On the Latent Space of Deep Generative Models

Honors Thesis Presentation

Naren Manoj

August 2018

Outline

Outline

Deep Generative Models

- "[Generative Adversarial Networks], and the variations that are now being proposed is the most interesting idea in the last 10 years in ML, in my opinion." - Yann LeCun (courtesy of Quora post)

Deep Generative Models

- "[Generative Adversarial Networks], and the variations that are now being proposed is the most interesting idea in the last 10 years in ML, in my opinion." - Yann LeCun (courtesy of Quora post)
- Sample noise from distribution (draw a latent vector), pass through a function (typically a neural net), and get meaningful output (some image)

Variational Autoencoders

Consists of:

Variational Autoencoders

Consists of:

- *Encoder* mapping images to compressed representations (*latent vectors*)

Variational Autoencoders

Consists of:

- *Encoder* mapping images to compressed representations (*latent vectors*)
- *Decoder* (or *Generator*) mapping latent vectors to image

Variational Autoencoders

Consists of:

- *Encoder* mapping images to compressed representations (*latent vectors*)
- *Decoder* (or *Generator*) mapping latent vectors to image

Minimize two error terms:

Variational Autoencoders

Consists of:

- *Encoder* mapping images to compressed representations (*latent vectors*)
- *Decoder* (or *Generator*) mapping latent vectors to image

Minimize two error terms:

- Discrepancy between distribution of encoder outputs and some desired distribution over latent vectors

Variational Autoencoders

Consists of:

- *Encoder* mapping images to compressed representations (*latent vectors*)
- *Decoder* (or *Generator*) mapping latent vectors to image

Minimize two error terms:

- Discrepancy between distribution of encoder outputs and some desired distribution over latent vectors
  - For instance, we could try forcing our latent vectors to follow a Gaussian
  - Use KL divergence as discrepancy

Variational Autoencoders

Consists of:

- *Encoder* mapping images to compressed representations (*latent vectors*)
- *Decoder* (or *Generator*) mapping latent vectors to image

Minimize two error terms:

- Discrepancy between distribution of encoder outputs and some desired distribution over latent vectors
  - For instance, we could try forcing our latent vectors to follow a Gaussian
  - Use KL divergence as discrepancy
- Reconstruction error

Variational Autoencoders

Consists of:

- *Encoder* mapping images to compressed representations (*latent vectors*)
- *Decoder* (or *Generator*) mapping latent vectors to image

Minimize two error terms:

- Discrepancy between distribution of encoder outputs and some desired distribution over latent vectors
  - For instance, we could try forcing our latent vectors to follow a Gaussian
  - Use KL divergence as discrepancy
- Reconstruction error
  - On average, how lossy is the compression that the VAE is performing?

Variational Autoencoders

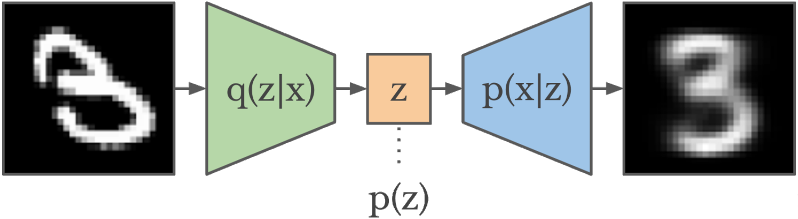
Consists of:

- *Encoder* mapping images to compressed representations (*latent vectors*)
- *Decoder* (or *Generator*) mapping latent vectors to image

Minimize two error terms:

- Discrepancy between distribution of encoder outputs and some desired distribution over latent vectors
  - For instance, we could try forcing our latent vectors to follow a Gaussian
  - Use KL divergence as discrepancy
- Reconstruction error
  - On average, how lossy is the compression that the VAE is performing?
  - Use ℓ_2 distance between two images

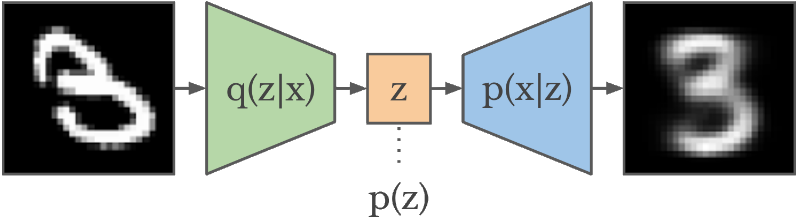
VAE Architecture



[1]

[1] Diagram courtesy of Danijar Hafner.

