UNIVERSITÀ PARTHENOPE
DiST
Dipartimento di Scienze e Tecnologie

Machine Learning (part II)

Autoencoders

Angelo Ciaramella

Introduction

- Autoencoder
  - neural network that is trained to attempt to copy its input to its output
- Modern autoencoders
  - have generalized the idea of an encoder and a decoder beyond deterministic functions to stochastic mappings

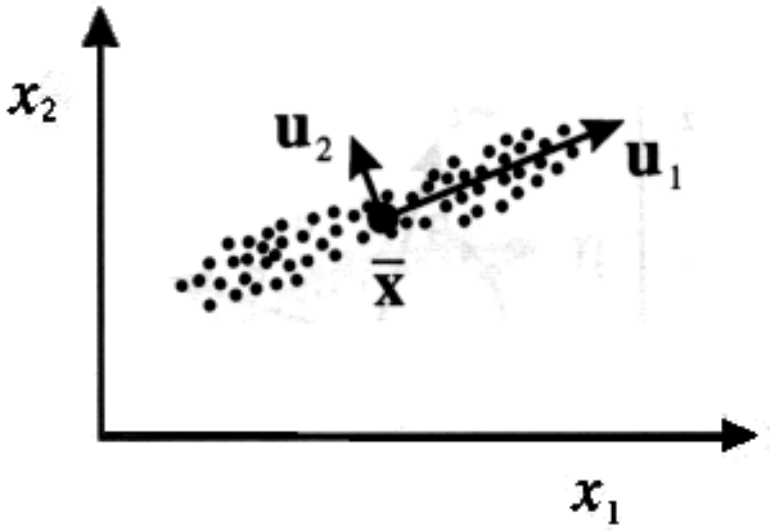
Principal Component Analysis

- Principal Component Analysis (PCA) is a statistical technique
  - Dimensionality reduction
  - Lossy data compression
  - Feature extraction
  - Data visualization
- It is also known as the *Karhunen-Loeve* transform
- PCA can be defined as the principal subspace such that the variance of the projected data is maximized

Second-Order methods

- The second-order methods are the most popular methods to find a linear transformation
- This methods find the representation using only the information contained in the covariance matrix of the data vector **x**
- PCA is widely used in signal processing, statistics, and neural computing

Principal Components



In a linear projection down to one dimension, the optimum choice of projection, in the sense of minimizing the sum-of-squares error, is obtained first subtracting off the mean of the data set, and then projecting onto the first eigenvector $\mathbf{u}_1$ of the covariance matrix.

Projection error minimization

- We introduce a complete orthonormal set of *D*-dimensional basis vectors (i=1,…,*D*)

$$\mathbf{u}_i^T \mathbf{u}_j = \delta_{ij}$$

- Because this basis is complete, each data point can be represented by a linear combination of the basis vectors

$$\mathbf{x}_n = \sum_{i=1}^{D} \alpha_{ni} \mathbf{u}_i$$

Projection error minimization

- We can write also that

$$\mathbf{x}_n = \sum_{i=1}^{D} \left(\mathbf{x}_n^T \mathbf{u}_i \right) \mathbf{u}_i \qquad \alpha_{nj} = \mathbf{x}_n^T \mathbf{u}_j$$

- Our goal is to approximate this data point using a representation involving a restricted number $M <$ D of variables corresponding to a projection onto a lower-dimensional subspace

$$\tilde{\mathbf{x}}_n = \sum_{i=1}^{M} z_{ni} \mathbf{u}_i + \sum_{i=M+1}^{D} b_i \mathbf{u}_i$$

Projection error minimization

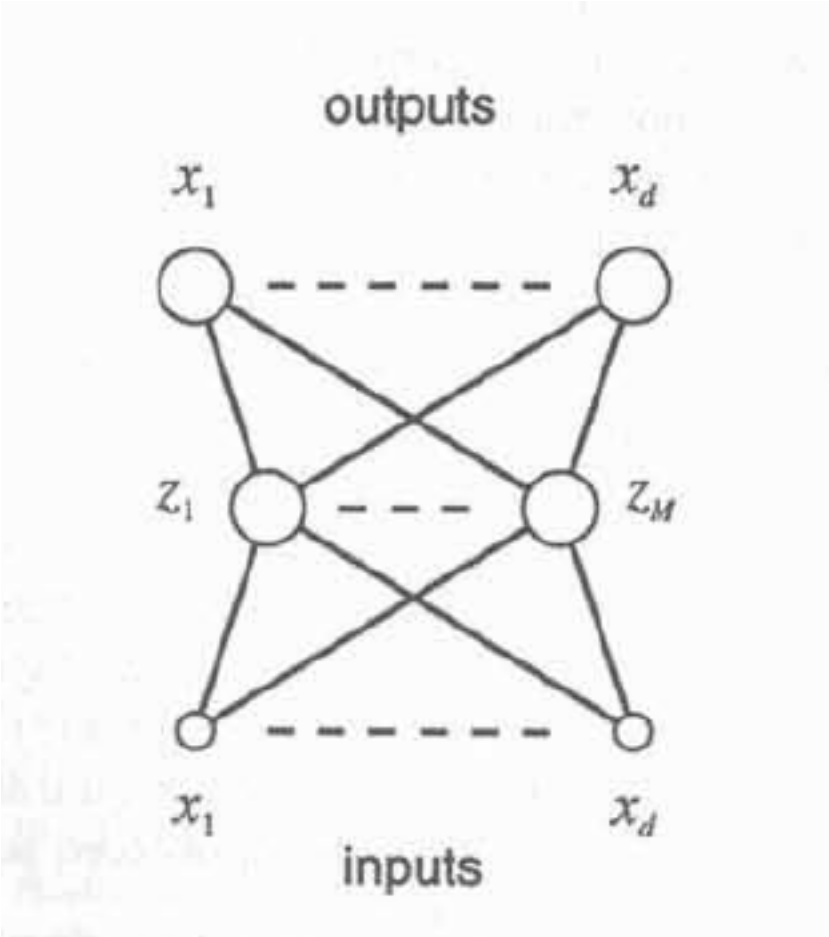
- As our distortion measure we shall use the squared distance between the original point and its approximation averaged over the data set so that our goal is to minimize

$$J = \frac{1}{N}\sum_{n=1}^{N}\left\|\mathbf{x}_n - \tilde{\mathbf{x}}_n\right\|^2$$

