

Chapter 14

Autoencoder

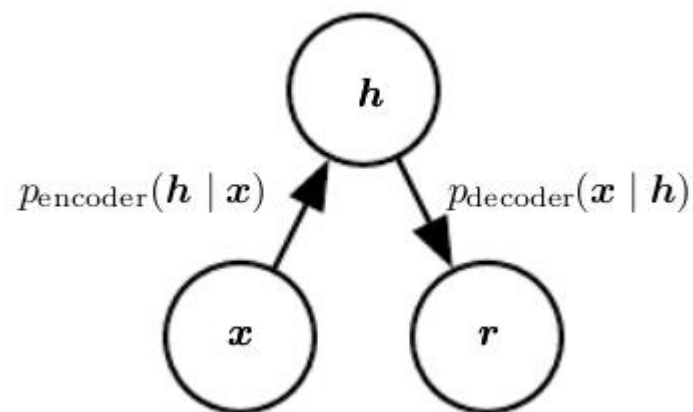
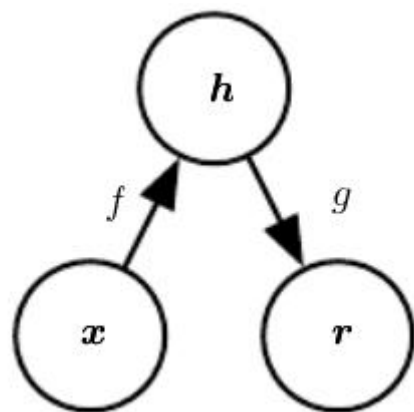
Sicong Liu

Content

- Autoencoder
- Regularized Autoencoder:
 - Sparse Autoencoder
 - Denoising Autoencoder
 - Contractive Autoencoder
- Learning Manifold with Autoencoders

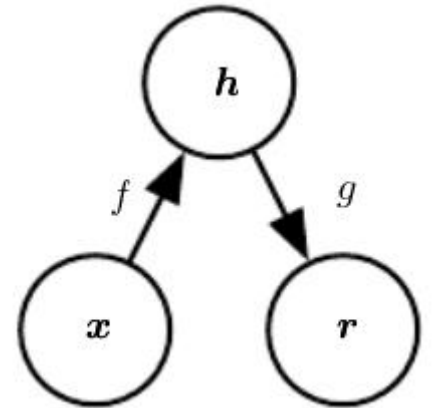
Autoencoder

- A neural network that attempt to copy input to output
- An encoder function $\mathbf{h} = f(\mathbf{x})$, or sotchastic mapping $p(\mathbf{h} | \mathbf{x})$
A decoder function $\mathbf{r} = g(\mathbf{h})$, or $p(\mathbf{x} | \mathbf{h})$



Design Idea

- Designed to be unable to learn to copy perfectly, Just learning a Identity function is useless
- Copy only input that resembles the training data, the encoder should capture useful features of the training data.



Undercomplete Autoencoder

- Let h to have less dimension than x (bottle-neck)
- When the decoder is linear and loss is the mean squared error, an undercomplete autoencoder span learns the same subspace as PCA.
- When encoder and decoder are too powerful, autoencoder can still learning an just copy network.

Regularized Autoencoder

Use a loss function to have other properties besides copying.

- Sparse Autoencoder: sparsity of the representation $\mathbf{h} = f(\mathbf{x})$
- Denoising Autoencoder: robustness to noise or to missing input
- Contractive Autoencoder: smallness of the derivative of the representation $\mathbf{h} = f(\mathbf{x})$

Sparse Autoencoder (SAE)

- Reconstructive Loss and Sparsity Regularizer

$$L(\mathbf{x}, g(f(\mathbf{x}))) + \Omega(\mathbf{h})$$

$$\Omega(\mathbf{h}) = \lambda \sum_i |h_i|$$

- Interpretation of the regularizer:

Weight decay regularizer $\Omega(\theta)$, a Gaussian prior over parameter θ

Sparsity regularizer, a Laplace prior over the activation \mathbf{h}

Connection to $\log Z$ in maximum likelihood

- For maximum likelihood: $p(\mathbf{x}) = \frac{1}{Z} \tilde{p}(\mathbf{x})$
minimizing $\log Z$ prevent $p(\mathbf{x})$ large everywhere
- Sparsity prevent h large everywhere

Sparse Autoencoder and Sparse coding

We can think of SAE as a **generative model** with latent variables, but the latent variable \mathbf{h} is the output of a parametric encoder rather than the result of an optimization.