VAE Architecture

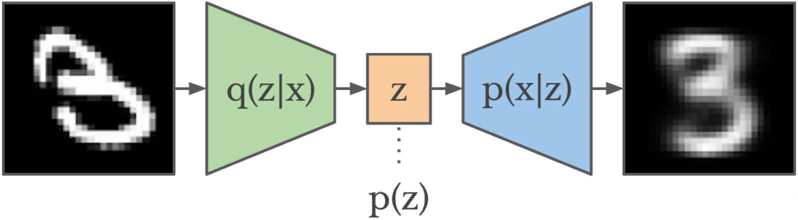


[1]

- $q(z|x)$ is distribution of encoder outputs

[1]Diagram courtesy of Danijar Hafner.

VAE Architecture

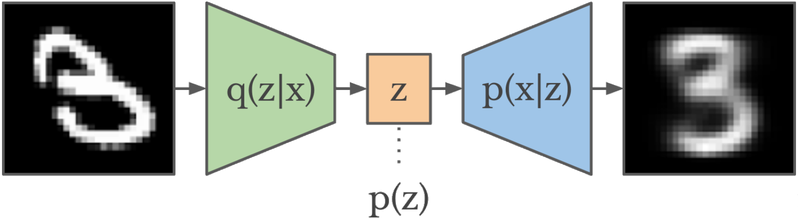


[1]

- $q(z|x)$ is distribution of encoder outputs
- $z \sim p(z)$ is latent variable

[1] Diagram courtesy of Danijar Hafner.

VAE Architecture



[1]

- $q(z|x)$ is distribution of encoder outputs
- $z \sim p(z)$ is latent variable
- $p(x|z)$ is distribution of decoder outputs

[1] Diagram courtesy of Danijar Hafner.

Generative Adversarial Networks

Consists of:

Generative Adversarial Networks

Consists of:

- *Discriminator* answering whether an input is real (from the data distribution) or fake (from somewhere else)

Generative Adversarial Networks

Consists of:

- *Discriminator* answering whether an input is real (from the data distribution) or fake (from somewhere else)
- *Generator* generating meaningful outputs from noise sampled from some distribution

Generative Adversarial Networks

Consists of:

- *Discriminator* answering whether an input is real (from the data distribution) or fake (from somewhere else)
- *Generator* generating meaningful outputs from noise sampled from some distribution

Trained via *adversarial training*:

Generative Adversarial Networks

Consists of:

- *Discriminator* answering whether an input is real (from the data distribution) or fake (from somewhere else)
- *Generator* generating meaningful outputs from noise sampled from some distribution

Trained via *adversarial training*:

- Generator tries to fool discriminator by producing realistic outputs

Generative Adversarial Networks

Consists of:

- *Discriminator* answering whether an input is real (from the data distribution) or fake (from somewhere else)
- *Generator* generating meaningful outputs from noise sampled from some distribution

Trained via *adversarial training*:

- Generator tries to fool discriminator by producing realistic outputs
- Discriminator tries to predict as accurately as possible whether an image is real or generated

Generative Adversarial Networks

Consists of:

- *Discriminator* answering whether an input is real (from the data distribution) or fake (from somewhere else)
- *Generator* generating meaningful outputs from noise sampled from some distribution

Trained via *adversarial training*:

- Generator tries to fool discriminator by producing realistic outputs
- Discriminator tries to predict as accurately as possible whether an image is real or generated
- Formulated as min-max game

Problems

GANs:

Problems

GANs:

- Might fail to learn entire modes of the distribution (mode collapse)

Problems

GANs:

- Might fail to learn entire modes of the distribution (mode collapse)
- Suffer from optimization instability

Problems

GANs:

- Might fail to learn entire modes of the distribution (mode collapse)
- Suffer from optimization instability
- ...but generate pretty pictures

Problems

GANs:

- Might fail to learn entire modes of the distribution (mode collapse)
- Suffer from optimization instability
- ...but generate pretty pictures

VAEs:

Problems

GANs:

- Might fail to learn entire modes of the distribution (mode collapse)
- Suffer from optimization instability
- ...but generate pretty pictures

VAEs:

- Don't generate samples as sharp and realistic as those from GANs

Problems

GANs:

- Might fail to learn entire modes of the distribution (mode collapse)
- Suffer from optimization instability
- ...but generate pretty pictures

VAEs:

- Don't generate samples as sharp and realistic as those from GANs
- ...but has a more stable objective

Problems

GANs:

- Might fail to learn entire modes of the distribution (mode collapse)
- Suffer from optimization instability
- ...but generate pretty pictures

VAEs:

- Don't generate samples as sharp and realistic as those from GANs
- ...but has a more stable objective

Generally, theoretical understanding of deep generative models is still lacking.

