

14. Autoencoder

Presenter : Dong Hyun Kim
www.github.com/henniekim

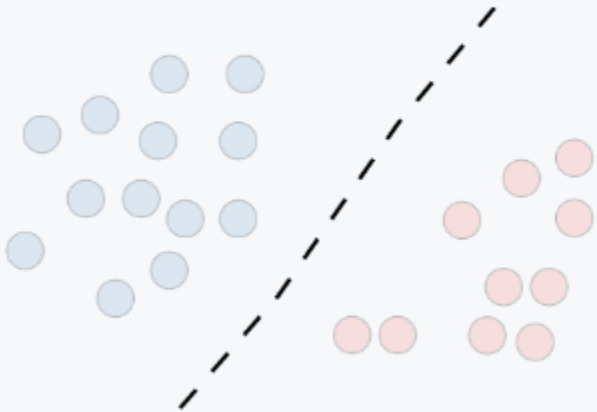
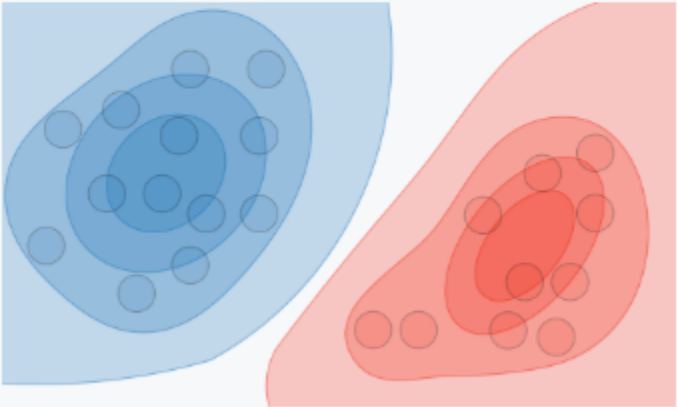


Contents

- What we cover
 - Generative vs Discriminative
 - Manifold Learning
 - Curse of dimensionality
 - Manifold hypothesis
 - Dimension reduction
 - Autoencoder
 - AE
 - DAE
 - CAE

Generative vs Discriminative

- 생성 모델(generative model)은 각 클래스의 분포를 모델링 한다.
- 판별 모델(discriminative model)은 클래스 사이의 경계를 학습한다.

	Discriminative model	Generative model
Goal	Directly estimate $P(y x)$	Estimate $P(x y)$ to then deduce $P(y x)$
What's learned	Decision boundary	Probability distributions of the data
Illustration		
Examples	Regressions, SVMs	GDA, Naive Bayes

- 입력 데이터 x 가 주어지고 데이터를 label y 로 분류하고 싶을 때.
- Generative 모델은 joint probability distribution $p(x, y)$ 을 학습한다.
- Discriminative 모델은 조건부 확률 분포 $p(y|x)$ 를 학습한다.

Simple example

$(x, y): (1,0), (1,0), (2,0), (2,1)$ 의 데이터가 주어졌을 때

$p(x, y) :$

	Y=0	Y=1
X=1	1/2	0
X=2	1/4	1/4

-> Generative model

$p(y|x) :$

	Y=0	Y=1
X=1	1	0
X=2	1/2	1/2

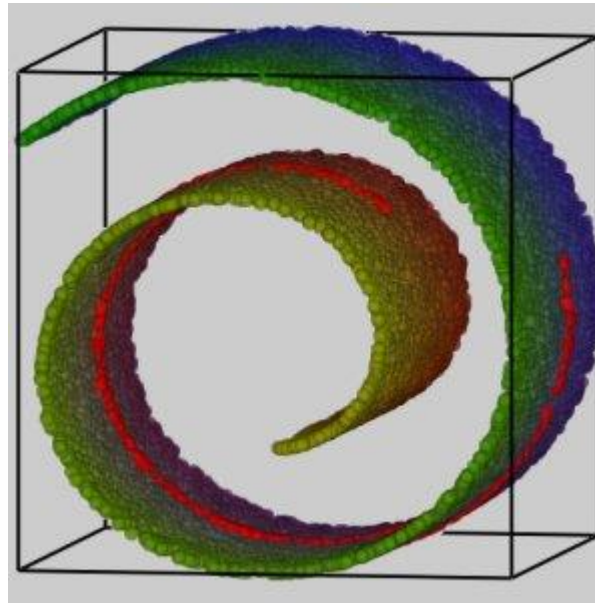
-> Discriminative model

- 코끼리($y=1$)와 개($y=0$)을 분류할 때, x 를 동물의 특징 벡터라고 하면
 - ,
 - 학습 데이터가 주어졌을 때 logistic regression이나 perceptron 알고리즘은 두 동물을 나누는 decision boundary를 찾으려 함.
 - 새로운 동물 데이터가 들어왔을 때 경계선을 기준으로 코끼리인지 개인지를 판별하게 됨.
= **discriminative** 모델
-
- 코끼리와 개를 나타내는 함수를 데이터로부터 직접 모델링함
 - 새로운 데이터가 들어왔을 때 모델링한 함수로부터 어떤 동물에 가까운지 판별함. = **generative** 모델

Manifold learning

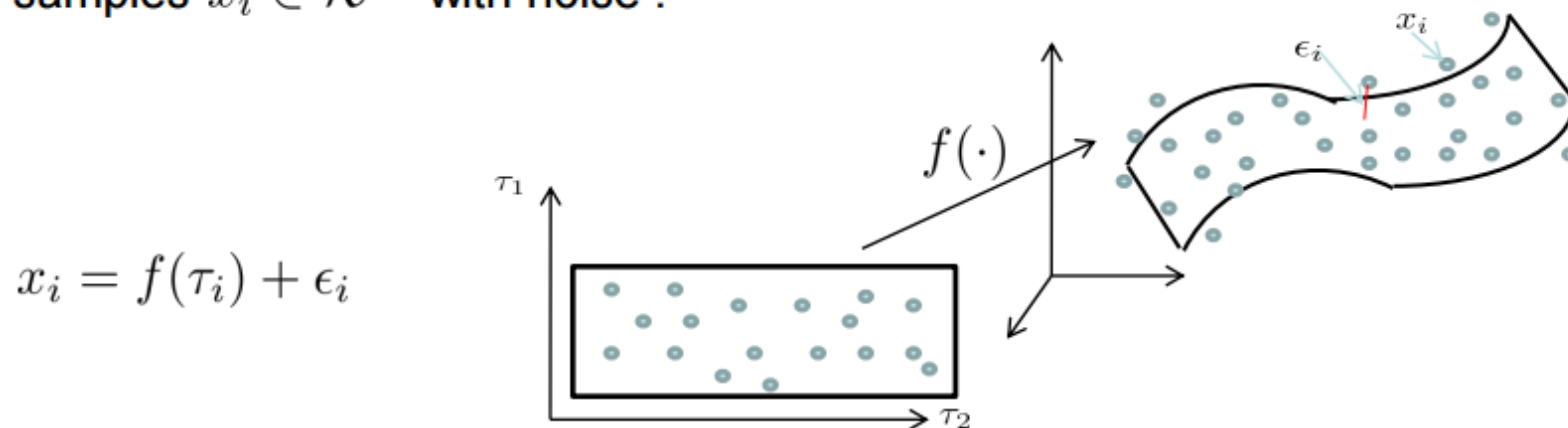
- Can be viewed as the non-linear version of PCA
- What if the best representation lies in some weirdly shaped surface?