So SAE could be optimized using Gradient Descent instead of EM.

Why the features \mathbf{h} learned by the autoencoder are useful: they describe the latent variables that explain the input.

Predictive Sparse Decomposition (PSD)

Training SAE just like Sparse coding:

- Alternates minimization with respect to \mathbf{h} and minimization with respect to the model.

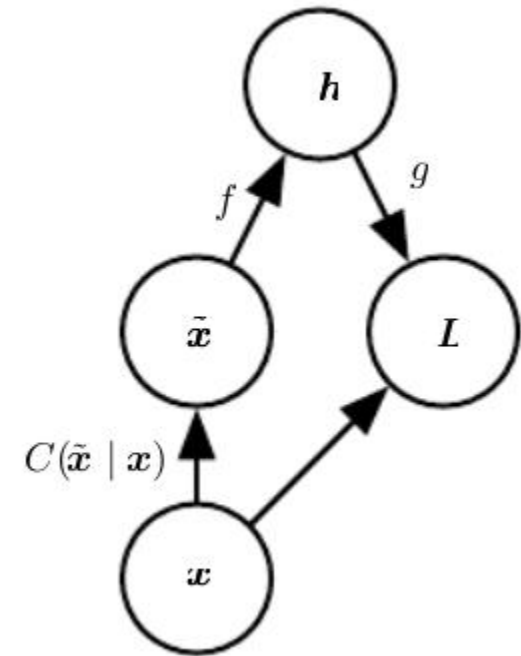
$$||\mathbf{x} - g(\mathbf{h})||^2 + \lambda ||\mathbf{h}||_1 + \gamma ||\mathbf{h} - f(\mathbf{x})||^2$$

- PSD is kind of learned approximate inference.

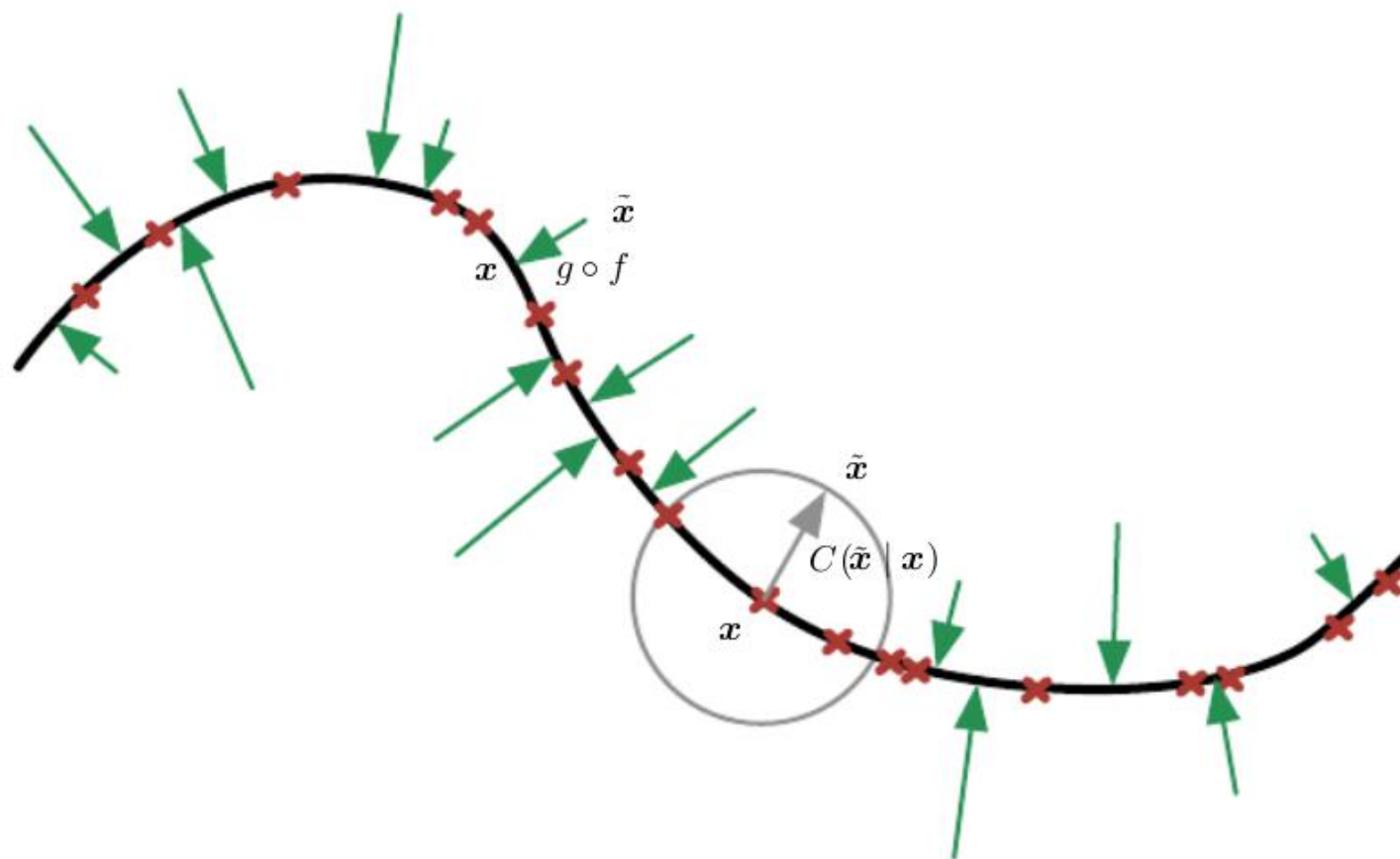
Denoising Autoencoder(DAE)