Outline

Deep Generative Model

Definition (Deep Generative Model)

A *deep generative model* is a (possibly randomized) function, denoted g, that accepts some vector $z \in \mathbb{R}^d$ for some d as input and outputs some image $I \in \mathcal{I}$, where $\mathcal{I}$ represents the space of images.

Encoder

Definition (Encoder)

An *encoder* is a (possibly randomized) function, denoted f, that accepts some input image $I \in \mathcal{I}$ (where, again $\mathcal{I}$ is the space of images) and outputs some vector $z \in \mathbb{R}^d$.

Class

Definition (Class)

Suppose we are given a classifier into K classes $c : \mathcal{I} \to [K]$ that accepts as input an image and outputs a class label which is an integer index in $[K] := \{i\}_{i=1}^{K}$. Then, the *class* of an image I under c is given by:

$$K_{c(I)} := \{z' \mid c\left(g\left(z'\right)\right) = c(I)\}$$

Class

Definition (Class)

Suppose we are given a classifier into K classes $c : \mathcal{I} \to [K]$ that accepts as input an image and outputs a class label which is an integer index in $[K] := \{i\}_{i=1}^{K}$. Then, the *class* of an image I under c is given by:

$$K_{c(I)} := \{z' \mid c(g(z')) = c(I)\}$$

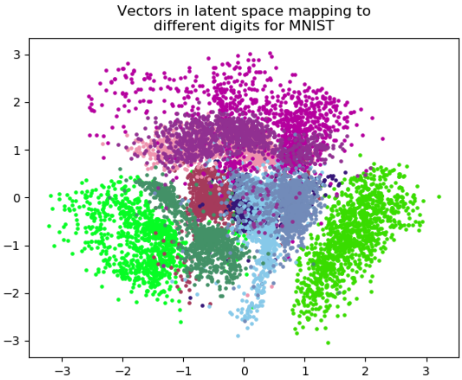
In words, K_i is the subset of the latent space that is classified into label i when a point $z \in K_i$ is passed through g and subsequently classified by c.

Latent Hole

Definition (Latent Hole)

A *hole* is a point $z \in \mathbb{R}^d$ such that there does not exist an image I in the data manifold for which $f(I) = z$.

Examples


Vectors in latent space mapping to
different digits for MNIST
3
2
1
0
−1
−2
−3
−3
−2
−1
0
1
2
3

Class Radius

Definition (Class Radius)

Consider the probability vector $p \in \mathbb{R}^K$ such that:

$$p_i \propto \nu(K_i)$$

Then, the *class radius* is the expected distance from a random point $z \sim p$ in some class to the closest point in a different class. Specifically, define $r(z)$ below, and let $c(g(z)) = i$:

$$r(z) := \inf_{z' \in \bigcup_{j \neq i} K_j} \left\| z - z' \right\|$$

Then the class radius is simply:

$$\mathop{\mathbb{E}}_{z \sim p} [r(z)]$$

Outline

Lack of Density in Latent Space is Bad

Two necessary and related conditions to achieve good coverage of the latent space:

Lack of Density in Latent Space is Bad

Two necessary and related conditions to achieve good coverage of the latent space:

- Most of the latent space (as measured by probability mass) maps to some image in the data manifold via a generator

Lack of Density in Latent Space is Bad

Two necessary and related conditions to achieve good coverage of the latent space:

- Most of the latent space (as measured by probability mass) maps to some image in the data manifold via a generator
  - Informal: Holes are bad (Makhzani et al. (2015))

Lack of Density in Latent Space is Bad

Two necessary and related conditions to achieve good coverage of the latent space:

- Most of the latent space (as measured by probability mass) maps to some image in the data manifold via a generator
  - Informal: Holes are bad (Makhzani et al. (2015))
- Most of the classes are close to each other

Lack of Density in Latent Space is Bad

Two necessary and related conditions to achieve good coverage of the latent space:

- Most of the latent space (as measured by probability mass) maps to some image in the data manifold via a generator
  - Informal: Holes are bad (Makhzani et al. (2015))
- Most of the classes are close to each other
  - Distant classes is bad for sample diversity

Lack of Density in Latent Space is Bad

Two necessary and related conditions to achieve good coverage of the latent space:

- Most of the latent space (as measured by probability mass) maps to some image in the data manifold via a generator
  - Informal: Holes are bad (Makhzani et al. (2015))
- Most of the classes are close to each other
  - Distant classes is bad for sample diversity (our theoretical contribution)