요런 요상하게 생긴 데이터는
PCA로 분석하기 힘들!



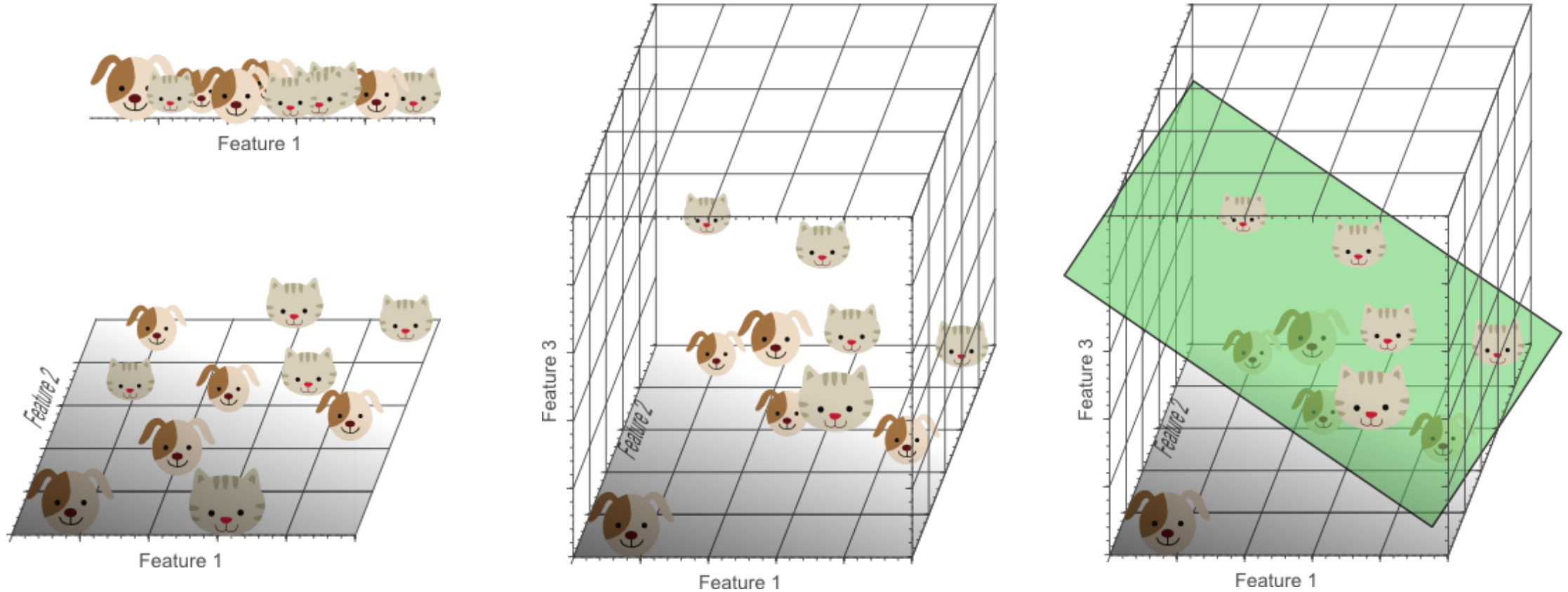
What is manifold learning?

- A d dimensional manifold \mathcal{M} is embedded in an m dimensional space, and there is an explicit mapping $f : \mathcal{R}^d \rightarrow \mathcal{R}^m$ where $d \leq m$. We are given samples $x_i \in \mathcal{R}^m$ with noise.

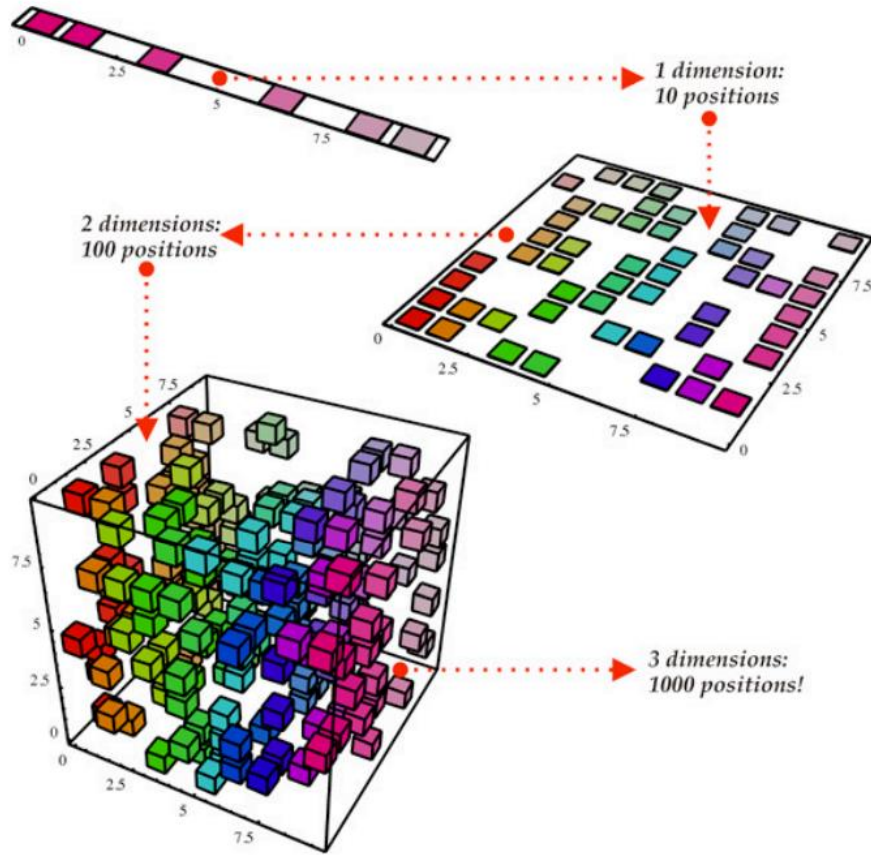


- $f(\cdot)$ is called **embedding function**, m is the **extrinsic dimension**, d is the **intrinsic dimension** or dimension of the latent space.
- Finding either $f(\cdot)$ or τ_i from given x_i is called **manifold learning**.
- We don't have any information about the function $f(\cdot)$, distribution of the data in low dimension τ_i , and the distribution of the noise.
- We assume $p(\tau)$ is **smooth**, τ is distributed **uniformly**, and noise is **small**.