- DAE receives a corrupted data point as input and is trained to predict the original, uncorrupted data point as its output.
- $C(x' | x)$ is a corruption process: add noise or delete part of input.

$$- \mathbb{E}_{\mathbf{x} \sim \hat{p}_{\text{data}}(\mathbf{x})} \mathbb{E}_{\tilde{\mathbf{x}} \sim C(\tilde{\mathbf{x}} | \mathbf{x})} \log p_{\text{decoder}}(\mathbf{x} | \mathbf{h} = f(\tilde{\mathbf{x}}))$$



Example



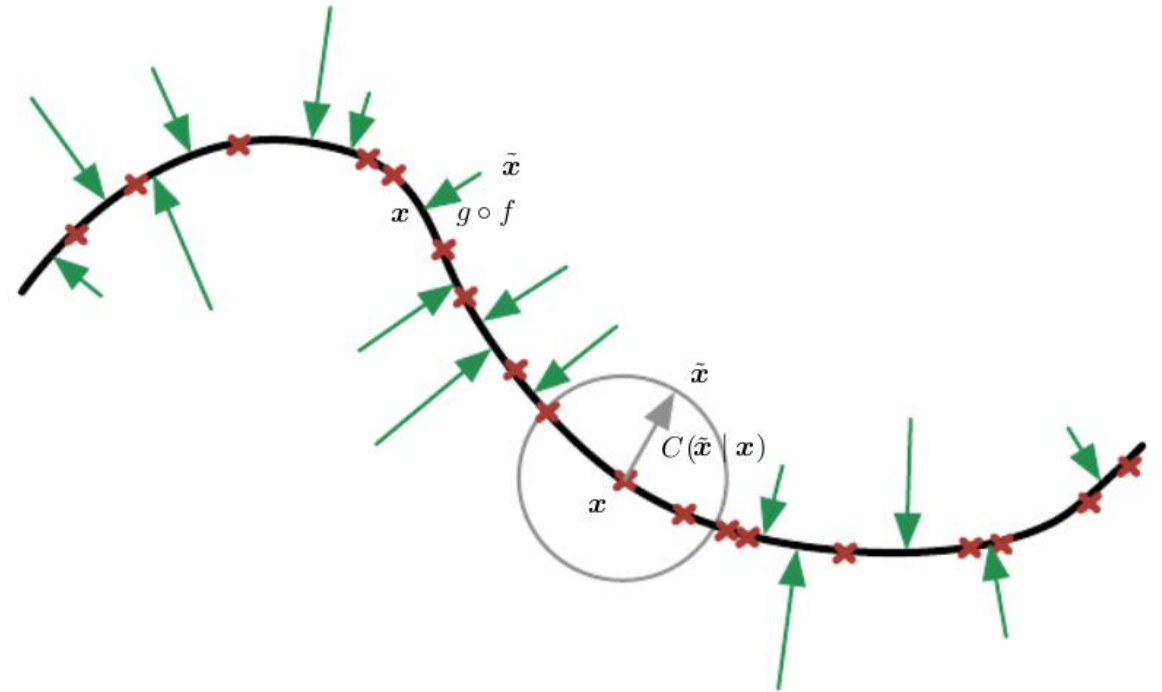
Score Matching

- an **alternative** to **maximum likelihood**
- encourage the model have same **score** as the data distribution at every training point \mathbf{x} .
- In this context, the score is a particular **gradient field**:

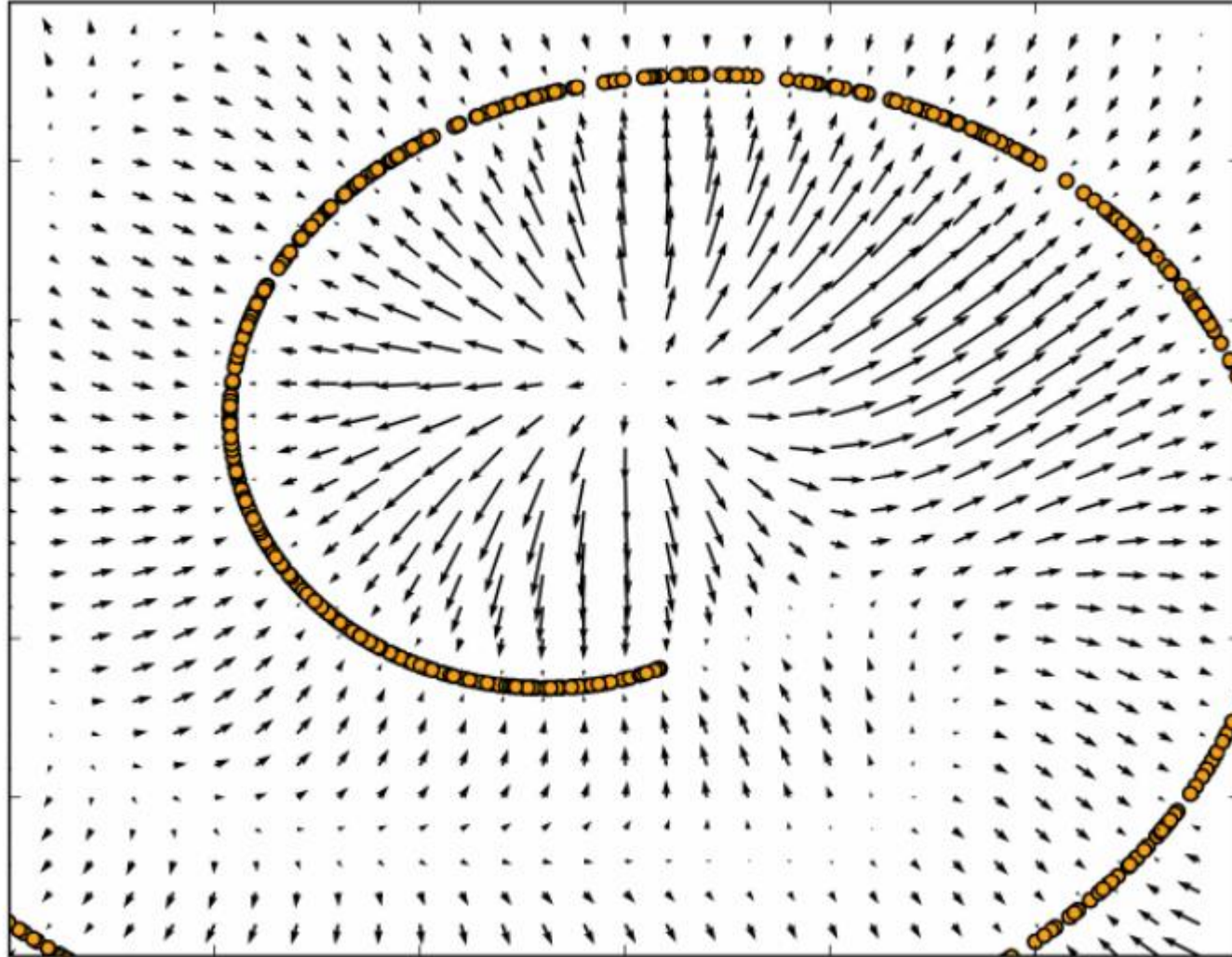
$$\nabla_{\mathbf{x}} \log p(\mathbf{x})$$