Lack of Density in Latent Space is Bad

Two necessary and related conditions to achieve good coverage of the latent space:

- Most of the latent space (as measured by probability mass) maps to some image in the data manifold via a generator
  - Informal: Holes are bad (Makhzani et al. (2015))
- Most of the classes are close to each other
  - Distant classes is bad for sample diversity (our theoretical contribution)
  - Attempted resolution: bring classes closer to each other and cover more probability

Lack of Density in Latent Space is Bad

Two necessary and related conditions to achieve good coverage of the latent space:

- Most of the latent space (as measured by probability mass) maps to some image in the data manifold via a generator
  - Informal: Holes are bad (Makhzani et al. (2015))
- Most of the classes are close to each other
  - Distant classes is bad for sample diversity (our theoretical contribution)
  - Attempted resolution: bring classes closer to each other and cover more probability (our empirical contribution)

Outline

Assumptions

Probability Decay

$$\|p\|_\infty \leq \min\left(\frac{1}{5}, h(K) \in \left[\Omega\left(\frac{1}{K}\right), o(1)\right]\right)$$

Assumptions

Probability Decay

$$\|p\|_\infty \leq \min\left(\frac{1}{5}, h(K) \in \left[\Omega\left(\frac{1}{K}\right), o(1)\right]\right)$$

- No class is asymptotically dominant

Assumptions

Probability Decay

$$\|p\|_\infty \leq \min\left(\frac{1}{5}, h(K) \in \left[\Omega\left(\frac{1}{K}\right), o(1)\right]\right)$$

- No class is asymptotically dominant

Partition of Latent Space

$$\bigcup_{i=1}^{K} K_i = \mathbb{R}^d$$

Assumptions

Probability Decay

$$\|p\|_\infty \leq \min\left(\frac{1}{5}, h(K) \in \left[\Omega\left(\frac{1}{K}\right), o(1)\right]\right)$$

- No class is asymptotically dominant

Partition of Latent Space

$$\bigcup_{i=1}^{K} K_i = \mathbb{R}^d$$

- Every latent vector maps to some image in the data manifold

Outline

Class Radius to Diversity

Theorem

Under the aforementioned assumptions, we have the following regarding the expected class radius of each image in the latent space of our generative model:

$$\mathbb{E}\left[r(z)\right] \leq \frac{\log\left(4\pi \log\left(1/\left\|p\right\|_2^2\right)\right)}{\sqrt{2\log\left(1/\left\|p\right\|_2^2\right)}}$$

- Upper bound depends *only* on $\|p\|_2^2$

Outline

$\|p\|_2^2$ as a Measure of Diversity

Problem

I have a bunch of socks in a drawer distributed according to p. If I draw two socks, what is the probability that they are of the same type?

$\|p\|_2^2$ as a Measure of Diversity

Problem

I have a bunch of socks in a drawer distributed according to p. If I draw two socks, what is the probability that they are of the same type?

Solution

- $\mathbb{P}\left[\text{both socks are of type } i\right] = p_i^2$

$\|p\|_2^2$ as a Measure of Diversity

Problem

I have a bunch of socks in a drawer distributed according to p. If I draw two socks, what is the probability that they are of the same type?

Solution

- $\mathbb{P}\left[\textit{both socks are of type } i\right] = p_i^2$
- $\sum p_i^2 = \|p\|_2^2$

$\|p\|_2^2$ as a Measure of Diversity

Definition (α-order Rényi Entropy)

$$H_\alpha(p) = \frac{1}{1-\alpha} \log \left(\sum_{i=1}^{K} p_i^\alpha \right)$$

$\|p\|_2^2$ as a Measure of Diversity

Definition (α-order Rényi Entropy)

$$H_\alpha(p) = \frac{1}{1-\alpha} \log \left(\sum_{i=1}^{K} p_i^\alpha \right)$$

Special Case: Shannon Entropy

$$\lim_{\alpha \to 1} H_\alpha(p) = -\sum_{i=1}^{K} p_i \log\left(p_i\right)$$

$\|p\|_2^2$ as a Measure of Diversity

Definition (α-order Rényi Entropy)

$$H_\alpha(p) = \frac{1}{1-\alpha} \log \left(\sum_{i=1}^{K} p_i^\alpha \right)$$

Special Case: Shannon Entropy

$$\lim_{\alpha \to 1} H_\alpha(p) = -\sum_{i=1}^{K} p_i \log(p_i)$$

Special Case: Collision Entropy

$$H_2(p) = -\log\left(\sum p_i^2\right) = \log\left(\frac{1}{\|p\|_2^2}\right)$$

$\|p\|_2^2$ as a Measure of Diversity

Fact (Maximum Entropy Distribution)

For all α, the distribution maximizing the α-order Rényi Entropy is the uniform distribution, yielding:

$$H_\alpha(p) \leq \log(K)$$

$\|p\|_2^2$ as a Measure of Diversity

Fact (Maximum Entropy Distribution)

For all α, the distribution maximizing the α-order Rényi Entropy is the uniform distribution, yielding:

$$H_\alpha(p) \leq \log(K)$$

Thus, our bound is *smallest* when p is the uniform distribution.

