_X, P_G)} \mathbb{E}_{\Gamma(X,Y)}[c(x, y)] = \inf_{Q: Q_z = P_Z} \mathbb{E}_{P_X}\mathbb{E}_{Q(Z|X)}[c(X, G(Z))]$$

Why can we expect such a formula?

Wasserstein AE: Motivation

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



Remember what we need:
1. AE structure
2. Prior P(Z)
3. Deterministic decoder

- Derivation
  $\mathbb{E}_{\Gamma(X,Y)}[c(X,Y)] = \mathbb{E}_{P_X}\mathbb{E}_{\Gamma(Y|X)}[c(X,Y)] =$
  $\mathbb{E}_{P_X}\mathbb{E}_{Q(Z|X)}\mathbb{E}_{P_G(Y|Z)}[c(X,Y)] = \mathbb{E}_{P_X}\mathbb{E}_{Q(Z|X)}[c(X,G(Z))]$
- Constraint
  $\left\{ Q(Z|X) : P_X \to P_Z \right\} \Rightarrow Q_Z = \mathbb{E}_{P_X} Q(Z|X) = P_Z$
  (*Marginal Distribution Constraint*)
- Generating Data
  Similar to Generator in GAN : $G(Z) = P(X|Z)$,
  therefore P_Z *should be controlled*

Wasserstein AE: Relaxation & AE Structure

- Relaxation fomulation
  From restriction term

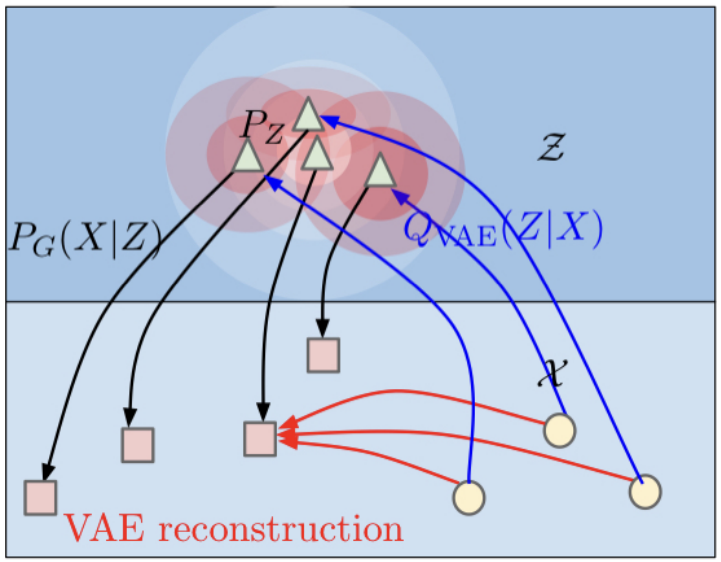
$$\inf_{Q:Q_Z=P_Z} \mathbb{E}_{P_X}\mathbb{E}_{Q(Z|X)}[c(X, G(Z))]$$

  to penalty term

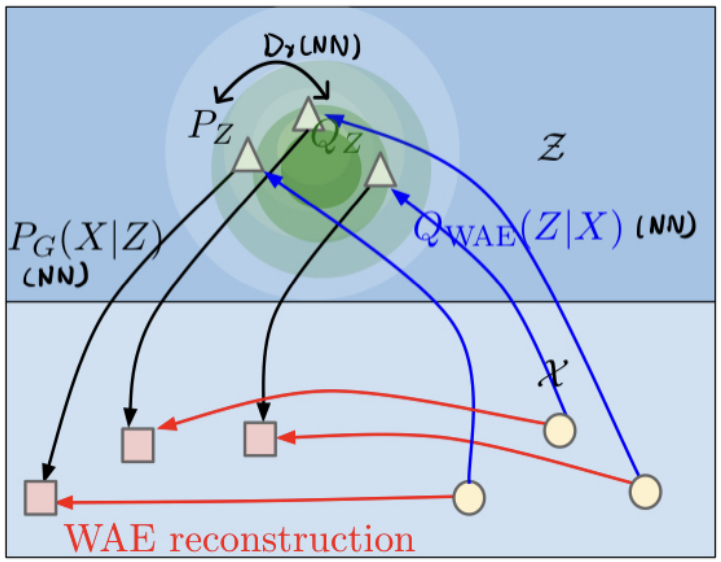
$$D_{WAE}(P_X, P_G) = \inf_{Q(Z|X)\in Q} \mathbb{E}_{P_X}\mathbb{E}_{Q(Z|X)}[c(X, G(Z))] + \lambda D_Z(Q_Z, P_Z)$$

- AE Structure

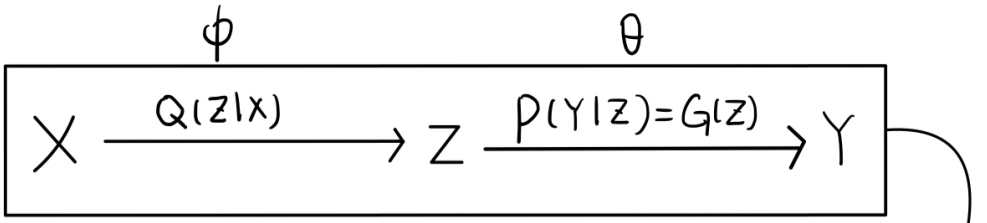
(a) VAE

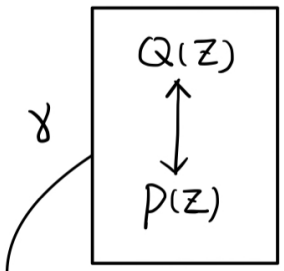


(b) WAE



Wasserstein AE: Penalties & Optimization





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

$$GAN-based:\ D_Z(P_Z, Q_Z) = \mathbb{E}_{z\sim P_Z} log D(z) + \mathbb{E}_{\tilde{z}\sim Q_Z} log(1 - D(\tilde{z}))$$

$$MMD-based:\ D_Z(P_Z, Q_Z) = \|\mathbb{E}_{P_Z} k(z, \cdot) - \mathbb{E}_{Q_Z} k(z, \cdot)\|_{\mathcal{H}_k}$$

Wasserstein AE: Algorithm

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 $\{x_1, \dots, x_n\}$ from the training set

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

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

Update D_γ by ascending:

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

Update Q_ϕ and G_θ by descending:

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

end while

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 $\{x_1, \dots, x_n\}$ from the training set

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

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

Update Q_ϕ and G_θ by descending:

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

end while

Wasserstein AE: Experimental Result

Encoder architecture:

$$\begin{aligned} x \in \mathcal{R}^{64\times64\times3} &\to \text{Conv}_{128} \to \text{BN} \to \text{ReLU} \\ &\to \text{Conv}_{256} \to \text{BN} \to \text{ReLU} \\ &\to \text{Conv}_{512} \to \text{BN} \to \text{ReLU} \\ &\to \text{Conv}_{1024} \to \text{BN} \to \text{ReLU} \to \text{FC}_{64} \end{aligned}$$

Decoder architecture:

$$\begin{aligned} z \in \mathcal{R}^{64} &\to \text{FC}_{8\times8\times1024} \\ &\to \text{FSConv}_{512} \to \text{BN} \to \text{ReLU} \\ &\to \text{FSConv}_{256} \to \text{BN} \to \text{ReLU} \\ &\to \text{FSConv}_{128} \to \text{BN} \to \text{ReLU} \to \text{FSConv}_{1} \end{aligned}$$

Adversary architecture for WAE-GAN:

$$\begin{aligned} z \in \mathcal{R}^{64} &\to \text{FC}_{512} \to \text{ReLU} \\ &\to \text{FC}_{512} \to \text{ReLU} \\ &\to \text{FC}_{512} \to \text{ReLU} \\ &\to \text{FC}_{512} \to \text{ReLU} \to \text{FC}_{1} \end{aligned}$$