DAE and Score Matching

- DAE learning vector field $(g(f(x)) - x)$
- The vector field estimates the score of the data distribution



Vector field learned by DAE



Contractive Autoencoder (CAE)

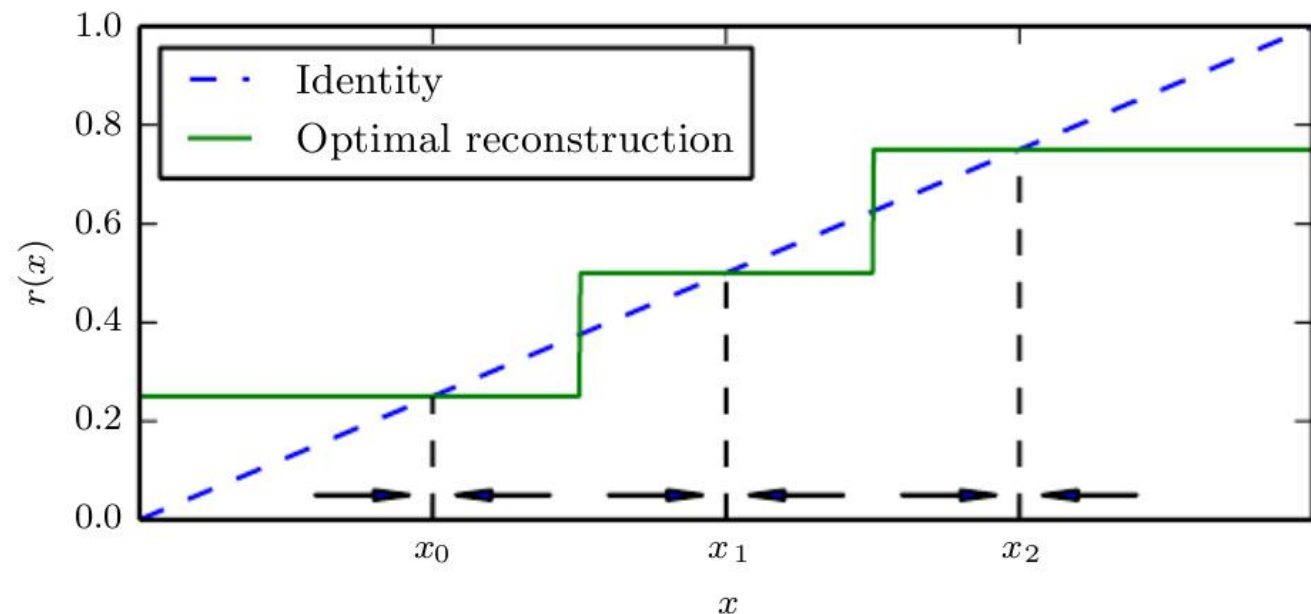
- encouraging the derivatives of encoder $f(\mathbf{x})$ to be as small as possible by a regularizer:

$$\Omega(\mathbf{h}) = \lambda \left\| \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} \right\|_F^2$$

- name **contractive** because CAE encourage to map two neighbor point more closer.

Contractive Autoencoder

- CAE **contractive** only **locally**, all perturbations of a training point x are mapped near to $f(x)$.
- Globally, two different point x and x' **may** be map farther than the original point.
- For sigmoidal units, CAE may make h saturate to 0 or 1.



DAE and CAE

- DAE make the **reconstruction function** ($g(f(x))$) resist **finite-sized perturbations** of the input.
- CAE make the **feature extraction function** ($f(x)$) resist **infinitesimal perturbations** of the input.