$\|p\|_2^2$ as a Measure of Diversity

Fact (Maximum Entropy Distribution)

For all α, the distribution maximizing the α-order Rényi Entropy is the uniform distribution, yielding:

$$H_\alpha(p) \leq \log(K)$$

Thus, our bound is *smallest* when p is the uniform distribution.

$$\mathbb{E}[r(z)] \leq \frac{\log(4\pi \log(K))}{\sqrt{2\log(K)}}$$

Outline

1. Introduction
   - Deep Generative Models Overview
   - Terms
   - Main Claim
2. Theory
   - Assumptions
   - Main Result
   - Relationship to Diversity
   - Proofs
   - Potential Implications
3. Experiments
   - VAE Modification
   - Experimental Setup
   - Results

Proof Outline of Main Theorem

1. Get upper bound on $\mathbb{P}\left[r(z) \geq \eta\right]$

Proof Outline of Main Theorem

1. Get upper bound on $\mathbb{P}\left[r(z) \geq \eta\right]$
   - Obtained by Fawzi et al. (2018)

Proof Outline of Main Theorem

1. Get upper bound on $\mathbb{P}\left[r(z) \geq \eta\right]$
   - Obtained by Fawzi et al. (2018)
2. Use $\mathbb{E}\left[X\right] = \int \mathbb{P}\left[X \geq \eta\right] \mathrm{d}\eta$

Proof Outline of Main Theorem

1. Get upper bound on $\mathbb{P}\left[r(z) \geq \eta\right]$
   - Obtained by Fawzi et al. (2018)
2. Use $\mathbb{E}\left[X\right] = \int \mathbb{P}\left[X \geq \eta\right] \mathrm{d}\eta$
   - Obtained by Fawzi et al. (2018)

Proof Outline of Main Theorem

1. Get upper bound on $\mathbb{P}\left[r(z) \geq \eta\right]$
   - Obtained by Fawzi et al. (2018)
2. Use $\mathbb{E}\left[X\right] = \int \mathbb{P}\left[X \geq \eta\right] \mathrm{d}\eta$
   - Obtained by Fawzi et al. (2018)
3. Massage a little

Proof Outline of Main Theorem

1. Get upper bound on $\mathbb{P}\left[r(z) \geq \eta\right]$
   - Obtained by Fawzi et al. (2018)
2. Use $\mathbb{E}\left[X\right] = \int \mathbb{P}\left[X \geq \eta\right] \mathrm{d}\eta$
   - Obtained by Fawzi et al. (2018)
3. Massage a little
   - Our contribution

Some Details

(Slightly rewritten) Result from Fawzi et al. (2018)

$$\mathbb{E}\left[r(z)\right] \leq \sum_{i=1}^{K} \Phi^{-1}(p_i) \cdot p_i + \frac{e^{-\Phi^{-1}(p_i)^2/2}}{\sqrt{2\pi}}$$

Some Details

(Slightly rewritten) Result from Fawzi et al. (2018)

$$\mathbb{E}\left[r(z)\right] \leq \sum_{i=1}^{K} \Phi^{-1}(p_i) \cdot p_i + \frac{e^{-\Phi^{-1}(p_i)^2/2}}{\sqrt{2\pi}}$$

Rewrite some:

$$\sum_{i=1}^{K} \Phi^{-1}(p_i) \cdot p_i + \frac{e^{-\Phi^{-1}(p_i)^2/2}}{\sqrt{2\pi}} \leq \sum_{i=1}^{K} p_i \left(\frac{\log\left(4\pi \log\left(1/p_i\right)\right)}{\sqrt{2\log\left(1/p_i\right)}} \right)$$

Some Details

Rewrite some:

$$\sum_{i=1}^{K} \Phi^{-1}(p_i) \cdot p_i + \frac{e^{-\Phi^{-1}(p_i)^2/2}}{\sqrt{2\pi}} \leq \sum_{i=1}^{K} p_i \left(\frac{\log\left(4\pi \log\left(1/p_i\right)\right)}{\sqrt{2\log\left(1/p_i\right)}} \right)$$