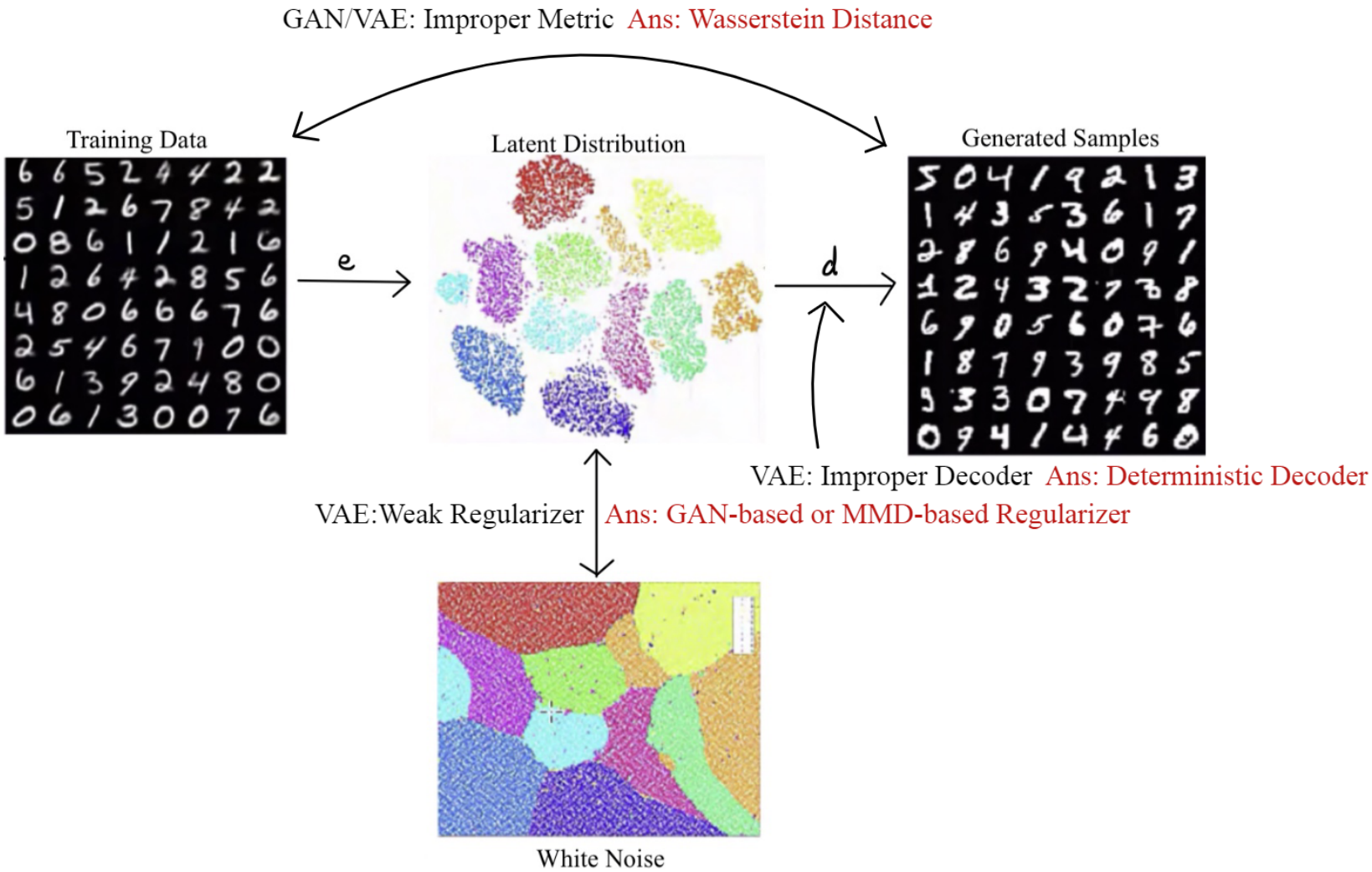
Wasserstein AE: Experimental Result

More close to data manifold

Algorithm	FID	Sharpness
VAE	63	3×10^{-3}
WAE-MMD	55	6×10^{-3}
WAE-GAN	42	6×10^{-3}
bigVAE	45	—
bigWAE-MMD	37	—
bigWAE-GAN	35	—
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.

Review of the Solutions



Outline

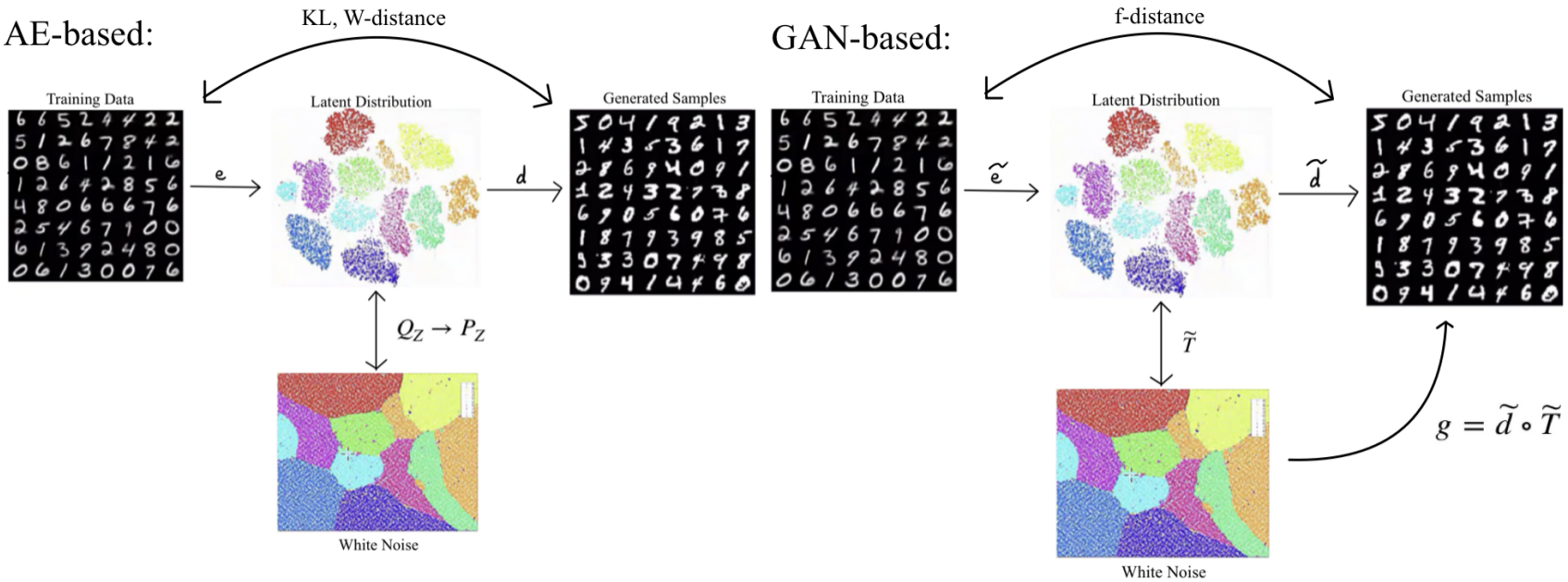
- Generative Models
- VAE
- GAN
- Similarities between VAE and GAN
- Drawbacks of VAE and GAN
- Solutions by Wasserstein Metric
  - Wasserstein GAN
  - Wasserstein AE
- **Unified Theory and Stronger Model**
- Further Problems

Unified Structure

Generative models mainly accomplish two tasks:

1. Manifold Learning: Dimension reduction and retaining the information of data distribution
2. Distribution Transportation (Measure Learning): Learn data distribution in low dimension

Key: Different models accomplish tasks in different ways:



Observation: both of them mix the two processes.

Problem Remained

1. What are the reasons for model collapse & mixture in GAN based models?
2. What are the reasons for relatively low quality of generated data in AE based models?