Curse of dimensionality



Curse of dimensionality



- 데이터를 표현하는 차원이 커질 수록 데이터를 생성할 수 있는 경우의 수가 증가
- 해당 데이터를 잘 모델링(=학습)하기 위해서는 보다 급격히 많은 양의 데이터가 필요함

Manifold hypothesis

- 고차원 데이터의 밀도는 낮지만, 이들의 집합을 포함하는 저차원의 manifold가 존재한다.
- 이 저차원의 manifold를 벗어나는 순간 급격히 밀도가 낮아진다.
- 200x200 이미지를 가정하면 10^{96329} 의 가능한 경우가 있음
- 임의로 생성한 데이터는 실제 세상의 데이터와 매우 동떨어져있음.
- 즉 자연에서 발생하는 이미지 공간은 대부분 비어 있음!

<http://colah.github.io/posts/2014-10-Visualizing-MNIST/>



여기서 좀 더 자세히

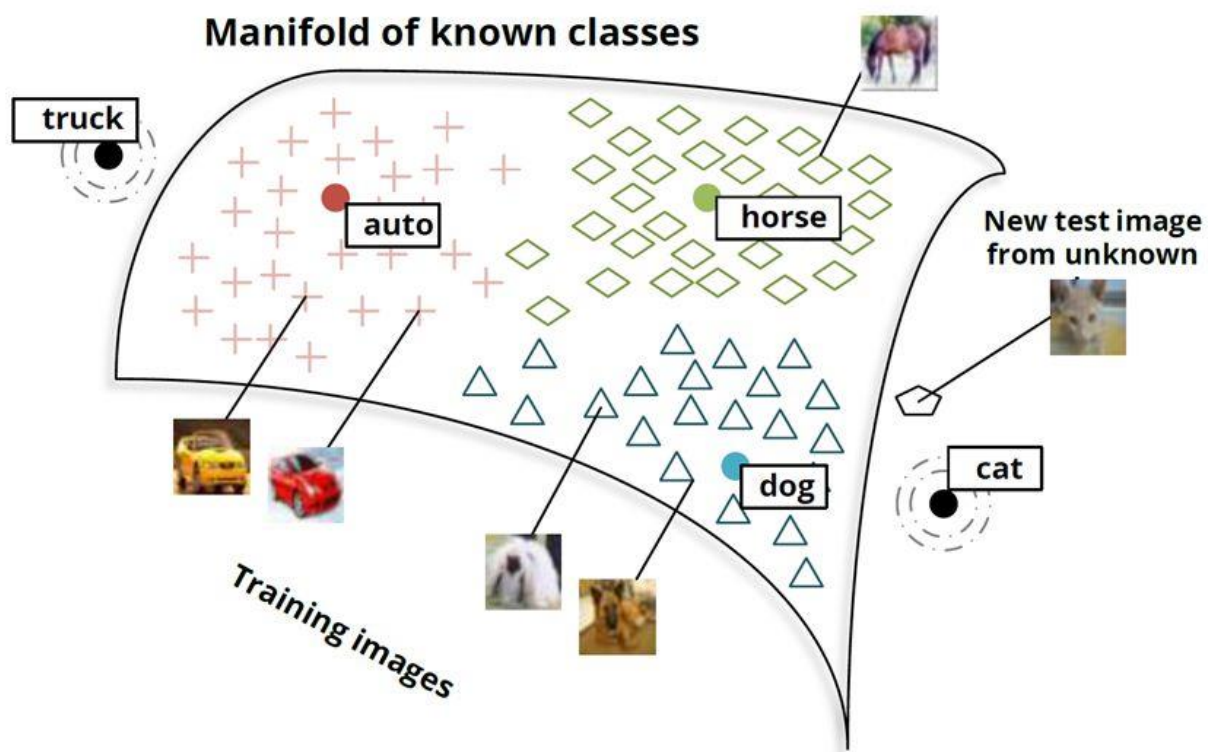
데이터는 어딘가에 몰려 있다!

- 우리가 보는 데이터는 공간상 특정 위치에 몰려 있다.
- 그렇지 않다면(=공간상에 골고루 분포한다면), 데이터를 random으로 sampling 했을 때 의미 있는 그림이 나와야 함!



하지만 Noise 같이 의미 없는
그림이 나옴...

Another View: Manifolds and Classes

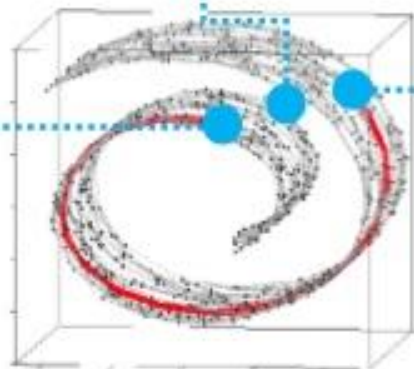


Reasonable distance metric

Interpolation in high dimension

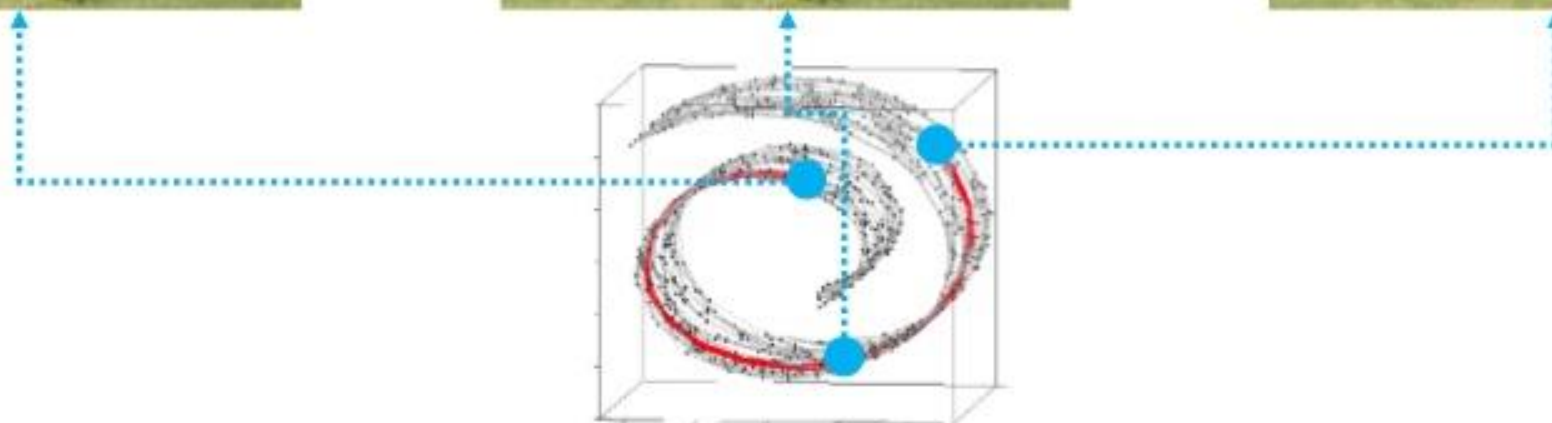


- 고차원 공간에서는, 의미적으로 가깝다고 생각되는 두 샘플들 간의 거리는 실제로는 먼 경우가 많다.
- 고차원 공간에서 가까운 두 샘플은 의미적으로는 굉장히 다를 수 있다.