Some Details

Rewrite some:

$$\sum_{i=1}^{K} \Phi^{-1}(p_i) \cdot p_i + \frac{e^{-\Phi^{-1}(p_i)^2/2}}{\sqrt{2\pi}} \leq \sum_{i=1}^{K} p_i \left(\frac{\log\left(4\pi \log\left(1/p_i\right)\right)}{\sqrt{2\log\left(1/p_i\right)}} \right)$$

Use concavity of:

$$g(x) := \frac{\log\left(4\pi \log\left(1/x\right)\right)}{\sqrt{2\log\left(1/x\right)}}$$

Some Details

Rewrite some:

$$\sum_{i=1}^{K} \Phi^{-1}(p_i) \cdot p_i + \frac{e^{-\Phi^{-1}(p_i)^2/2}}{\sqrt{2\pi}} \leq \sum_{i=1}^{K} p_i \left(\frac{\log\left(4\pi \log\left(1/p_i\right)\right)}{\sqrt{2\log\left(1/p_i\right)}} \right)$$

Use concavity of:

$$g(x) := \frac{\log\left(4\pi \log\left(1/x\right)\right)}{\sqrt{2\log\left(1/x\right)}}$$

Then Jensen:

$$\mathbb{E}\left[r(z)\right] \leq g\left(\sum_{i=1}^{K} p_i \cdot p_i \right) = g\left(\|p\|_2^2 \right) = \boxed{\frac{\log\left(4\pi \log\left(1/\|p\|_2^2\right)\right)}{\sqrt{2\log\left(1/\|p\|_2^2\right)}}}$$

Outline

Test for Diversity

Problem

Suppose you have two generators g_1 and g_2, classifier f, and access to $\mathbb{E}[r(z)]$. Compare the diversity of the generators.

Test for Diversity

Problem

Suppose you have two generators g_1 and g_2, classifier f, and access to $\mathbb{E}[r(z)]$. Compare the diversity of the generators.

Solution

Compare $\mathbb{E}[r(z)]$ for g_1 and g_2; claim that the generator with higher class radius is less diverse.

Test for Diversity

Problem

Suppose you have two generators g_1 and g_2, classifier f, and access to $\mathbb{E}[r(z)]$. Compare the diversity of the generators.

Solution

Compare $\mathbb{E}[r(z)]$ for g_1 and g_2; claim that the generator with higher class radius is less diverse.

Unfortunately, computing $\mathbb{E}[r(z)]$ might be more difficult than computing $\|p\|_2^2$ unless K is extremely large

Adversarial Examples

- Recall:

$$r(z) = \inf_{z' \in \bigcup_{j \neq c(g(z))} K_j} \left\| z - z' \right\|$$

Adversarial Examples

- Recall:

$$r(z) = \inf_{z' \in \bigcup_{j \neq c(g(z))} K_j} \left\| z - z' \right\|$$

- “Find an adversarial example in z-space”

Adversarial Examples

- Recall:

$$r(z) = \inf_{z' \in \bigcup_{j \neq c(g(z))} K_j} \left\| z - z' \right\|$$

- "Find an adversarial example in z-space"
- If g is L-Lipschitz, then on average, you don't have to perturb too much in image space:

$$L \cdot \frac{\log\left(4\pi \log\left(1/\left\|p\right\|_2^2\right)\right)}{\sqrt{2\log\left(1/\left\|p\right\|_2^2\right)}}$$

Adversarial Examples

- Recall:

$$r(z) = \inf_{z' \in \bigcup_{j \neq c(g(z))} K_j} \left\| z - z' \right\|$$

- "Find an adversarial example in z-space"
- If g is L-Lipschitz, then on average, you don't have to perturb too much in image space:

$$L \cdot \frac{\log\left(4\pi \log\left(1/\left\|p\right\|_2^2\right)\right)}{\sqrt{2\log\left(1/\left\|p\right\|_2^2\right)}}$$

- Under the probability decay assumption, above approaches 0 as the number of classes K grows arbitrarily large

Outline

Motivation

Problem

How do we bring different classes closer together while training a VAE?

Motivation

Problem

How do we bring different classes closer together while training a VAE?

Solution

Penalize distant encodings.

Motivation

Problem

How do we bring different classes closer together while training a VAE?

Solution

Penalize distant encodings.

$$\omega\left(I_1, I_2\right) := \frac{\left\|f\left(I_1\right) - f\left(I_2\right)\right\|_2^2}{\left\|I_1 - I_2\right\|_2^2}$$

Motivation

Problem

How do we bring different classes closer together while training a VAE?