- The general solution is obtained by choosing the basis to be eigenvectors of the covariance matrix given by

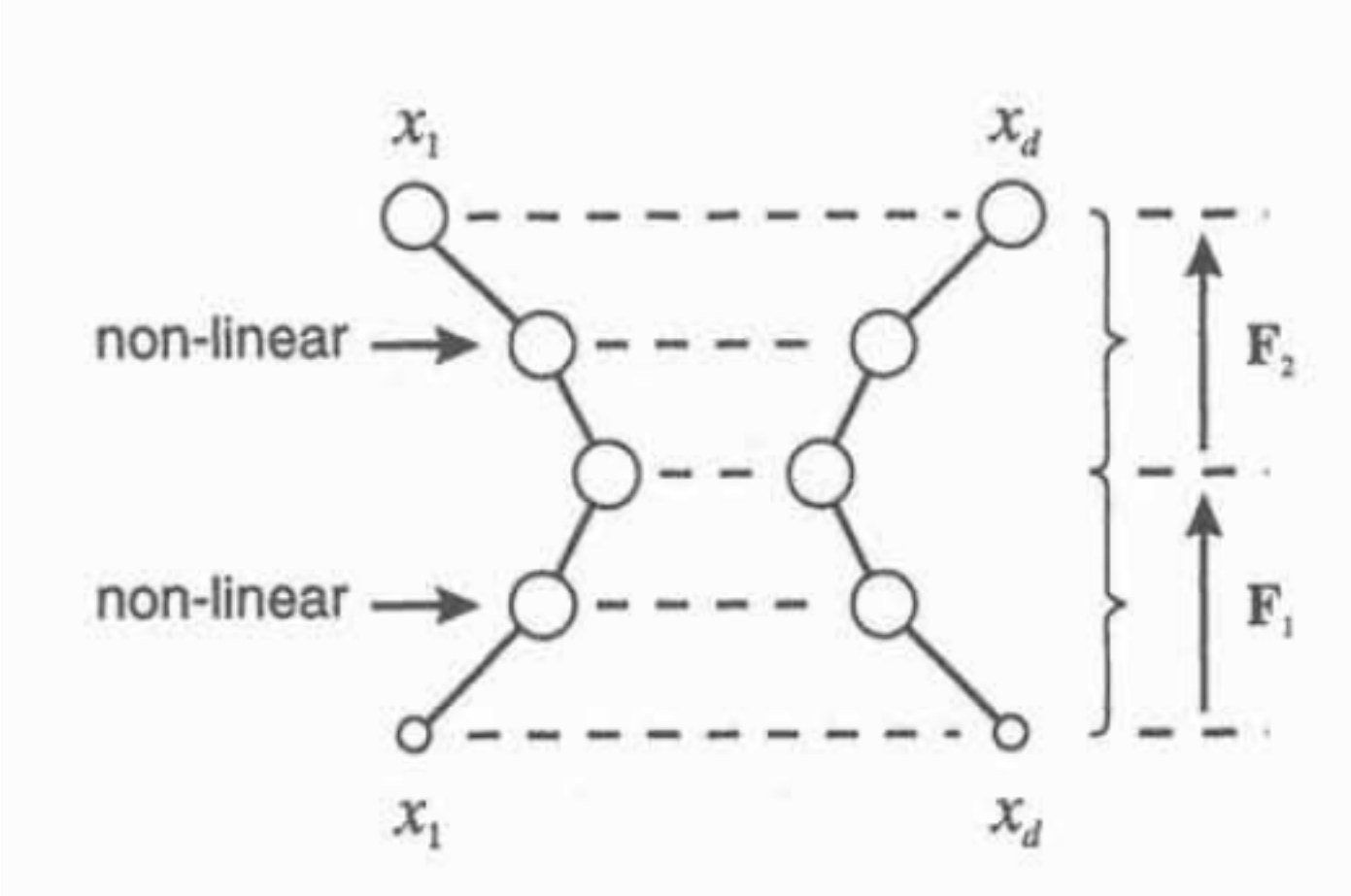
$$\mathbf{S}\mathbf{u}_i = \lambda_i \mathbf{u}_i$$