	CIFAR10		CelebA	
	Standard	ResNet	Standard	ResNet
WGAN-GP [14]	40.2	19.6	21.2	18.4
PGGAN [38]	-	18.8	-	16.3
SNGAN [32]	25.5	21.7	-	-
WGAN-div [20]	-	18.1	17.5	15.2
WGAN-QC [29]	-	-	-	12.9
AE-OT [2]	34.2	28.5	24.3	28.6
AE-OT-GAN	25.2	17.1	11.2	7.8

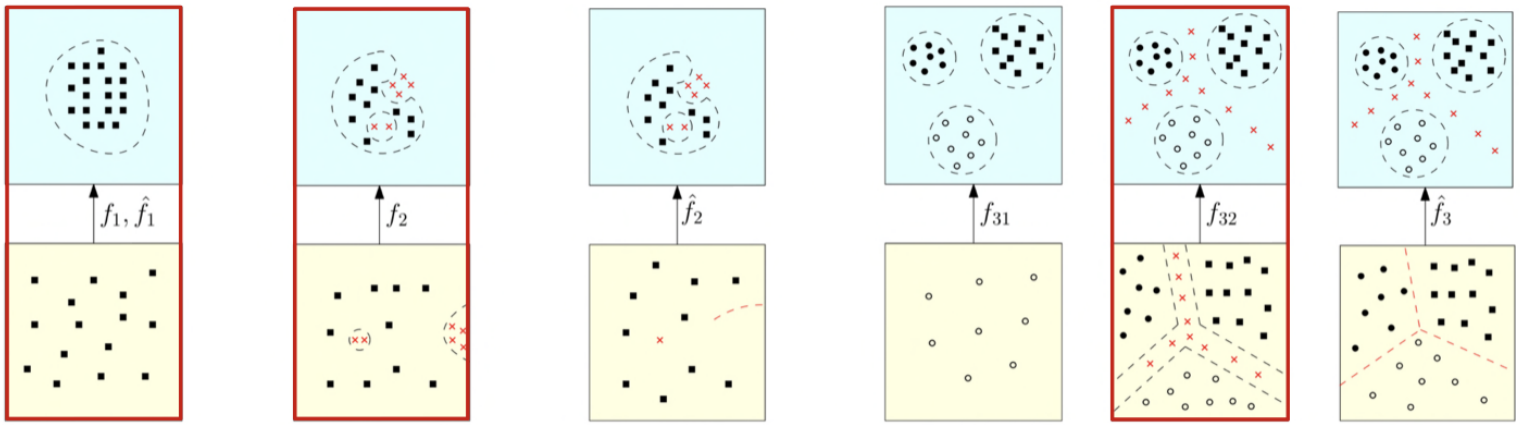
The comparison of FID between the proposed method and the state of the arts on Cifar10 and CelebA.

Algorithm	**FID**	**Sharpness**
VAE	63	3×10^{-3}
WAE-MMD	55	6×10^{-3}
WAE-GAN	42	6×10^{-3}
bigVAE	45	—
bigWAE-MMD	37	—
bigWAE-GAN	35	—
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.

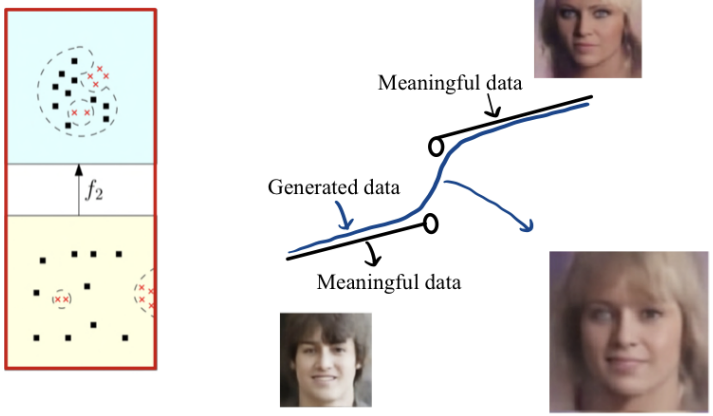
Reasons for Failure in GAN Based Models

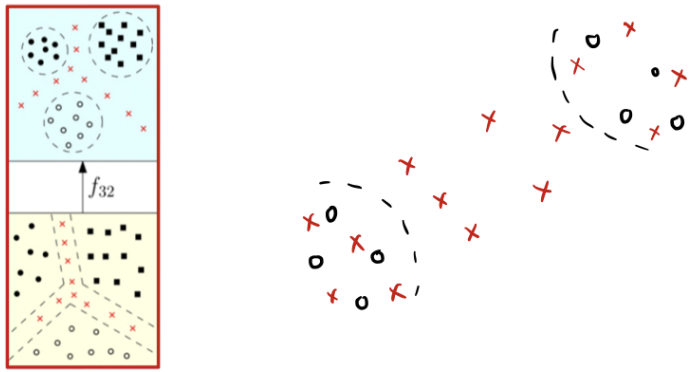
- Singular Point[3]



(a) convex support (b) concave support: single mode (c) concave support: multi modes

Singular point leads to discontinuity, while NN learns continuous map.

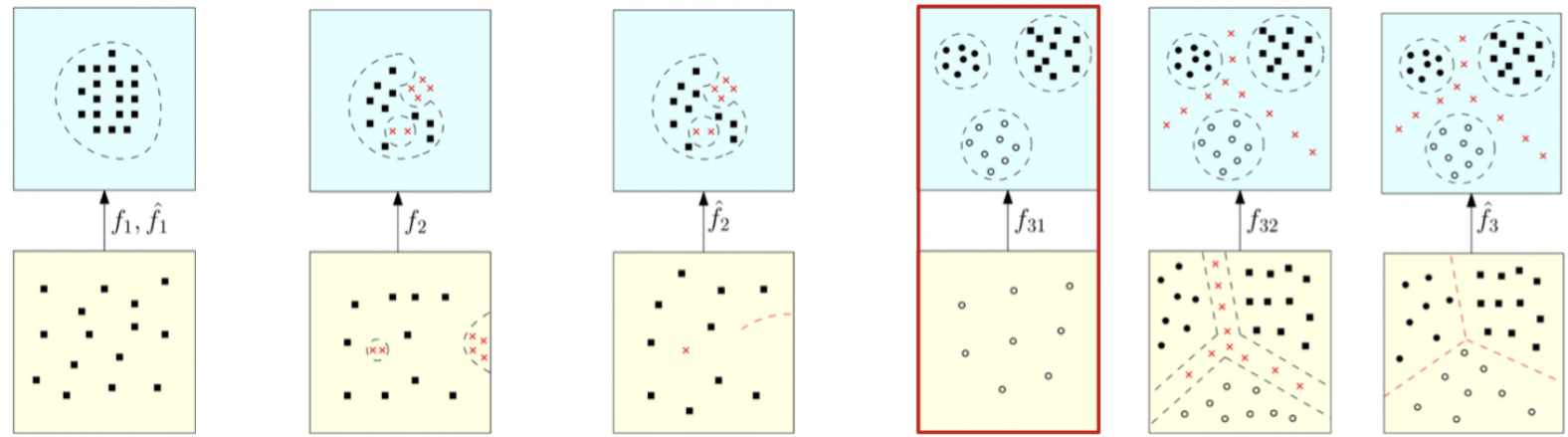




[3] Lei N., et al. Geometric Understanding of Deep Learning. 2018.