Solution

Penalize distant encodings.

$$\omega\left(I_1, I_2\right) := \frac{\left\|f\left(I_1\right) - f\left(I_2\right)\right\|_2^2}{\left\|I_1 - I_2\right\|_2^2}$$

Regularize with $\omega\left(\cdot,\cdot\right)$ *summed over many pairs of images.*

Motivation

Problem

How do we bring different classes closer together while training a VAE?

Solution

Penalize distant encodings.

$$\omega(I_1, I_2) := \frac{\|f(I_1) - f(I_2)\|_2^2}{\|I_1 - I_2\|_2^2}$$

Regularize with $\omega(\cdot,\cdot)$ summed over many pairs of images. To force ω to be low, enforce Lipschitz constant via network weight clipping.

Outline

Experimental Setup

- Used a modified VAE implementation inspired from PyTorch examples repository

Experimental Setup

- Used a modified VAE implementation inspired from PyTorch examples repository
  - DCGAN-like encoder and decoder

Experimental Setup

- Used a modified VAE implementation inspired from PyTorch examples repository
  - DCGAN-like encoder and decoder
- Vanilla VAE objective

Experimental Setup

- Used a modified VAE implementation inspired from PyTorch examples repository
  - DCGAN-like encoder and decoder
- Vanilla VAE objective
- Regularizer (batch size b; set $\lambda = 30$):

$$\lambda \cdot \sum_{i=1}^{b/2} \omega \left(I_i, I_{i+b/2} \right)$$

Experimental Setup

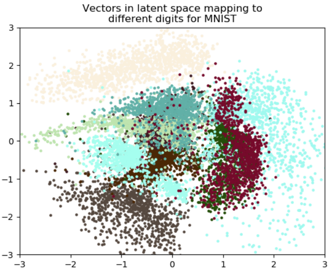
- Used a modified VAE implementation inspired from PyTorch examples repository
  - DCGAN-like encoder and decoder
- Vanilla VAE objective
- Regularizer (batch size b; set $\lambda = 30$):

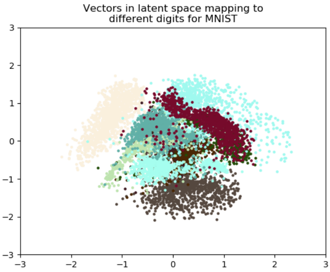
$$\lambda \cdot \sum_{i=1}^{b/2} \omega \left(I_i, I_{i+b/2} \right)$$

- Clip weights to $[-0.3, 0.3]$

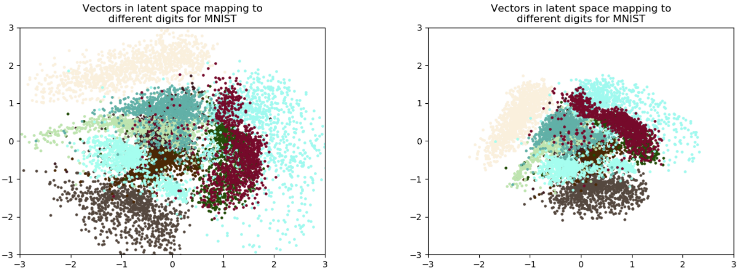
Outline

Geometric Implications





Geometric Implications



Conclusion: Modified VAE makes classes come closer to one another but decreases probability mass of space covered

Sample Quality

Sample Quality

Conclusion: Modified VAE does not change quality of generated samples noticeably

Interpretation of Results

Hyperparameter selection is tricky

Interpretation of Results

Hyperparameter selection is tricky

- Better weight clip constant might lead to better coverage of the latent space

Interpretation of Results

Hyperparameter selection is tricky

- Better weight clip constant might lead to better coverage of the latent space

Regularizer itself is not particularly useful

Interpretation of Results

Hyperparameter selection is tricky

- Better weight clip constant might lead to better coverage of the latent space

Regularizer itself is not particularly useful

- Network could be learning a (almost) scaled-down version of the latent space

Interpretation of Results

Hyperparameter selection is tricky

- Better weight clip constant might lead to better coverage of the latent space

Regularizer itself is not particularly useful

- Network could be learning a (almost) scaled-down version of the latent space

Problem

For future: Devise an optimization objective that favors both closer classes and better coverage of the latent space.

Conclusion

Summary

We care about coverage of the latent space.

Conclusion

Summary

We care about coverage of the latent space.

- As previously identified, this is indicative of the model having better learned the data manifold.

Conclusion

Summary

We care about coverage of the latent space.