Auto-associative MLP



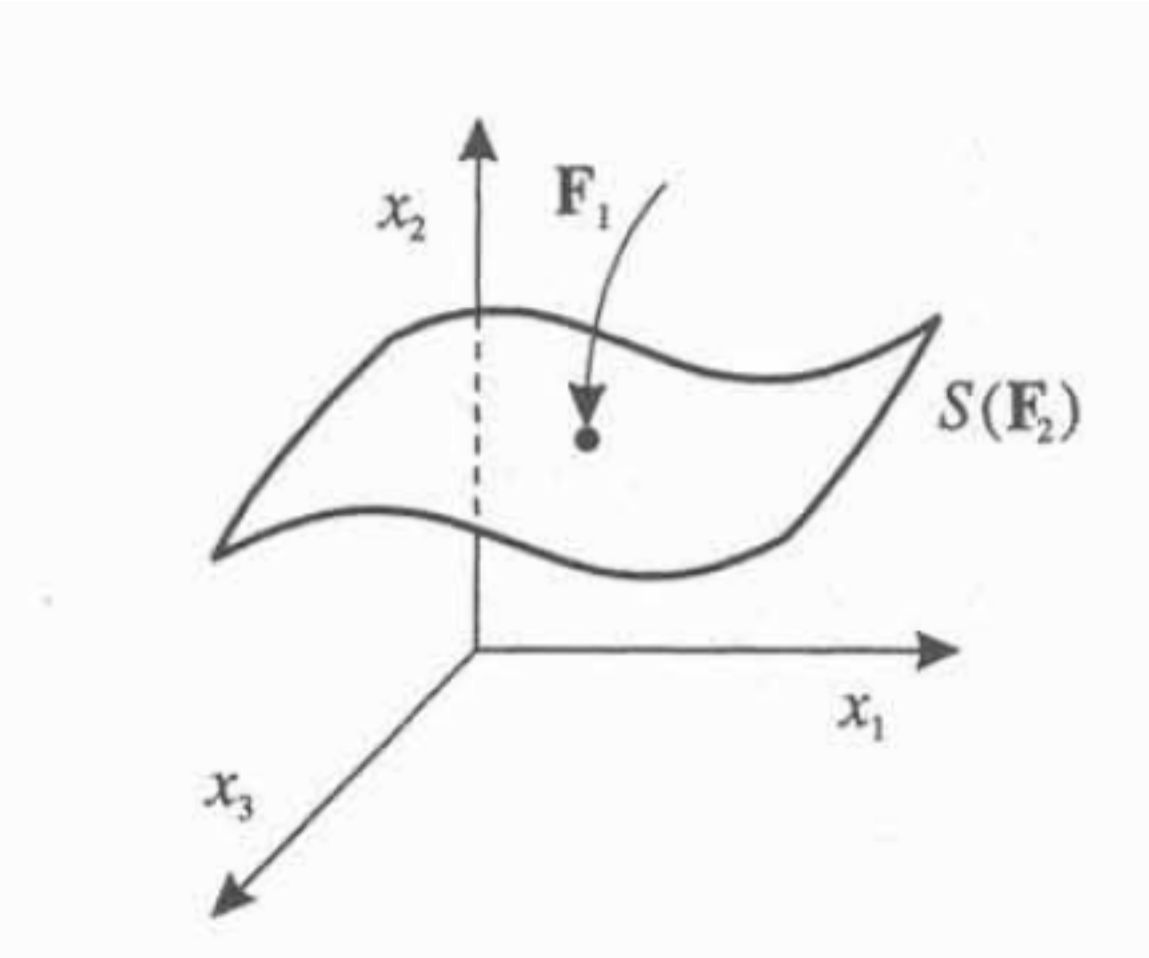
$$E = \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{d}\{y_k(\mathbf{x}^n) - x_k^n\}^2.$$

Auto-associative MPL



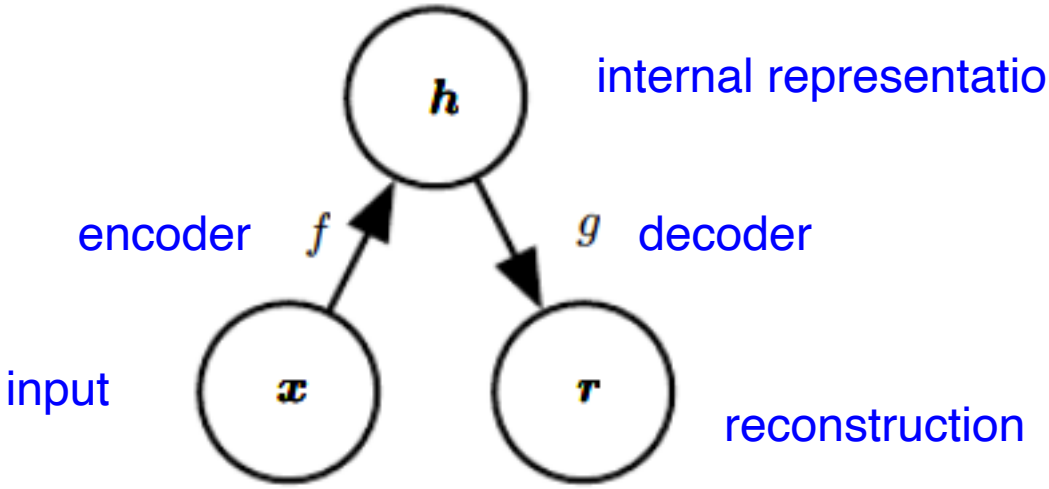
Non-liner units

Auto-associative MLP



Mapping of the non-linear auto-associative MLP

Autoencoder



General structure of an autoencoder

Autoencoder

- Undercomplete autoencoder
  - code dimension is less than the input dimension
  - representation forces the autoencoder to capture the most salient features of the training data
- Loss function

$$L(\boldsymbol{x}, g(f(\boldsymbol{x})))$$

- PCA subspace
  - decoder linear
  - L is mean squared error

Regularized autoencoder

- Regularized autoencoders
  - use a loss function that encourages the model to have other properties besides the ability to copy its input to its output
  - Examples
    - Sparsity of the representation
    - Smallness of the derivative of the representation
    - Robustness to noise
    - Robusstness to missing inputs
    - Generative models
      - Helmholtz machine
      - Variatiaonal autonecoder

Sparse autoencoder

- Sparse penalty

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

$$\boldsymbol{h} = f(\boldsymbol{x})$$ typically

- used to learn features for another task such as classification

Sparse autoencoder

- Joint distribution

$$p_{\text{model}}(\boldsymbol{x}, \boldsymbol{h}) = p_{\text{model}}(\boldsymbol{h}) p_{\text{model}}(\boldsymbol{x} \mid \boldsymbol{h})$$