Reasons for Failure in GAN Based Models

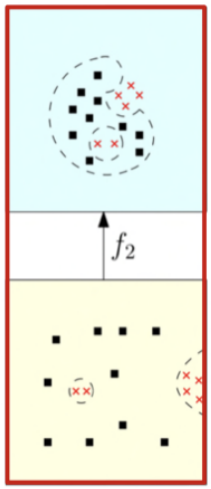
- Local Measure



convex support (b) concave support: single mode (c) concave support: multi modes

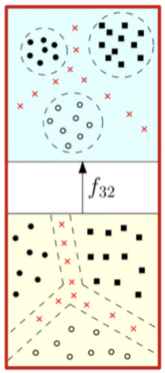
Reasons for Failure in GAN Based Models

- Single Mode Case

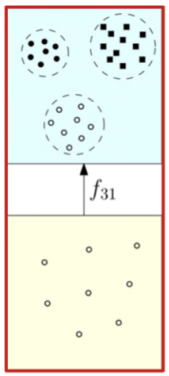


GAN/WGAN: Approximate a discontinuous map by NN

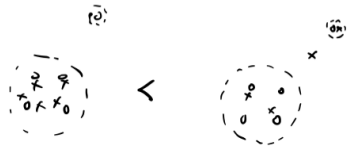
- Multi Mode Case



GAN/WGAN: Same as above

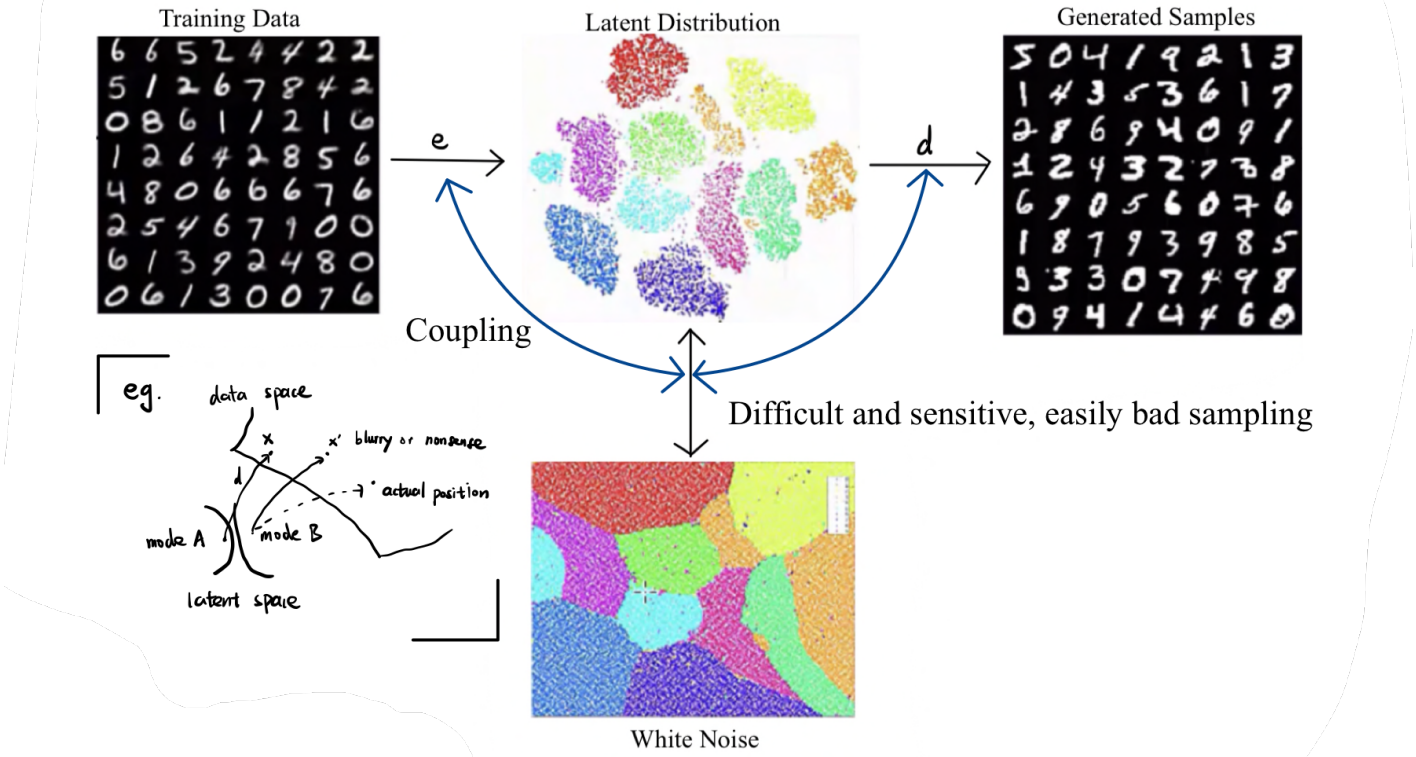


GAN: Local measure
WGAN: Average measure(far, small mode)



Reasons for Failure in AE Based Models

- Trade-off between nice latent manifold structure and quality of coding

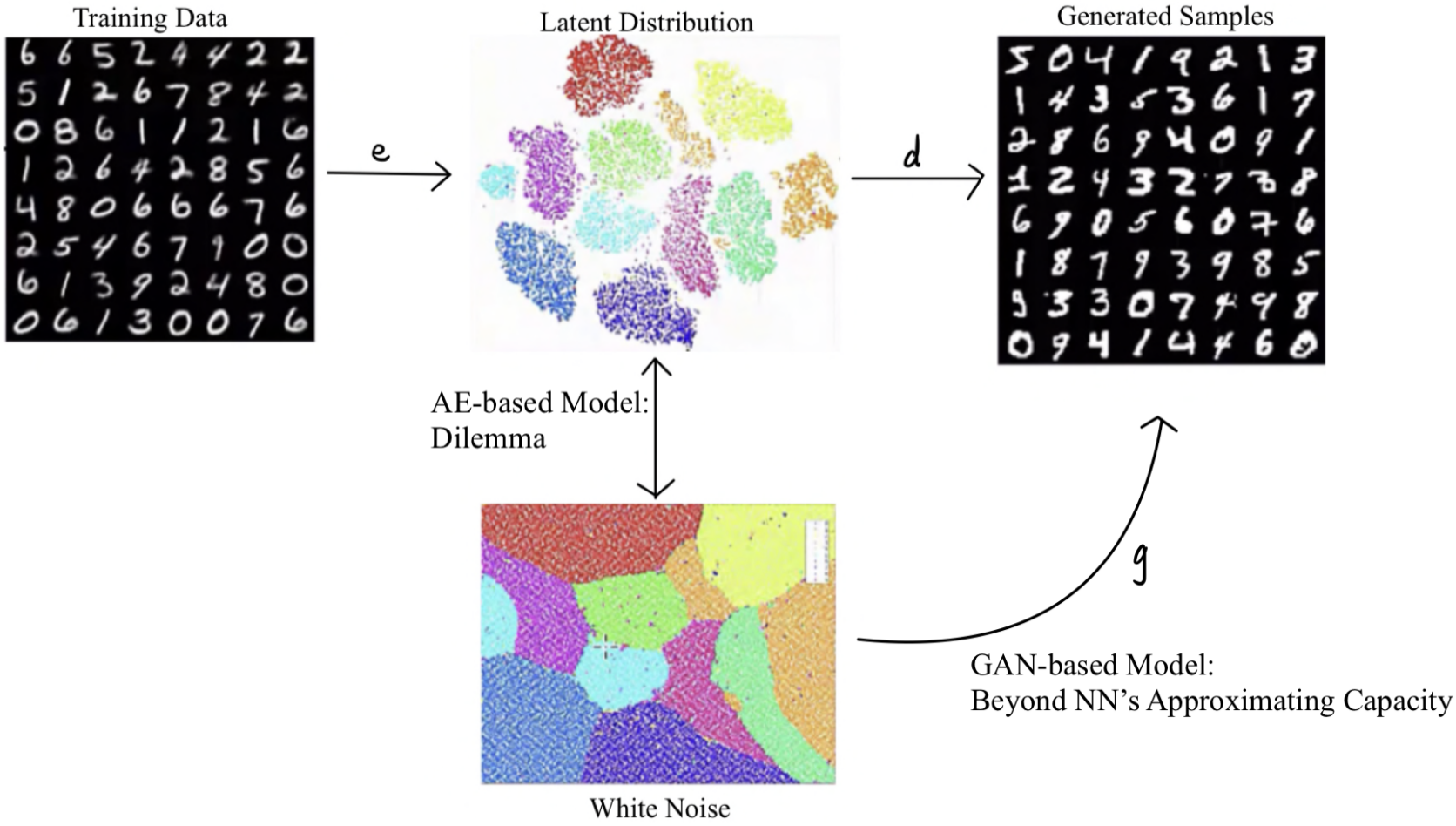


Problems of matching Q to prior P:

1. Difficult and sensitive, generating bad samples (too weak case)
2. Hinders the quality of coding (too strong case)

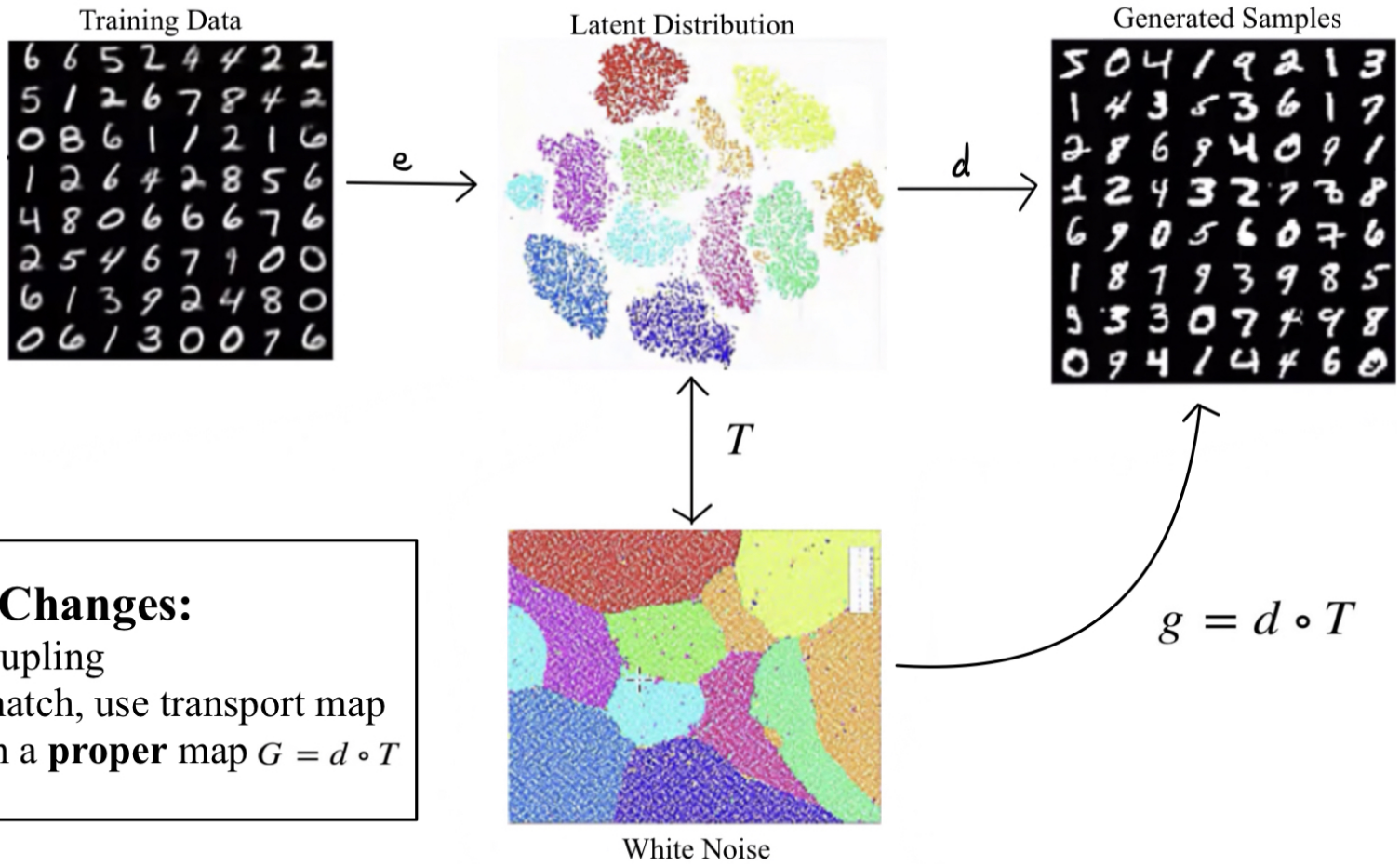
Dilemma: Match or Not Match?

Review of Problems Remained



Inspiration: At least we should not mix the two processes, i.e. manifold & measure learning.

Solution: A New Structure



Main Changes:
1. Decoupling
2. No match, use transport map
3. Learn a **proper** map $G = d \circ T$

Key: learn the proper manifold map d and measure transport map T respectively.

Measure Learning by OT

- Brenier Potential and OT[4]

Theorem 5

Suppose X and Y are the Euclidean space $\mathbb{R}^n$, and the transportation cost is the quadratic Euclidean distance $c(x, y) = |x - y|^2$. If μ is absolutely continuous and μ and ν have finite second order moment, then there exists a convex function $u : X \to \mathbb{R}$, its gradient map ∇u gives the Solution to the Monge's problem, where u is called Brenier's potential. Furthermore, the optimal mass transportation map is unique:

$$T = \nabla u$$

While T is discontinue, we can learn u instead, which is continue (convex) This is why we need T to be a OPTIMAL transport map, rather than ordinary transport map.

[4]Gu X., et al. Variational Principles for Minkowski Type Problems, Discrete Optimal Transport, and Discrete Monge-Ampere Equations. Mathematical Methods in Solid State & Superfluid Theory, 2013.

Measure Learning by OT

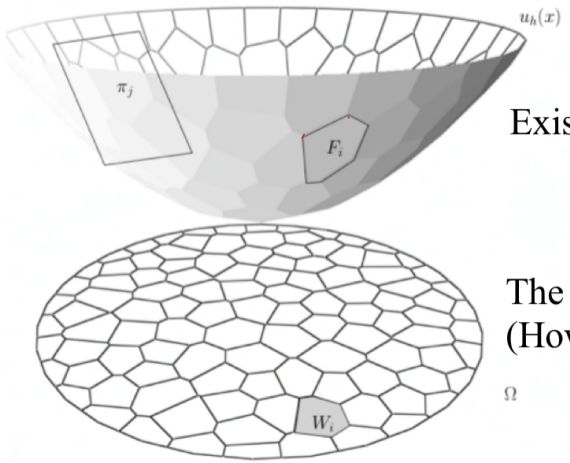
- Question: How to learn u?
  Analysis by geometic method provides some necessary conditions:
  - Locally: $\nabla u_h^i(x) = T(x) = y_i \Rightarrow u_h^i(x) =< x, y_i > + h_i, x \in \mathbb{R}^n$
  - Globally: u convex $\Rightarrow u(x) = \max_i\{u_h^i(x)\} = max_i< x, y_i > + h_i, x \in \mathbb{R}^n$
  - Measure Preserving
    $\Rightarrow w_i = v_i, W_i = \{x \in \mathbb{R}^n | u_h^i(x) = y_i\}, w_i = \mu_X(W_i \cap \Omega), \nu_i = \mu_Y(y_i)$

Measure Learning by OT

- Learning Brenier Potential by Convex Geometry Theory

Theorem 6 (the shape of u)

Suppose Ω is a compact convex polytope with non-empty interior in $\mathbb{R}^n$, $n_1, \cdots, n_k \subset \mathbb{R}^{n+1}$ are distinct k unit vectors, the $(n+1)$-th coordinates are negative, and $\nu_1, \cdots, \nu_k \geqslant 0$ so that $\sum_{i=1}^{k} \nu_i = vol(\Omega)$. Then there exists convex polytope $P \subset \mathbb{R}^{n+1}$ with exact k codimension-1 faces $F_1, \cdots, F_k$ so that n_i is the normal vector to F_i and the intersection between Ω and the projection of F_i is with volume v_i. Furthermore, such P is unique up to vertical translation.



Exist and unique, u is exactly the Brenier potential

The OT problem is transformed into a convex geometry problem (How to learn h?)