Practical issue with the CAE

Issue 1: It's **expensive** to compute the regularizer when autoencoder is **deep** because gradient is layer-wise related.

Solution: **separately** train a series of single-layer autoencoder

Practical issue with the CAE

Issue 2: The contraction penalty can obtain useless result if we do not constraint the scale of decoder. (Such as $f'(x)=\epsilon f(x)$, $g'(h)=g(h)/\epsilon$)

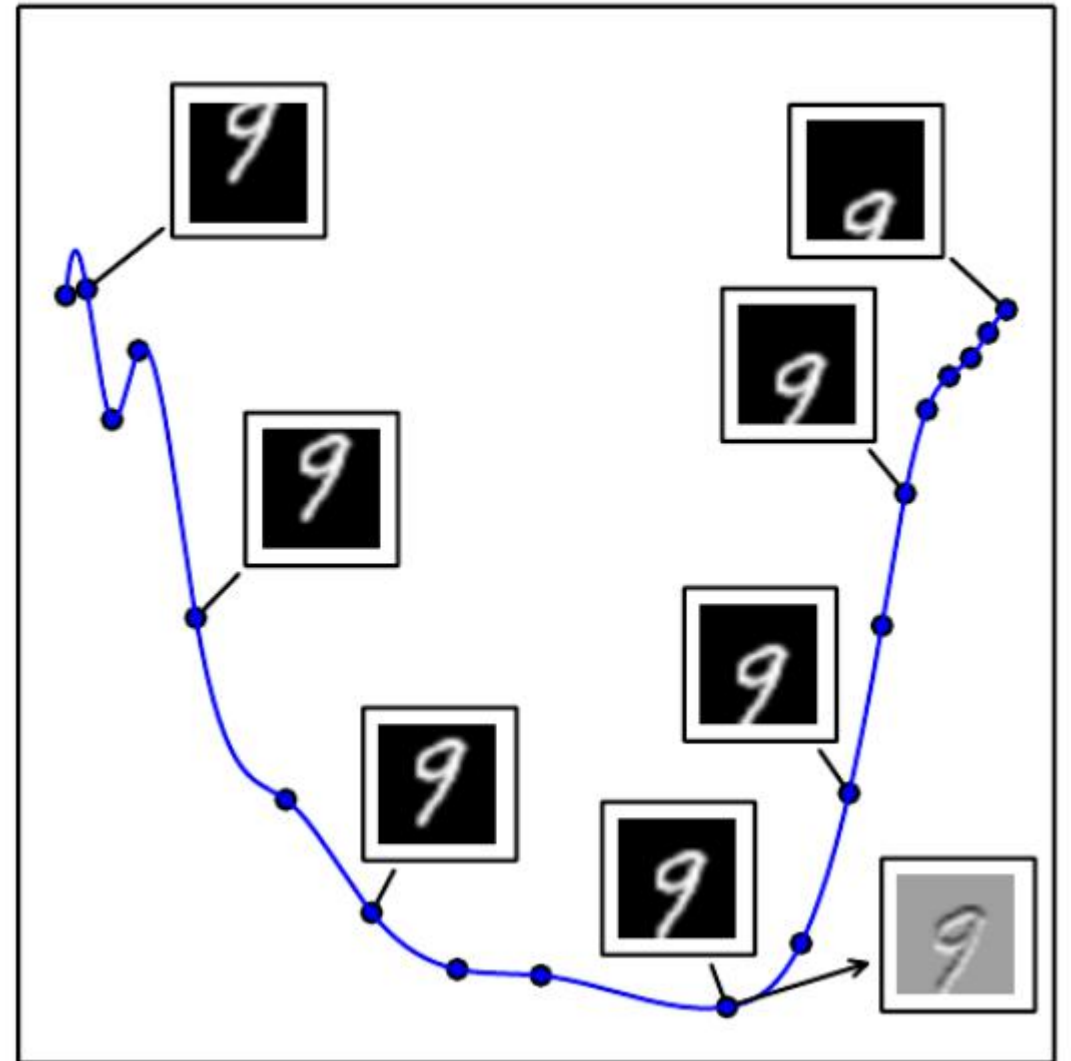
Solution: set the weight matrix of $g(h)$ to be the **transpose** of the weight matrix of $f(x)$.