- Log-likelihood

$$\log p_{\text{model}}(\boldsymbol{x}) = \log \sum_{\boldsymbol{h}} p_{\text{model}}(\boldsymbol{h}, \boldsymbol{x})$$

- Maximizing

$$\log p_{\text{model}}(\boldsymbol{h}, \boldsymbol{x}) = \log p_{\text{model}}(\boldsymbol{h}) + \log p_{\text{model}}(\boldsymbol{x} \mid \boldsymbol{h})$$

Sparse autoencoder

- e.g., Laplace prior

$$p_{\text{model}}(h_i) = \frac{\lambda}{2} e^{-\lambda |h_i|}$$

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

$$-\log p_{\text{model}}(\boldsymbol{h}) = \sum_i \left(\lambda |h_i| - \log \frac{\lambda}{2} \right) = \Omega(\boldsymbol{h}) + \text{const}$$

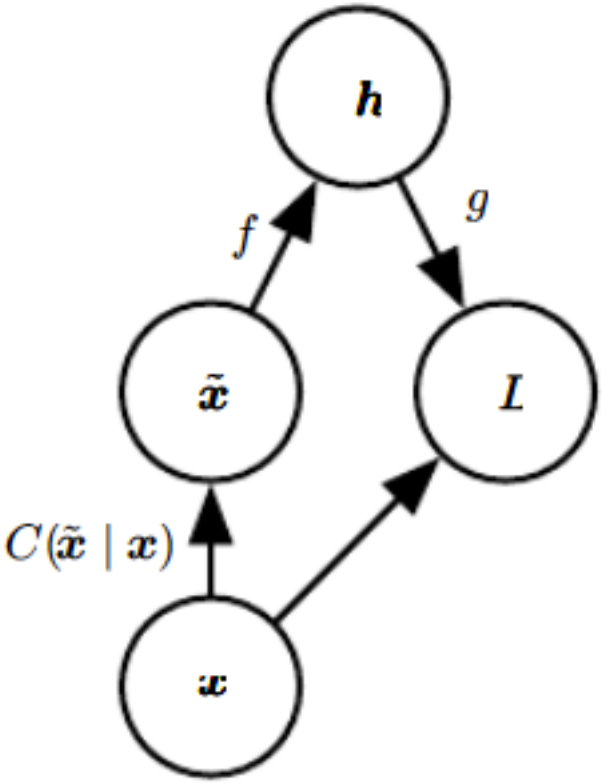
Way of approximately training a generative model