Reasonable distance metric

Interpolation in manifold

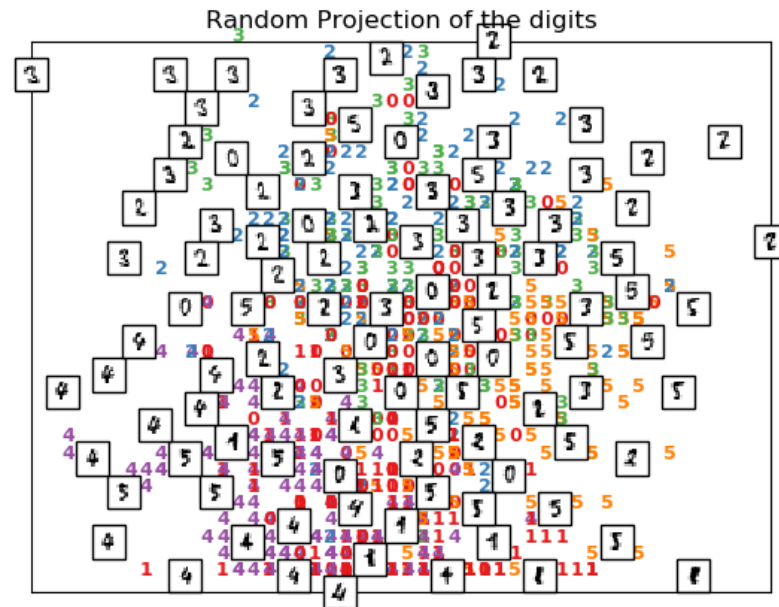


Dimension Reduction

- Random projection

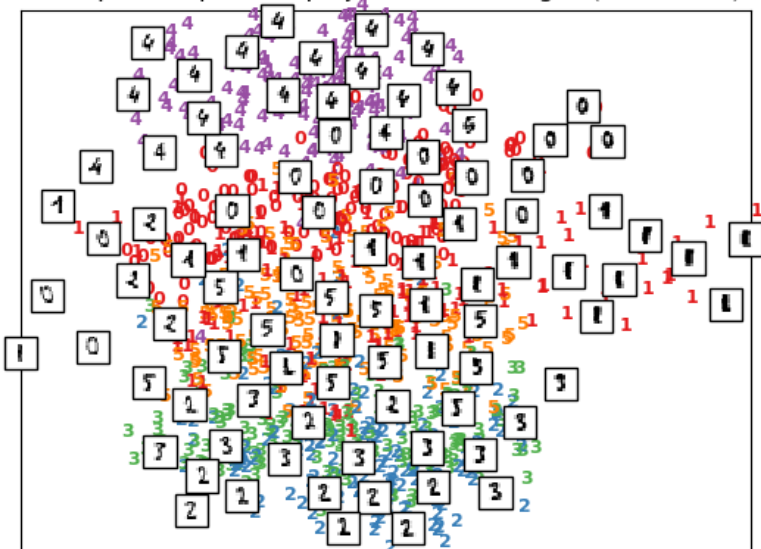
A selection from the 64-dimensional digits dataset

0	1	2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	5
5	5	0	4	1	3	5	1	0	0	2	2	2	0	1	2	3	3	3	3
4	4	1	5	0	5	2	2	0	0	1	3	2	1	4	3	1	3	1	4
3	4	4	0	5	3	1	5	4	4	2	2	2	5	5	4	4	0	0	1
2	3	4	5	0	1	2	3	4	5	0	1	2	3	4	5	0	5	5	5
0	4	4	3	5	1	0	0	2	2	1	0	4	2	3	3	7	3	4	4
1	5	0	5	2	1	0	0	1	3	1	4	3	1	3	4	4	3	1	4
0	5	7	4	5	4	4	1	1	1	5	5	4	4	0	0	1	2	3	4
5	0	4	2	3	4	5	0	4	2	3	4	5	0	5	5	5	0	4	1
3	5	1	0	0	2	2	2	0	4	2	3	3	3	3	4	4	1	5	0
5	2	2	0	0	1	3	2	1	4	3	1	3	1	4	3	1	4	0	5
3	1	5	4	4	2	2	2	5	5	4	4	0	3	0	1	2	3	4	5
0	1	2	3	4	5	0	1	2	3	4	5	0	5	5	5	0	4	1	3
5	1	0	0	1	2	1	0	1	2	3	3	3	3	4	4	1	5	0	5
1	2	0	0	1	3	2	1	4	3	1	3	1	4	3	1	4	0	5	3
1	5	4	4	2	2	2	5	5	4	4	0	0	1	2	3	4	5	0	1
2	3	4	5	0	1	2	3	4	5	0	5	5	5	0	4	1	3	5	4
0	0	2	1	2	0	1	2	3	3	3	3	4	4	1	5	0	5	1	2
0	0	1	3	1	1	4	3	1	3	1	4	3	1	4	0	5	3	1	5
4	4	2	2	1	5	5	4	4	0	1	2	3	4	5	0	1	2	3	3



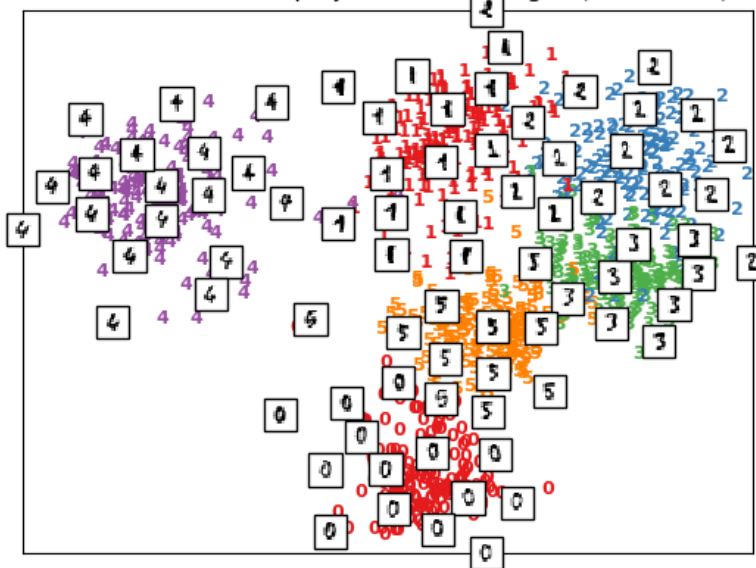
- PCA

Principal Components projection of the digits (time 0.00s)