Measure Learning by OT

- Learning Brenier Potential by Convex Geometry Theory

Theorem 7 (compute u)

Let Ω be a compact convex domain in $\mathbb{R}^n$, $\{y_1, \cdots, y_k\}$ be a set of distinct points in $\mathbb{R}^n$ and μ a probability measure on Ω. Then for any $v_1, \cdots, v_k \geqslant 0$ with $\sum_{i=1}^k v_i = vol(\Omega)$, there exists $h = (h_1, \cdots, h_k) \in \mathbb{R}^k$, unique up to adding a constant $(c, \cdots, c)$, so that $w_i(h) = v_i$, for all i. The vectors h are exactly maximum points of the concave function

$$E(h) = \sum_{i=1}^{k} h_i v_i - \int_0^h \sum_{i=1}^{k} w_i(\eta) d\eta_i$$

on the open convex set $H = \{h \in \mathbb{R}^k | w_i(h) \geqslant 0\}$. Furthermore, ∇u_h minimizes the quadratic cost $\int_\Omega |x - T(x)|^2 d\mu(x)$ among all transport maps $T_{\#}\mu = \nu$, where the Dirac measure $\nu = \sum_{i=1}^k v_i \delta_{y_i}$.

Measure Learning by OT

- Learning Brenier Potential by Convex Geometry Theory
  The optimization is a convex problem.
  Jacobian:

$$\nabla E(h) = (\nu_1 - w_1(h), \nu_2 - w_2(h), \cdots, \nu_k - w_k(h))^T$$

  Hessian:

$$\frac{\partial^2 E(h)}{\partial h_i^2} = \frac{\partial w_i(h)}{\partial h_i} = \sum_{j \neq i} \frac{\partial w_i(h)}{\partial h_j} = -\frac{\mu(D_{ij})}{|e_{ij}|}$$

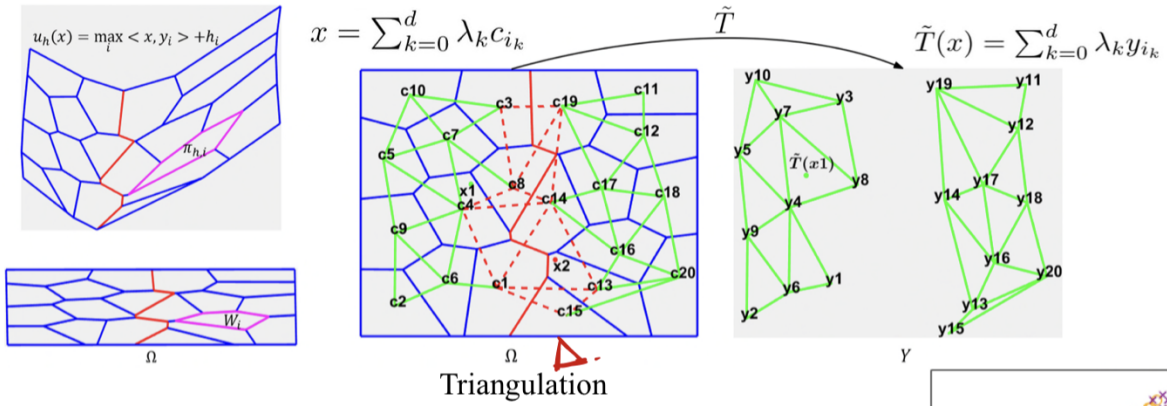
$$\frac{\partial^2 E(h)}{\partial h_i^2} = \frac{\partial w_i(h)}{\partial h_i} = \sum_{j \neq i} \frac{\partial w_i(h)}{\partial h_j}$$

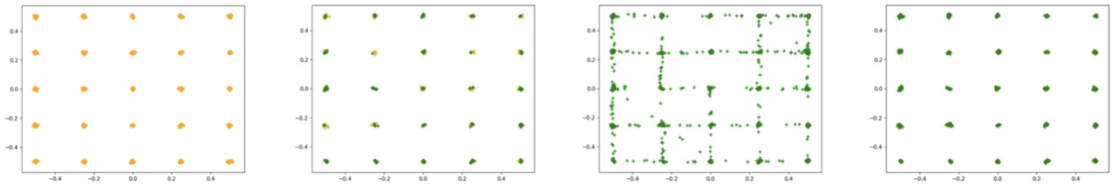
  Optimized by the Newton Method:

$$x_{n+1} = x_n - [Hf(x_n)]^{-1} \nabla f(x_n), n \geqslant 0$$

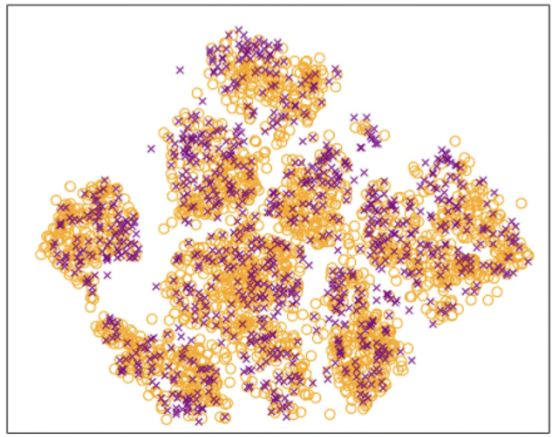
Measure Learning by OT

- Singularity Test and Extension of SDOT[5]



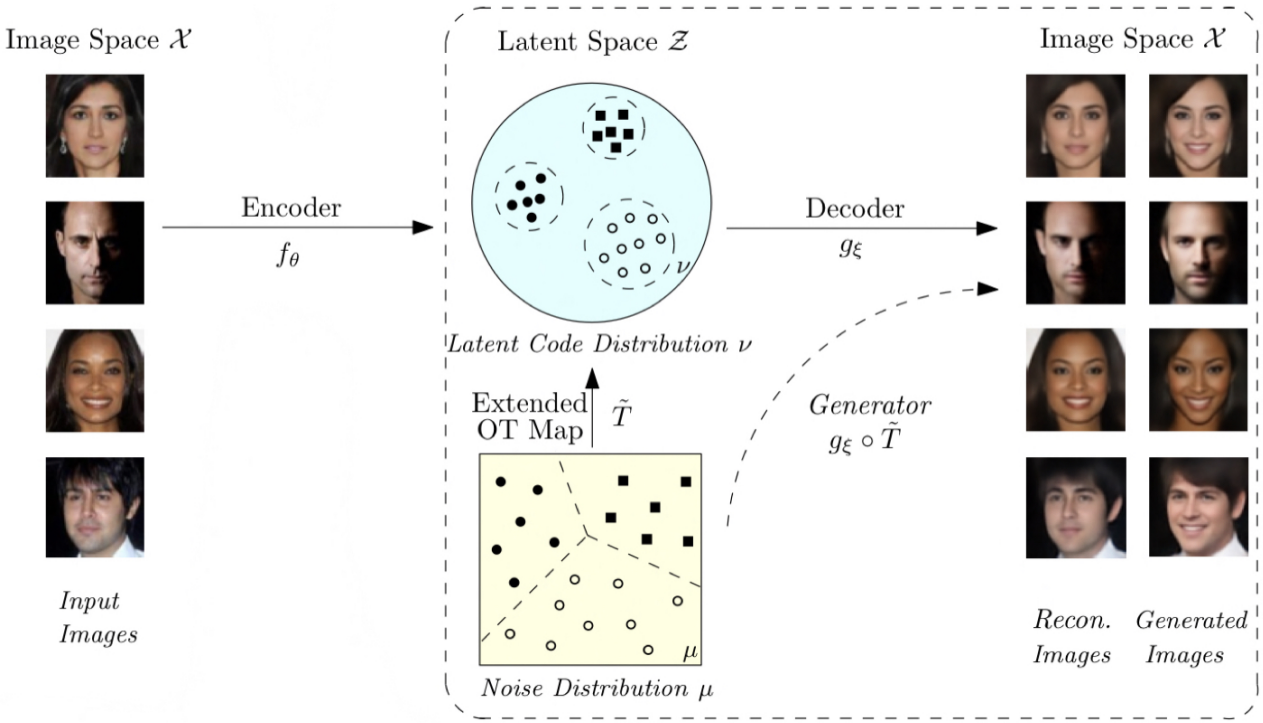


(a) Target Distribution (b) $\hat{\theta}$ too small (c) $\hat{\theta}$ too large (d) Proper $\hat{\theta}$



[5] An D., et al. AE-OT-GAN: Training GANs from data specific latent distribution. 2020.