Denoising autoencoder

- Minimization

$$L(\boldsymbol{x}, g(f(\tilde{\boldsymbol{x}})))$$

Noisy version of x

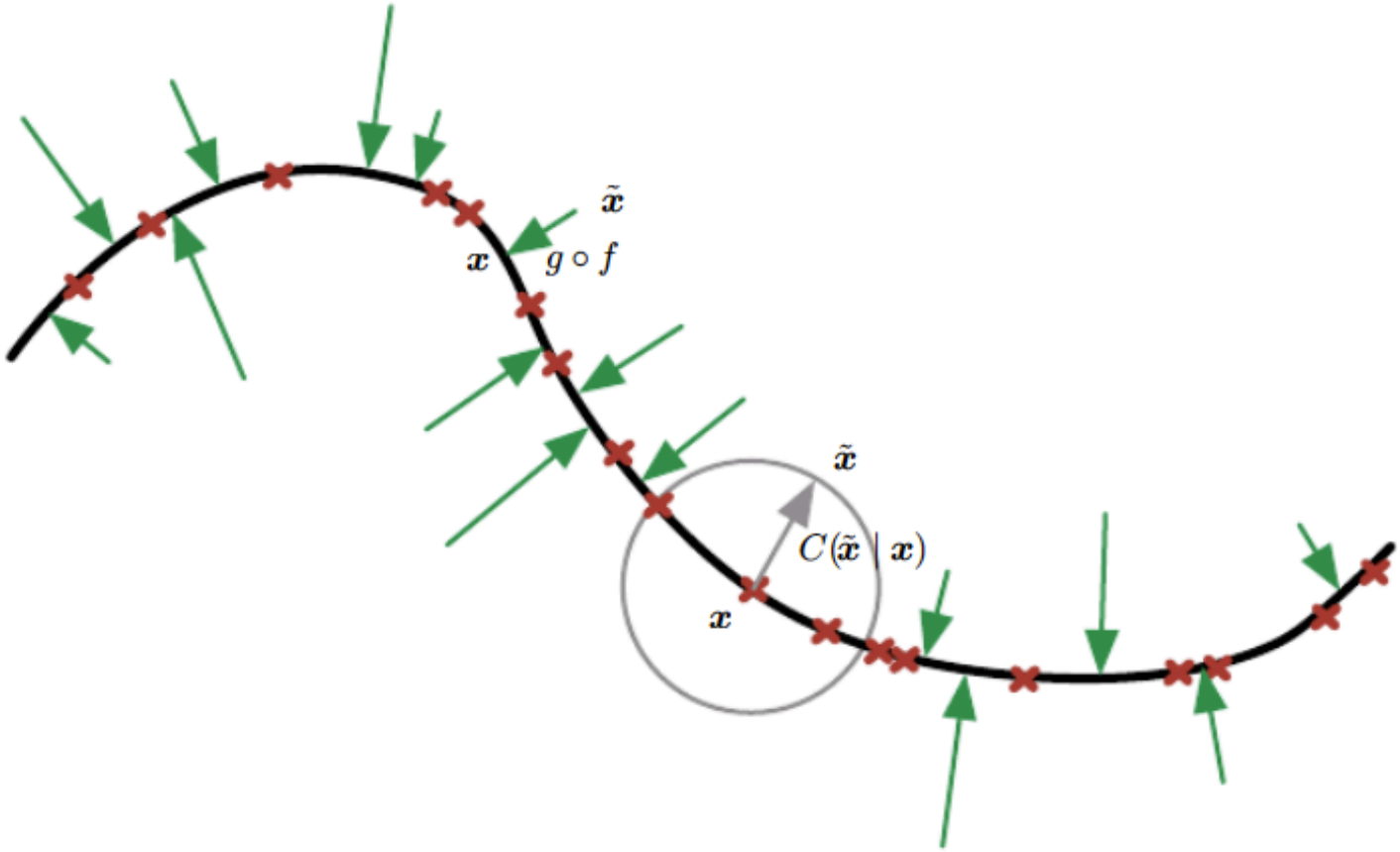


Denoising autoencoder

1. Sample a training example $\boldsymbol{x}$ from the training data.
2. Sample a corrupted version $\tilde{\boldsymbol{x}}$ from $C(\tilde{\mathbf{x}} \mid \mathbf{x} = \boldsymbol{x})$.
3. Use $(\boldsymbol{x}, \tilde{\boldsymbol{x}})$ as a training example for estimating the autoencoder reconstruction distribution $p_{\text{reconstruct}}(\boldsymbol{x} \mid \tilde{\boldsymbol{x}}) = p_{\text{decoder}}(\boldsymbol{x} \mid \boldsymbol{h})$ with $\boldsymbol{h}$ the output of encoder $f(\tilde{\boldsymbol{x}})$ and p_{decoder} typically defined by a decoder $g(\boldsymbol{h})$.

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

Denoising autoencoder



denoising autoencoder is trained to map a corrupted data

Penalizing Derivatives

- Minimization

contractive autoencoder

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

$$\Omega(\boldsymbol{h}, \boldsymbol{x}) = \lambda \sum_i ||\nabla_{\boldsymbol{x}} h_i||^2$$

Forces the autoencoder to learn features that capture information about the training distribution

Depth

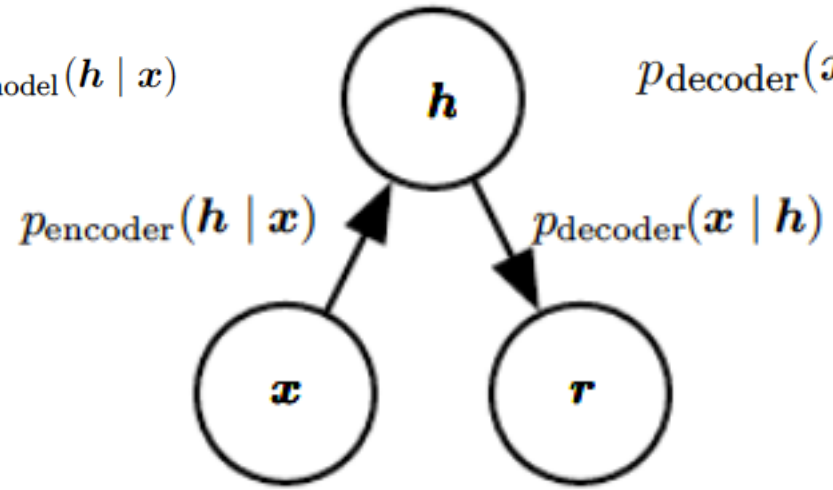
- Depth
  - using deep encoders and decoders offers many advantages
  - One major advantage of non-trivial depth is that the universal approximator theorem guarantees
  - yield much better compression than corresponding shallow or linear autoencoders

Stochastic autoencoder

- Depth

$p_{\text{encoder}}(\boldsymbol{h} \mid \boldsymbol{x}) = p_{\text{model}}(\boldsymbol{h} \mid \boldsymbol{x})$

$p_{\text{decoder}}(\boldsymbol{x} \mid \boldsymbol{h}) = p_{\text{model}}(\boldsymbol{x} \mid \boldsymbol{h})$



Both the encoder and the decoder are not simple functions but instead involve some noise injection, meaning that their output can be seen as sampled from a distribution