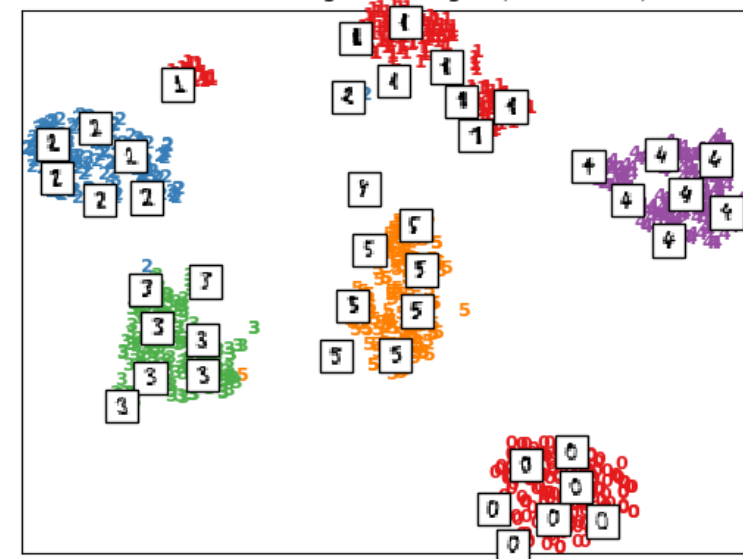
- Linear Discriminant Projection

Linear Discriminant projection of the digits (time 0.05s)

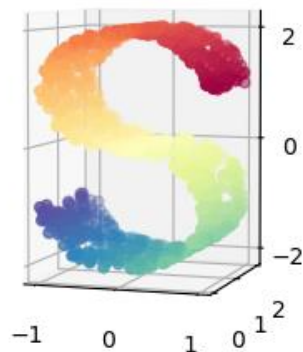


- t-SNE

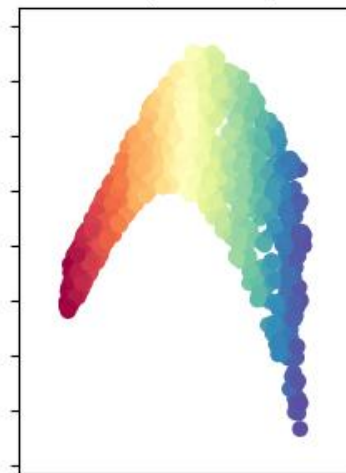
t-SNE embedding of the digits (time 4.57s)



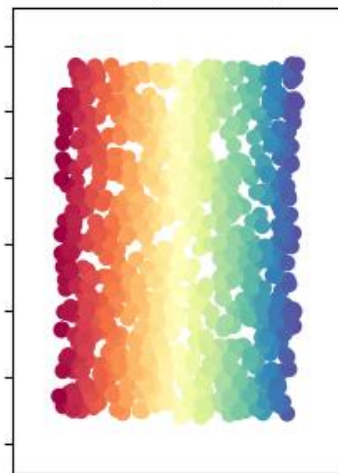
Manifold Learning with 1000 points, 10 neighbors



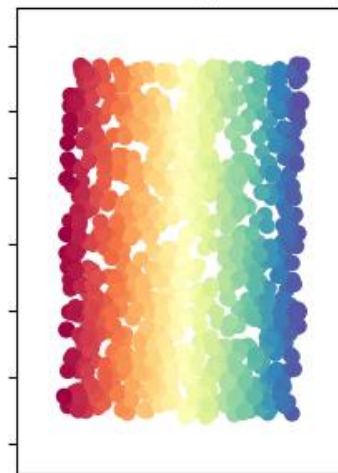
LLE (0.11 sec)



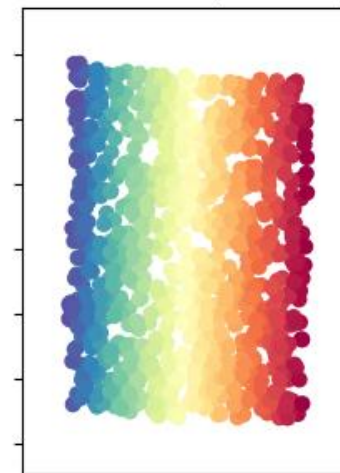
LTSA (0.16 sec)



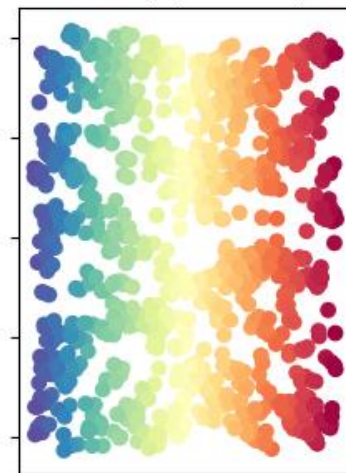
Hessian LLE (0.36 sec)



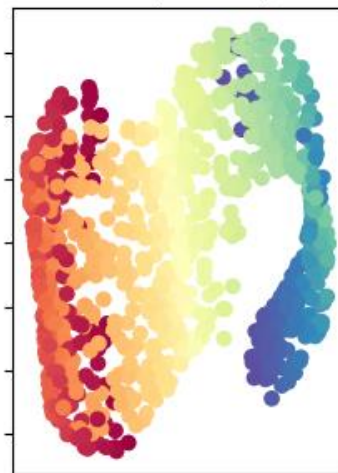
Modified LLE (0.21 sec)



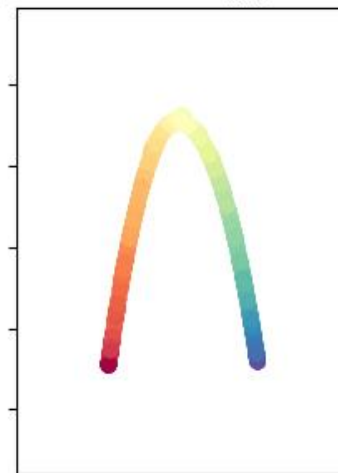
Isomap (0.34 sec)



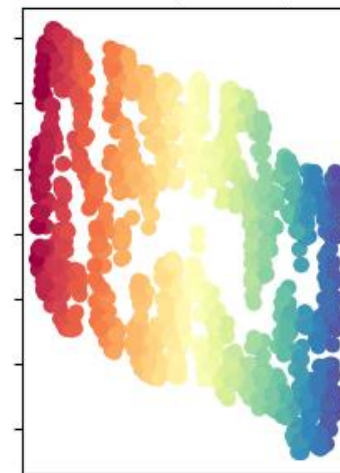
MDS (2.7 sec)