AE-OT: Theory



Algorithm 1 Semi-discrete OT Map

Input: Latent codes $Y = \{y_i\}_{i \in \mathcal{I}}$, empirical latent code distribution $\nu = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \delta_{y_i}$, number of Monte Carlo samples N, positive integer s.
Output: Optimal transport map $T(\cdot)$.
Initialize $h = (h_1, h_2, \ldots, h_{|\mathcal{I}|}) \leftarrow (0, 0, \ldots, 0)$.
repeat
  Generate N uniformly distributed samples $\{x_j\}_{j=1}^N$.
  Calculate $\nabla h = (\hat{w}_i(h) - \nu_i)^T$.
  $\nabla h = \nabla h - mean(\nabla h)$.
  Update h by Adam algorithm with $\beta_1 = 0.9, \beta_2 = 0.5$.
  if $E(h)$ has not decreased for s steps **then**
    $N \leftarrow N \times 2$.
  end if
until Converge
OT map $T(\cdot) \leftarrow \nabla(\max_i \langle \cdot, y_i \rangle + h_i)$.

Algorithm 2 Generate latent code

Input: Optimal transport map $T(\cdot)$, number of samples to generate n, angle threshold $\hat{\theta}$.
Output: Generated latent code P.
Compute $\hat{c}_i$ by Monte Carlo method.
repeat
  Sample $x \sim \mu$, Find the smallest $d+1$ vertex around x as $\{d(x, \hat{c}_{i_0}), d(x, \hat{c}_{i_1}), \ldots, d(x, \hat{c}_{i_d})\}$.
  Compute dihedral angles θ_{i_k} between π_{i_0} and π_{i_k}.
  Select θ_{i_k} with $\theta_{i_k} \leq \hat{\theta}$, result in $\hat{i}_k = 0, 1, \ldots, d_1$.
  if $\forall k, \theta_{i_k} > \hat{\theta}$ **then** Abandon x
  else Generate latent code $\tilde{T}(x) = \sum_{k=0}^{d_1} \lambda_k T(\hat{c}_{\hat{i}_k})$ with $\lambda_k = d^{-1}(x, \hat{c}_{\hat{i}_k}) / \sum_{j=0}^{d_1} d^{-1}(x, \hat{c}_{\hat{i}_j})$.
  end if
until Generate n new latent code

AE-OT: Result

- MM & MC are solved

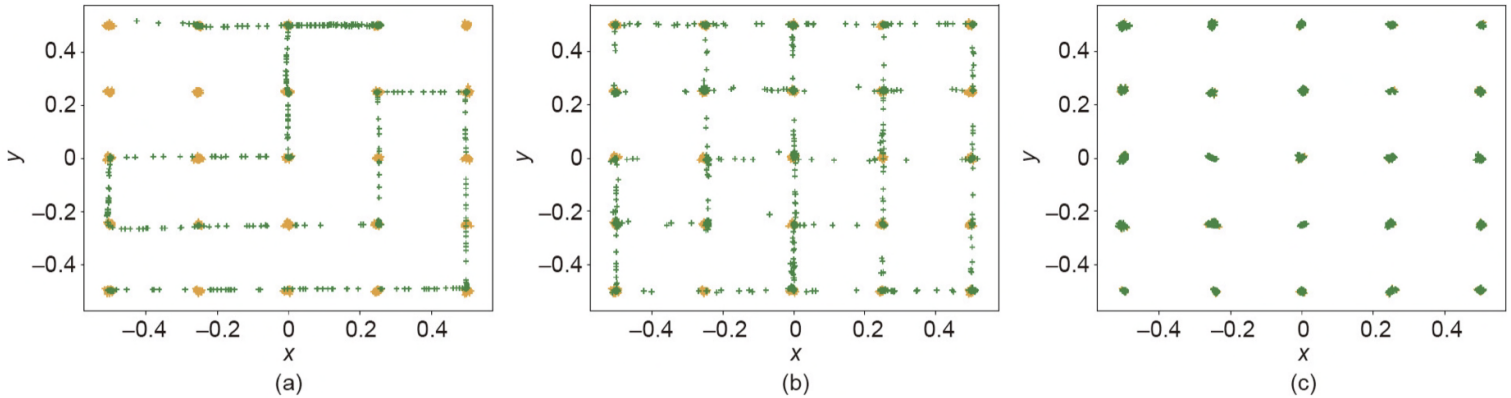


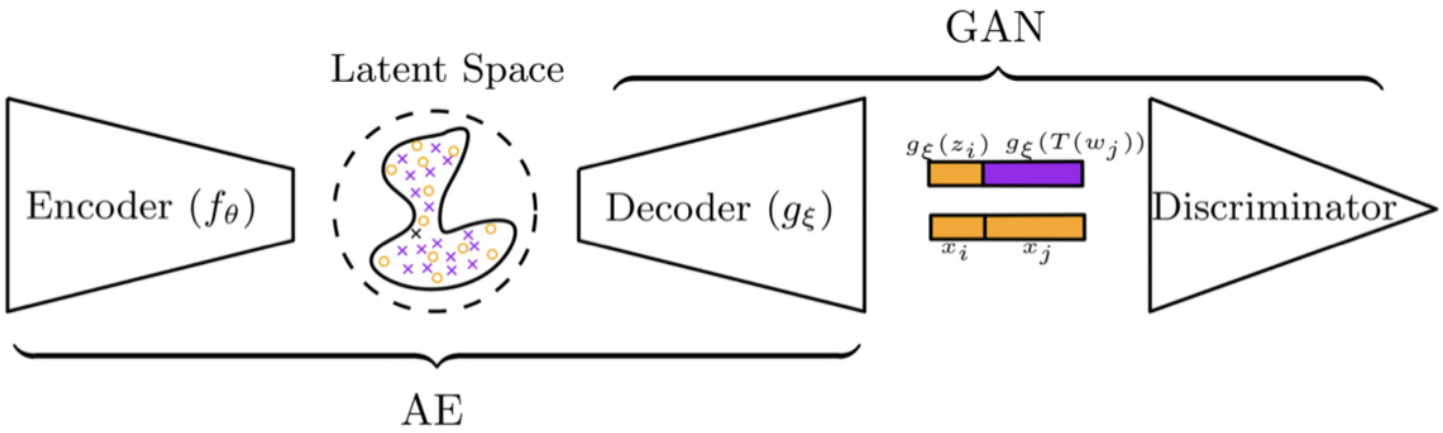
Fig. 13. Mode collapse comparison on a 2D grid dataset. (a) GAN; (b) PacGAN4; (c) AE–OT. Orange marks are real samples and green marks are generated ones.

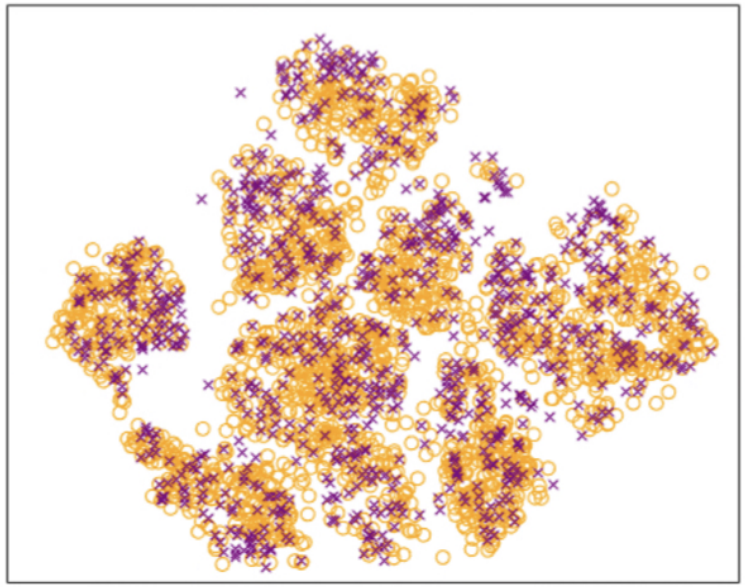
- Generated data quality

Table 2: Quantitative comparison with FID

	Adversarial				Non-Adversarial		Reference	
Dataset	NS GAN	LSGAN	WGAN	BEGAN	VAE	GLANN	AE	Ours
MNIST	6.8±0.5	7.8±0.6	6.7±0.4	13.1±1.0	23.8±0.6	8.6±0.1	5.5	**6.2±0.2**
Fansion	26.5±1.6	30.7±2.2	21.5±1.6	22.9±0.9	58.7±1.2	13.0±0.1	4.7	**10.1±0.3**
CIFAR-10	58.5±1.9	87.1±47.5	55.2±2.3	71.4±1.6	65.4±0.2	46.5±0.2	28.2	**38.3±0.5**
CelebA	55.0±3.3	53.9±2.8	41.3±2.0	**38.9±0.9**	85.7±3.8	46.3±0.1	67.5	68.4± 0.5