Contractive atuoencoders

- Encouraging the derivatives of f to be as small as possible

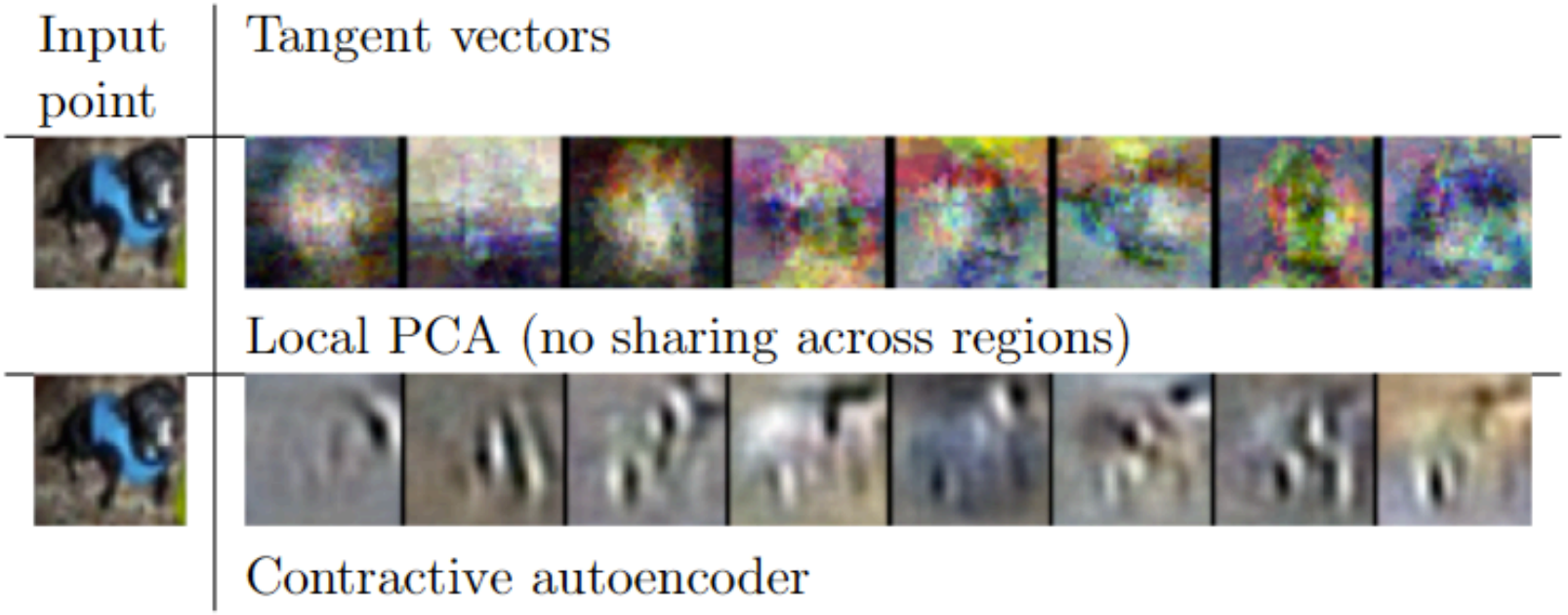
Jacobian matrix

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

Frobenius norm (sum of squared elements)

Contractive atuoencoders

largest singular values are interpreted as the tangent directions that the contractive autoencoder has learned



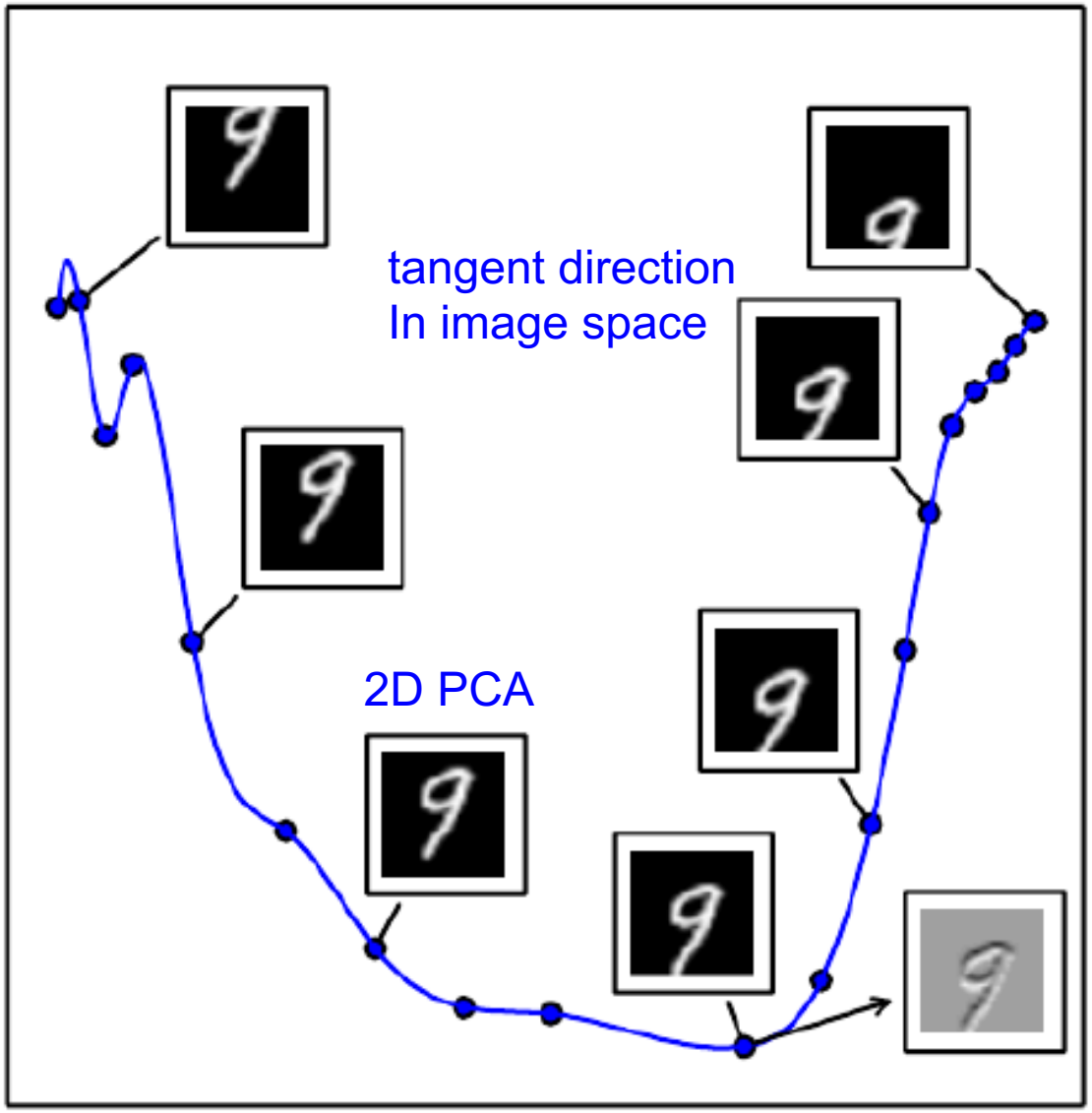
learn tangent vectors that show how the image changes as objects in the image gradually change pose

Manifold learning

- data concentrates around a low-dimensional manifold

- tangent planes
  - data concentrates around a low-dimensional manifold
  - At a point x on a d-dimensional manifold, the tangent plane is given by d basis vectors that span the local directions of variation allowed on the manifold