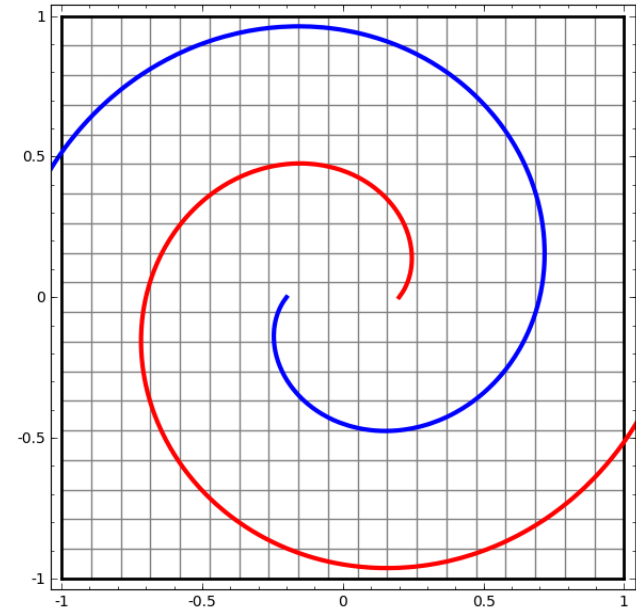
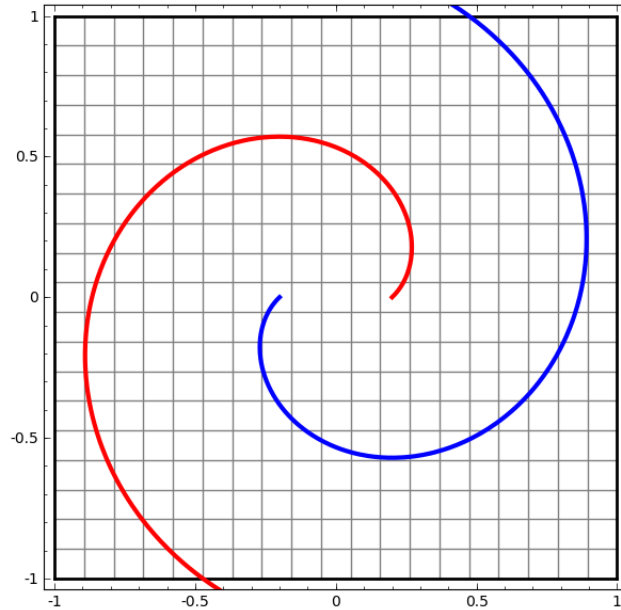
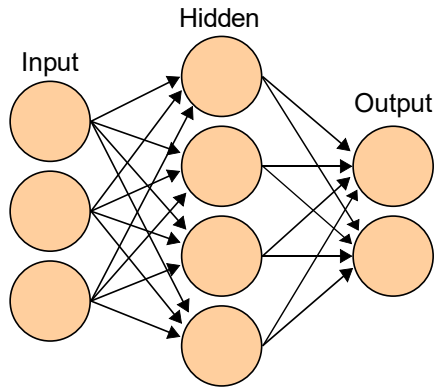
SpectralEmbedding (0.13 sec)

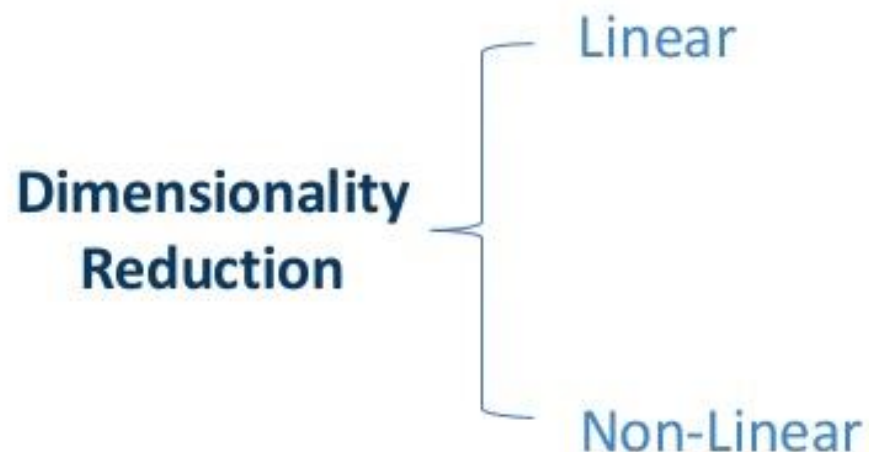


t-SNE (5.9 sec)



What Neural Networks really do?





- Principal Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)
- etc..
- **Autoencoders (AE)**
- t-distributed stochastic neighbor embedding (t-SNE)
- Isomap
- Locally-linear embedding (LLE)
- etc..

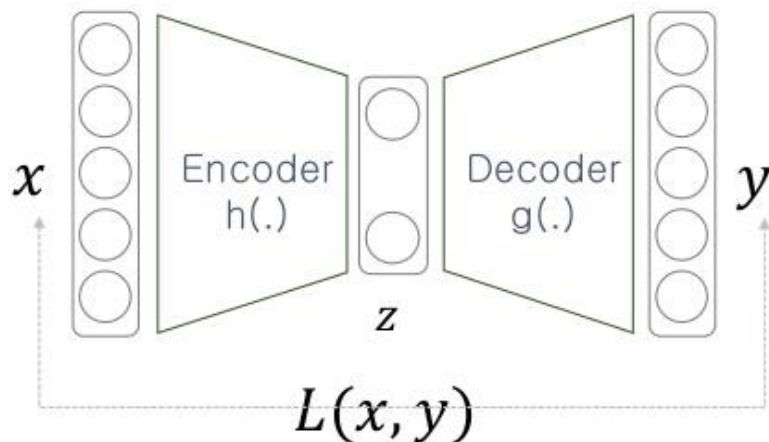
Autoencoder

INTRODUCTION

Notations

AUTOENCODERS

2 / 24



$$z = h(x) \in \mathbb{R}^{d_z}$$

$$y = g(z) = g(h(x))$$

$$L_{AE} = \sum_{x \in D} L(x, y)$$

입력이 출력과 최대한 비슷하도록,
그 차이를 줄이는 방향으로 학습함 (학습 자체는 매우 simple함)

- Make output layer same size as input layer

$$x, y \in \mathbb{R}^d$$

- Loss encourages output to be close to input

$$L(x, y)$$

입출력이 동일한 네트워크

- Unsupervised Learning → Supervised Learning

비교사 학습 문제를 교사 학습 문제로 바꾸어서 해결

Decoder가 최소한 학습 데이터는 생성해 낼 수 있게 된다.
→ 생성된 데이터가 학습 데이터 좀 닮아 있다.

Encoder가 최소한 학습 데이터는 잘 latent vector로 표현
할 수 있게 된다.
→ 데이터의 추상화를 위해 많이 사용된다.