- As previously identified, this is indicative of the model having better learned the data manifold.

We care about density of the latent space.

Conclusion

Summary

We care about coverage of the latent space.

- As previously identified, this is indicative of the model having better learned the data manifold.

We care about density of the latent space.

- As we prove, this is indicative of the model learning a more diverse distribution.

Conclusion

Summary

We care about coverage of the latent space.

- As previously identified, this is indicative of the model having better learned the data manifold.

We care about density of the latent space.

- As we prove, this is indicative of the model learning a more diverse distribution.

Future Work

We want to leverage these findings empirically.

Conclusion

Summary

We care about coverage of the latent space.

- As previously identified, this is indicative of the model having better learned the data manifold.

We care about density of the latent space.

- As we prove, this is indicative of the model learning a more diverse distribution.

Future Work

We want to leverage these findings empirically.

- Develop optimization procedures to achieve these properties

Conclusion

Summary

We care about coverage of the latent space.

- As previously identified, this is indicative of the model having better learned the data manifold.

We care about density of the latent space.

- As we prove, this is indicative of the model learning a more diverse distribution.

Future Work

We want to leverage these findings empirically.

- Develop optimization procedures to achieve these properties
- Test generative models for diversity

Conclusion

Summary

We care about coverage of the latent space.

- As previously identified, this is indicative of the model having better learned the data manifold.

We care about density of the latent space.

- As we prove, this is indicative of the model learning a more diverse distribution.

Future Work

We want to leverage these findings empirically.

- Develop optimization procedures to achieve these properties
- Test generative models for diversity
- Meta-Problem: Theoretically understand generative models better

References I

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Sanjeev Arora, Andrej Risteski, and Yi Zhang. Do GANs learn the distribution? some theory and empirics. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=BJehNfW0-`.

Yoshua Bengio and Yann LeCun. Scaling learning algorithms towards AI. In *Large Scale Kernel Machines*. MIT Press, 2007.

Ali Borji. Pros and cons of gan evaluation measures. *arXiv preprint arXiv:1802.03446*, 2018.

GM Bosyk, M Portesi, and A Plastino. Collision entropy and optimal uncertainty. *Physical Review A*, 85(1):012108, 2012.

Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

References II

Lutz Duembgen. Bounding standard gaussian tail probabilities. *arXiv preprint arXiv:1012.2063*, 2010.

Alhussein Fawzi, Hamza Fawzi, and Omar Fawzi. Adversarial vulnerability for any classifier. *arXiv preprint arXiv:1802.08686*, 2018.

Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18: 1527–1554, 2006.

References III

He Huang, Phillip Yu, and Changhu Wang. An introduction to image synthesis with generative adversarial nets. *arXiv preprint arXiv:1803.04469*, 2018.

Matt Jordan, Naren Manoj, Surbhi Goel, and Alex Dimakis. Combined adversarial attacks. *NIPS 2018 (submitted)*, 2018.

Valentin Khrulkov and Ivan Oseledets. Geometry score: A method for comparing generative adversarial networks. *arXiv preprint arXiv:1802.02664*, 2018.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Yann LeCun and Corinna Cortes. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/, 2010. URL `http://yann.lecun.com/exdb/mnist/`.

References IV

Fei-Fei Li, Justin Johnson, and Serena Yeung. Lecture notes for convolutional neural networks for visual recognition, May 2018.

Zinan Lin, Ashish Khetan, Giulia Fanti, and Sewoong Oh. Pacgan: The power of two samples in generative adversarial networks. *arXiv preprint arXiv:1712.04086*, 2017.

Erik Linder-Norn. Pytorch-gan. `https://github.com/eriklindernoren/PyTorch-GAN`, 2018.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

References V

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pp. 2234–2242, 2016.

Shibani Santurkar, Ludwig Schmidt, and Aleksander Madry. A classification-based perspective on gan distributions. *arXiv preprint arXiv:1711.00970*, 2017.

Shibani Santurkar, Ludwig Schmidt, and Aleksander Madry. A classification-based study of covariate shift in GAN distributions. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4487–4496, Stockholmsmssan, Stockholm Sweden, 10–15 Jul 2018. PMLR. URL `http://proceedings.mlr.press/v80/santurkar18a.html`.

References VI

Akash Srivastava, Lazar Valkoz, Chris Russell, Michael U Gutmann, and Charles Sutton. Veegan: Reducing mode collapse in gans using implicit variational learning. In *Advances in Neural Information Processing Systems*, pp. 3308–3318, 2017.

Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. Wasserstein auto-encoders. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=HkL7n1-0b`.