A Characterizaion of Manifolds

- The set of manifolds' tangent planes
- At a point x on a d -dimensional manifold, the tangent plane is given by d basis vectors

An illustration of a **tangent hyperplane**.

- A one-dimension manifold given by transform a image vertically.
- Projected into two-dimension by PCA.
- Change x infinitesimally while staying on the manifold.



Learning Manifolds with Autoencoder

Training an autoencoder involve a compromise between two forces:

- Learning a representation h of a training example x such that x can be approximately recovered from h through a decoder.
- Satisfying the constraint or regularization penalty.

Learning Manifolds with Autoencoder

Copying the input to the output is not useful, nor is ignoring the input.

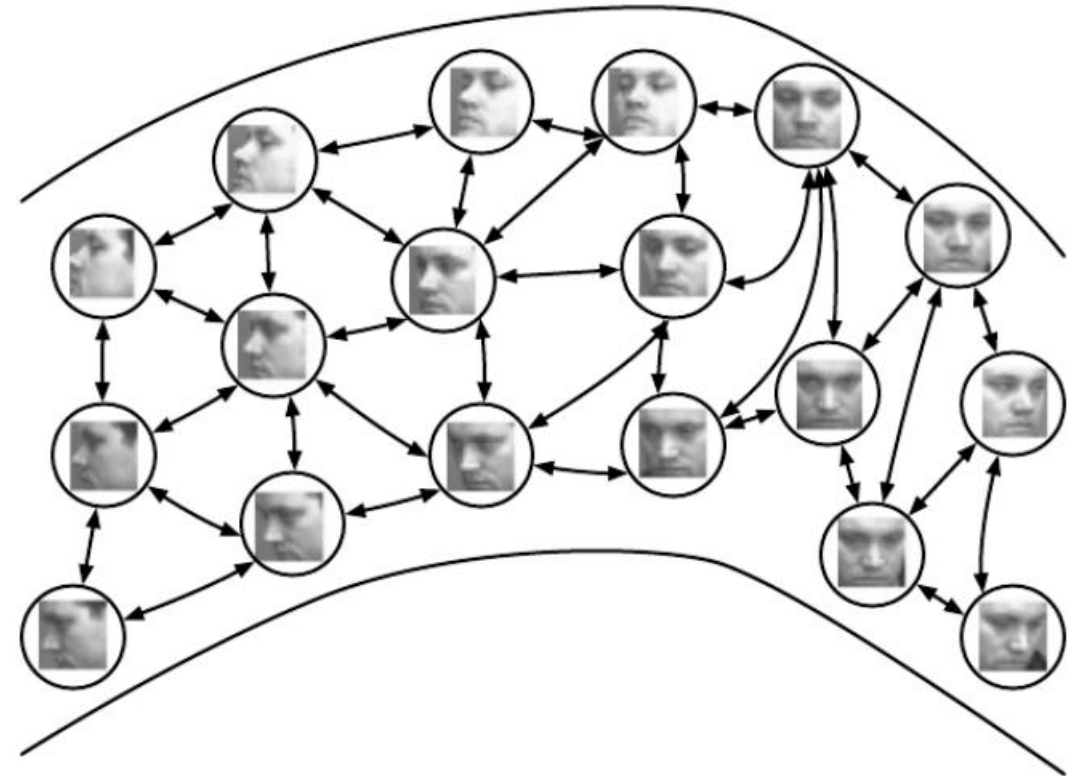
- The autoencoder can afford to represent only the variations that are needed to reconstruct training examples.
- The variations correspond to changes in $\mathbf{h} = f(\mathbf{x})$.

Learning Manifolds with non-parametric methods

based on the nearest-neighbor graph.

For each point x and its neighbors, we could get a local coordinate system.

By an optimization or solving a linear system, we could get a global coordinate system.



Learning Manifolds with non-parametric methods

- To get representation to a new example, a form of interpolation may be used.
- But if the manifolds are not very smooth, one may need a very large number of example to cover each one of all variations.
- And interpolation in high-dimension space may be meaningless.

Applications of Autoencoders

- Dimensionality reduction
- As semantic hashing on information retrieval
- Semi-supervised learning
- Pretrain for deep model (no longer needed now)

Thanks