Denoising Autoencoder (DAE)

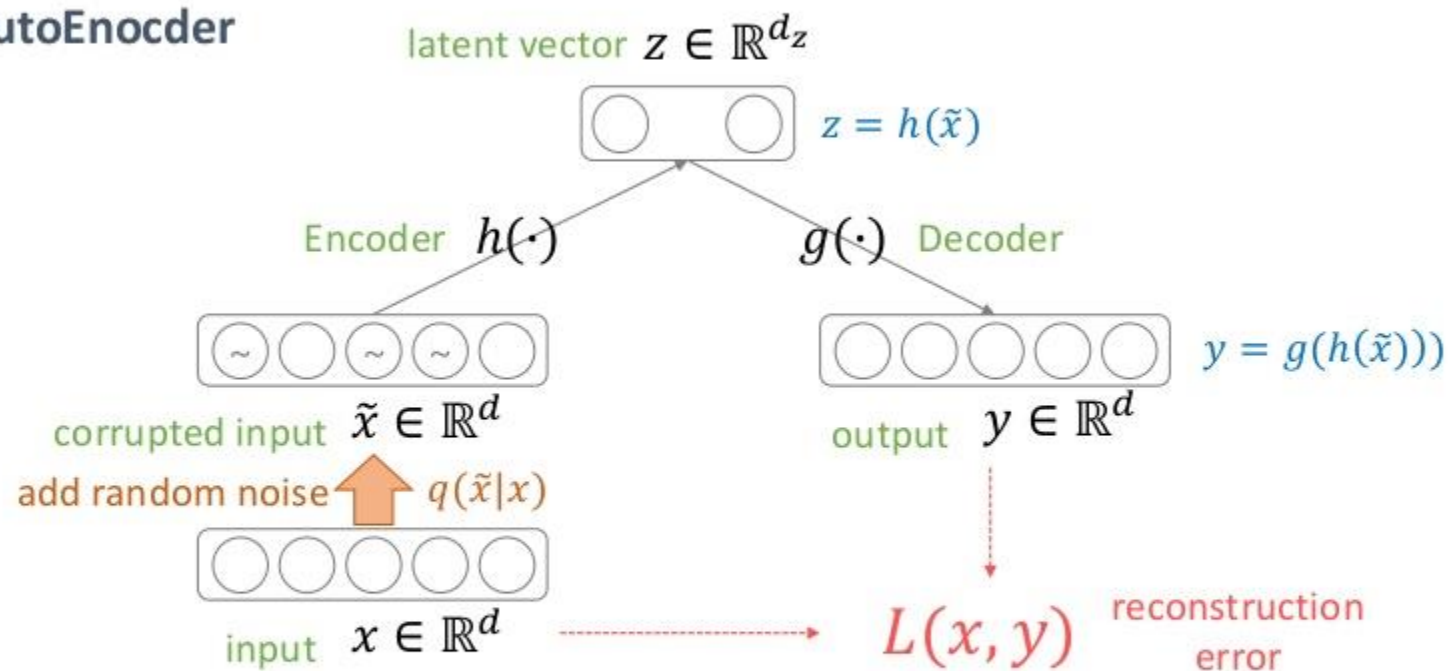
DAE

Introduction

AUTOENCODERS

10 / 24

Denoising AutoEncoder

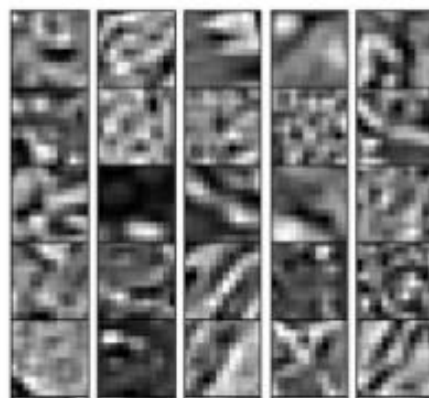


$$\text{Minimize } L_{DAE} = \sum_{x \in D} E_{q(\tilde{x}|x)} [L(x, g(h(\tilde{x})))]$$

입력에 noise가 추가된 이미지
(그러나 원본에서 크게 벗어나지 않은)
를 가지고 원본 이미지를 복구할 수 있도록 학습

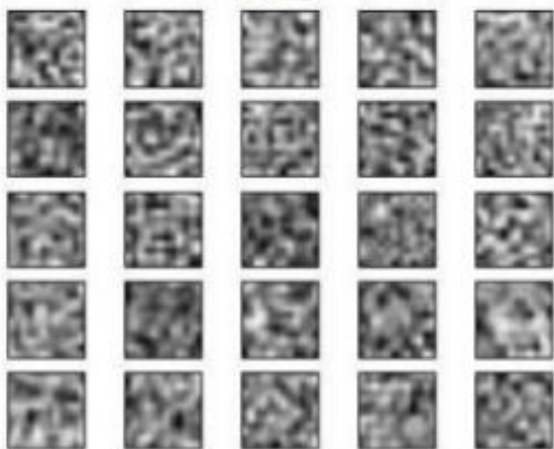
Natural image patches (12x12 pixels) : 100 hidden units

랜덤값으로 초기화하였기 때문에
노이즈처럼 보이는 필터일 수록 학습이
잘 안 된 것이고 edge filter와 같은 모습
일 수록 학습이 잘 된 것이다.

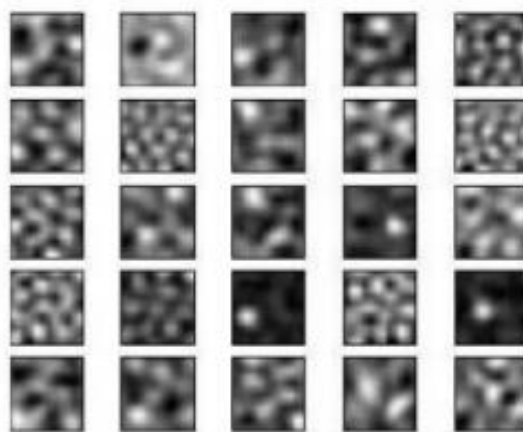


- Mean Squared Error
- 100 hidden units
- Salt-and-pepper noise

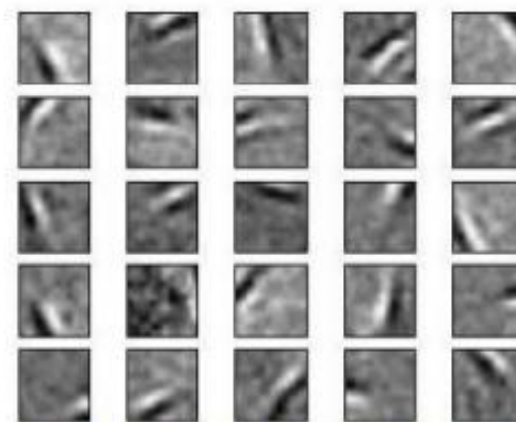
AE



AE with weight decay



DAE



10% salt-and-pepper noise

Contractive autoencoder (CAE)

CAE

Loss function

AUTOENCODERS

21 / 24

$$L_{SCAE} = \sum_{x \in D} \underbrace{L(x, g(h(x)))}_{\text{Reconstruction Error}} + \lambda \underbrace{\left\| \frac{\partial h}{\partial x}(x) \right\|_F^2}_{\text{Analytic Contractive Regularization}}$$

x가 약간 변했을 때 값이 변하지 않는 함수를 배우게 됨

For training examples, encourages both:

- small reconstruction error
- representation insensitive to small variations around example