Tangent plane

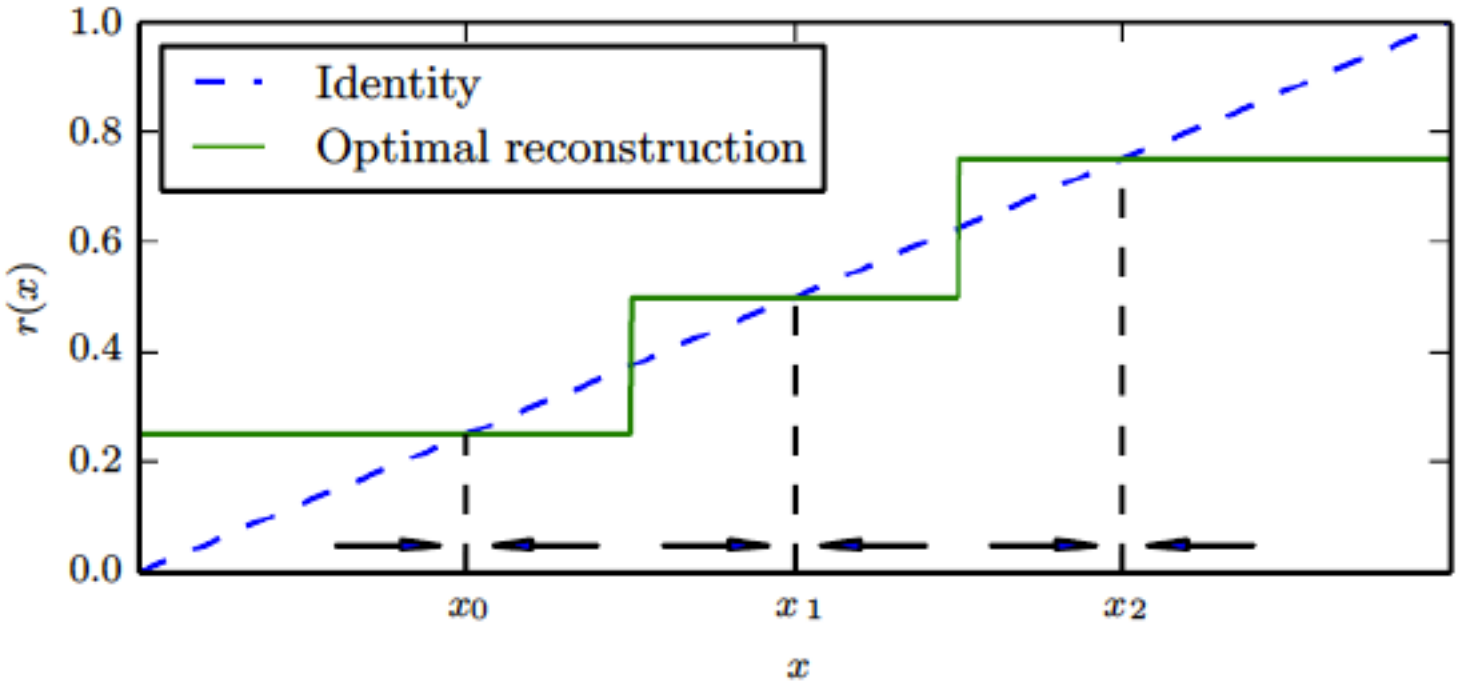


one-dimensional manifold
in 784-dimensional space

translating it vertically

vertical translation
defines a coordinate
along a one-dimensional
manifold

Reconstruction



If the autoencoder learns a reconstruction function that is invariant to small perturbations near the data points, it captures the manifold structure of the data.

Variational Autoencoders

- Inference as optimization
  - Approximate inference algorithms may then be derived by approximating the underlying optimization problem
  - Aim

$$\log p(\boldsymbol{v}; \boldsymbol{\theta})$$

- Evidence lower bound (ELBO)

$$\mathcal{L}(\boldsymbol{v}, \boldsymbol{\theta}, q) = \log p(\boldsymbol{v}; \boldsymbol{\theta}) - D_{\mathrm{KL}}\left(q(\boldsymbol{h} \mid \boldsymbol{v}) \| p(\boldsymbol{h} \mid \boldsymbol{v}; \boldsymbol{\theta})\right)$$

$\boldsymbol{v}$ observed variables

$\boldsymbol{h}$ latent variables

q arbitrary probability distribution

Variational Autoencoders

- Canonical defintion

$$\mathcal{L}(\boldsymbol{v}, \boldsymbol{\theta}, q) = \mathbb{E}_{\mathbf{h} \sim q} \left[\log p(\boldsymbol{h}, \boldsymbol{v}) \right] + H(q)$$

$$\begin{aligned}
\mathcal{L}(\boldsymbol{v}, \boldsymbol{\theta}, q) &= \log p(\boldsymbol{v}; \boldsymbol{\theta}) - D_{\mathrm{KL}}(q(\boldsymbol{h} \mid \boldsymbol{v}) \| p(\boldsymbol{h} \mid \boldsymbol{v}; \boldsymbol{\theta})) \\
&= \log p(\boldsymbol{v}; \boldsymbol{\theta}) - \mathbb{E}_{\mathbf{h} \sim q} \log \frac{q(\boldsymbol{h} \mid \boldsymbol{v})}{p(\boldsymbol{h} \mid \boldsymbol{v})} \\
&= \log p(\boldsymbol{v}; \boldsymbol{\theta}) - \mathbb{E}_{\mathbf{h} \sim q} \log \frac{q(\boldsymbol{h} \mid \boldsymbol{v})}{\frac{p(\boldsymbol{h}, \boldsymbol{v}; \boldsymbol{\theta})}{p(\boldsymbol{v}; \boldsymbol{\theta})}} \\
&= \log p(\boldsymbol{v}; \boldsymbol{\theta}) - \mathbb{E}_{\mathbf{h} \sim q} \left[\log q(\boldsymbol{h} \mid \boldsymbol{v}) - \log p(\boldsymbol{h}, \boldsymbol{v}; \boldsymbol{\theta}) + \log p(\boldsymbol{v}; \boldsymbol{\theta}) \right] \\
&= -\mathbb{E}_{\mathbf{h} \sim q} \left[\log q(\boldsymbol{h} \mid \boldsymbol{v}) - \log p(\boldsymbol{h}, \boldsymbol{v}; \boldsymbol{\theta}) \right].
\end{aligned}$$