AE-OT-GAN: Theory





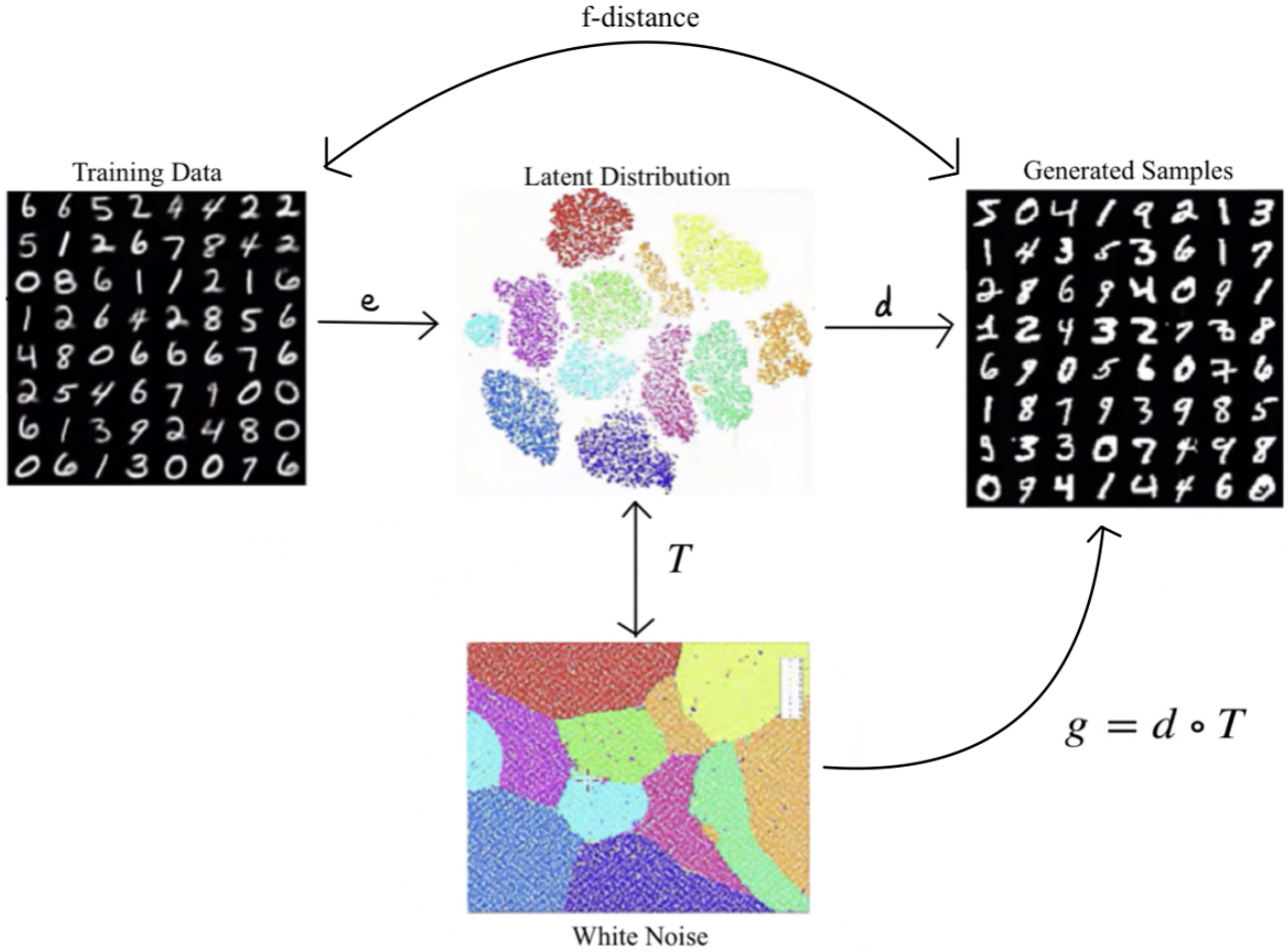
AE-OT-GAN: Experimental Result

	CIFAR10		CelebA	
	Standard	ResNet	Standard	ResNet
WGAN-GP [14]	40.2	19.6	21.2	18.4
PGGAN [38]	-	18.8	-	16.3
SNGAN [32]	25.5	21.7	-	-
WGAN-div [20]	-	18.1	17.5	15.2
WGAN-QC [29]	-	-	-	12.9
AE-OT [2]	34.2	28.5	24.3	28.6
AE-OT-GAN	25.2	17.1	11.2	7.8

Table 1. The comparison of FID between the proposed method and the state of the arts on Cifar10 and CelebA.

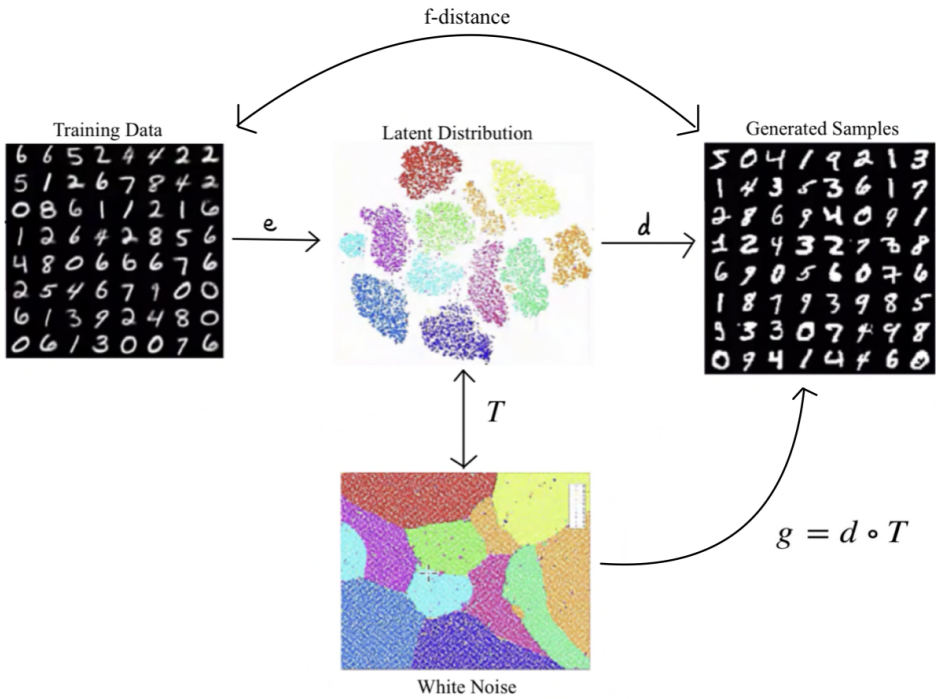
Conclusion and Further Problems

- Decoupling Method
  1. Manifold Learning process remains uncertain.
  2. The computation complexity of OT map.



Conclusion and Further Problems

- Coupling Method
  1. Mode Collapse & Mode Mixture
  2. Is W-distance really useful?[6]



[6] Stanczuk J., et al. Wasserstein GANs Work Because They Fail (to Approximate the Wasserstein Distance). 2021.