Analytic contractive regularization term is

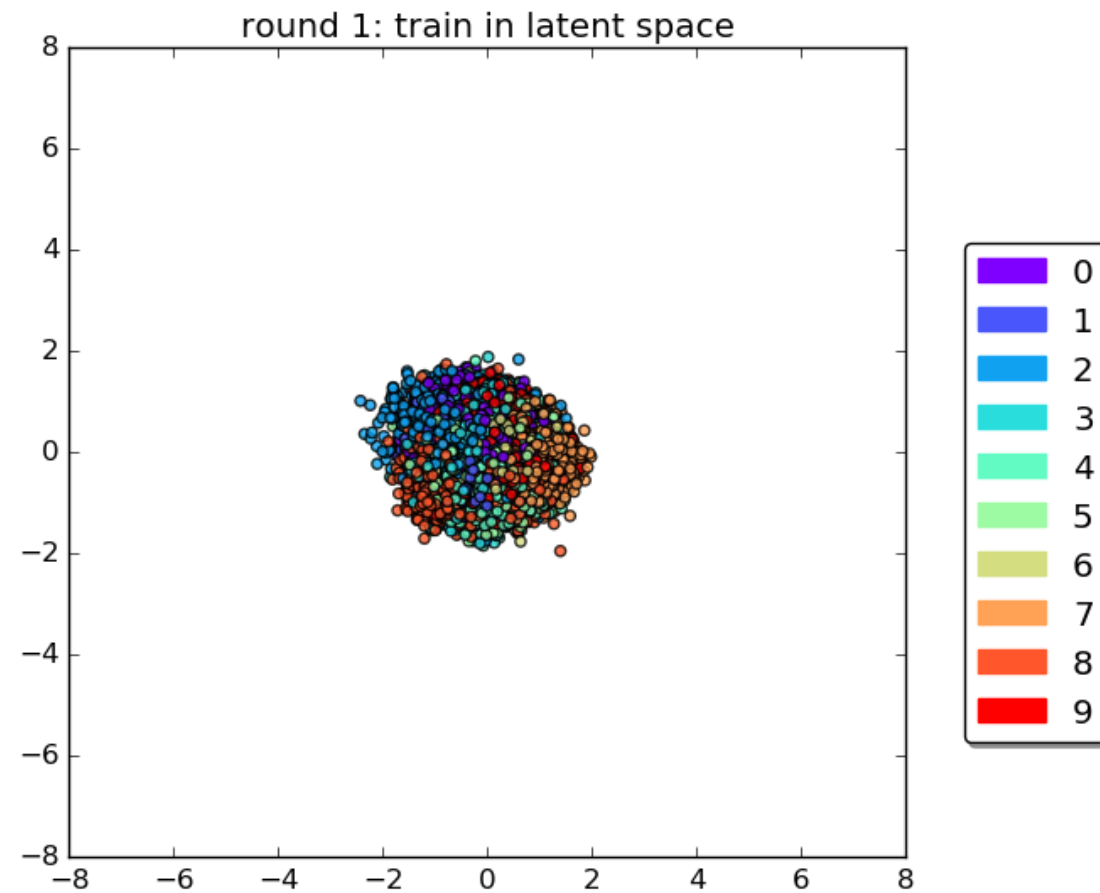
- Frobenius norm of the Jacobian of the non-linear mapping

$$\left\| \frac{\partial h}{\partial x}(x) \right\|_F^2 = \sum_{ij} \left(\frac{\partial z_j}{\partial x_i}(x) \right)^2 = \|J_h(x)\|_F^2$$

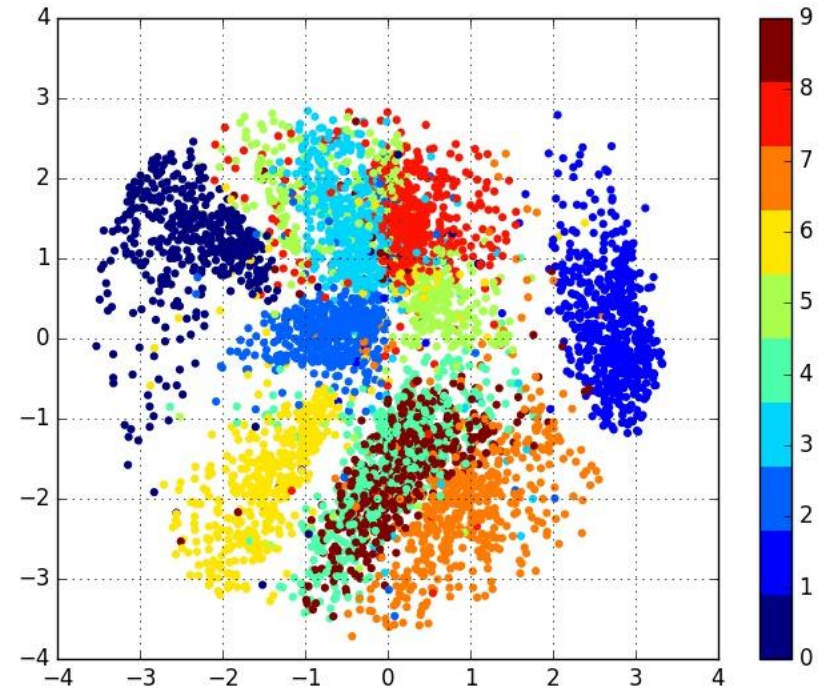
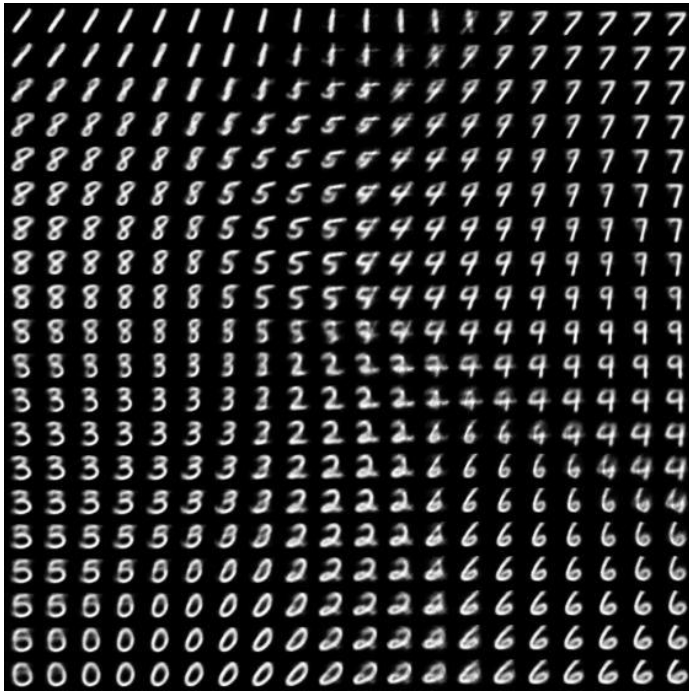
Penalizing $\|J_h(x)\|_F^2$ encourages the mapping to the feature space to be contractive in the neighborhood of the training data.

highlights the advantages for representations to be locally invariant in many directions of change of the raw input.

(요즘 잘 안씀...)
이런게 있다 정도
만 알고 넘어가기



Learned MNIST manifold



Q & A

Notice

- 다음 강의 일정 : 1월 17일 목요일
- 다음 발표 주제 : 15장 표현 학습 (representation learning)
- 다음 발표자 : 권선우