Variational Autoencoders

- Varational Autoenoders

$p_{\text{model}}(\boldsymbol{z})$

distribution

$g(\boldsymbol{z})$

generator network

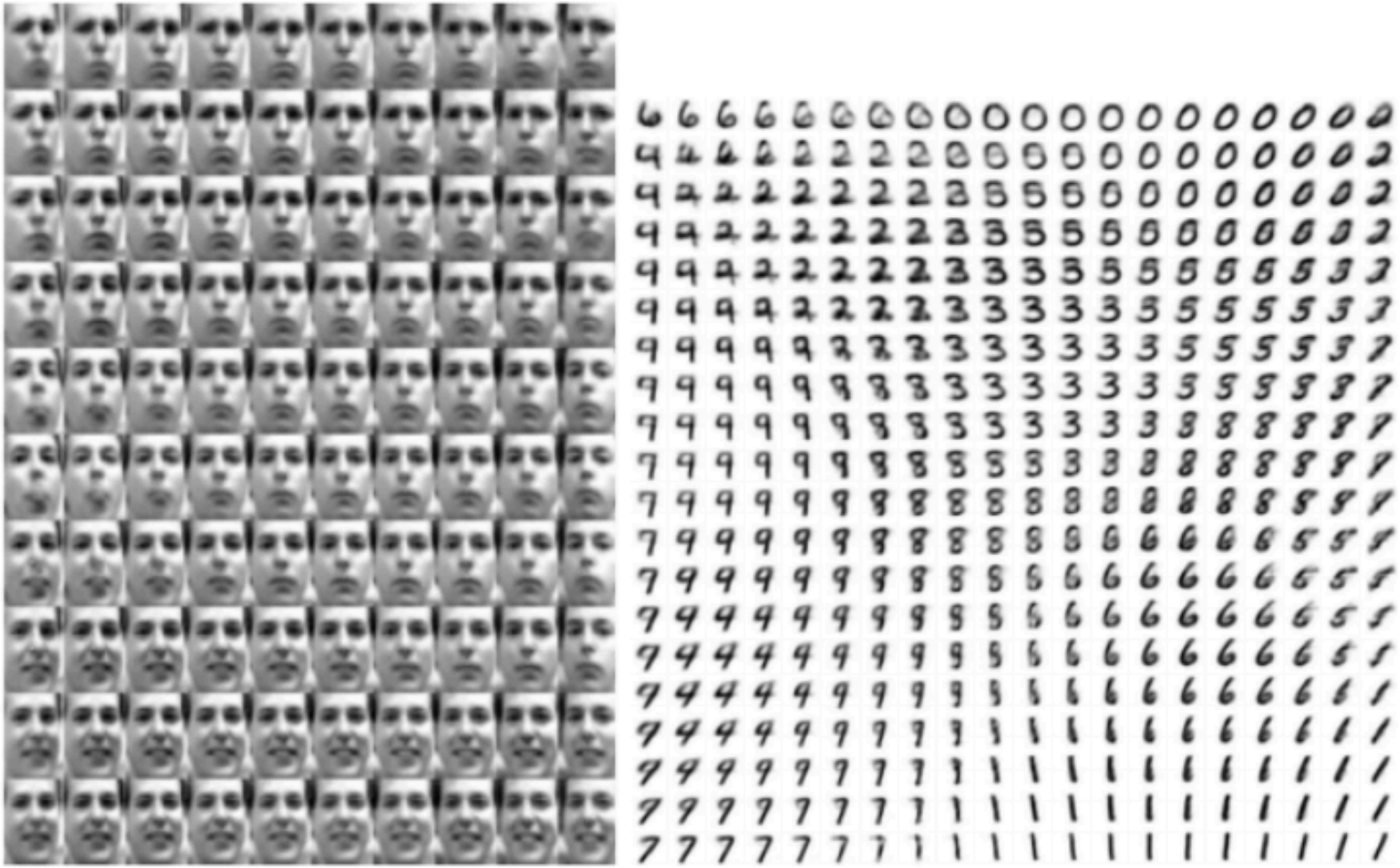
joint log-likelihood

$$\begin{aligned}\mathcal{L}(q) &= \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z}|\boldsymbol{x})} \log p_{\text{model}}(\boldsymbol{z}, \boldsymbol{x}) + \mathcal{H}(q(\mathbf{z} \mid \boldsymbol{x})) \\ &= \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z}|\boldsymbol{x})} \log p_{\text{model}}(\boldsymbol{x} \mid \boldsymbol{z}) - D_{\text{KL}}(q(\mathbf{z} \mid \boldsymbol{x})||p_{\text{model}}(\mathbf{z})) \\ &\leq \log p_{\text{model}}(\boldsymbol{x}).\end{aligned}$$

Variational Autoencoders

- Training
  - train a parametric encoder that produces the parameters of q
  - maximizing L with respect to the parameters of the encoder and decoder
  - All of the expectations in L may be approximated by Monte Carlo sampling

Interesting properties



algorithm discovered two independent factors of variation present in images of faces: angle of rotation and emotional expression