EE269
Signal Processing for Machine Learning

Lecture 17

Instructor : Mert Pilanci

Stanford University

November 19, 2020

Principal Component Analysis

The idea behind PCA is to approximate

$$x_i \approx \mu + \boldsymbol{A}\theta_i$$

where

$$\mu \in \mathbb{R}^d$$

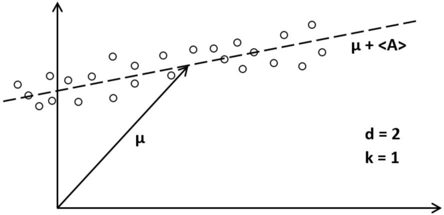
$$\boldsymbol{A} \in A_k$$

$$\theta_i \in \mathbb{R}^k$$

Mathematically, we define $\mu, \boldsymbol{A}, \theta_1, \cdots, \theta_n$ to be the solution of

$$\min_{\substack{\mu \in \mathbb{R}^d \\ \boldsymbol{A} \in A_k \\ \theta_i \in \mathbb{R}^k}} \quad \sum_{i=1}^{n} \|x_i - \mu - \boldsymbol{A}\theta_i\|^2$$

Principal Component Analysis



Principal Component Analysis

$$\min_{\substack{\mu \in \mathbb{R}^d \\ \boldsymbol{A} \in A_k \\ \theta_i \in \mathbb{R}^k}} \quad \sum_{i=1}^{n} \|x_i - \mu - \boldsymbol{A}\theta_i\|^2$$

$$S = \frac{1}{n} \sum_{i=1}^{n} (x_i - \overline{x})(x_i - \overline{x})^T$$

$$S = U \Lambda U^T$$

$$U = [u_1 \cdots u_d] \text{ and } \lambda = \begin{bmatrix} \lambda_1 \cdots 0 \\ \vdots \\ 0 \cdots \lambda_d \end{bmatrix}$$

- solution to PCA:

$$\begin{aligned} \mu &= \overline{x} \\ \boldsymbol{A} &= [u_1 \cdots u_k] \\ \theta_i &= \boldsymbol{A}^T (x - \overline{x}) \end{aligned}$$

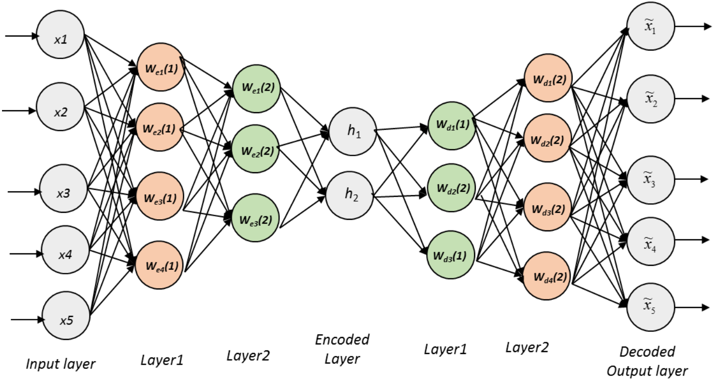
Autoencoder Networks

- L-layer neural network $f_\Theta(X) = \phi(\phi(XW_1)W_2)\ldots W_L$
  nonlinear activation $\phi(u)$, e.g., $\phi(u) = \max(0, u)$
- data matrix $X \quad : n \times d$
- layer weights Θ : $W_1 \in \mathbb{R}^{d\times m_1}$,...,$W_L \in \mathbb{R}^{m_{L-1}\times d}$
- autoencoders approximate the data as $X \approx f(X)$
- training problem

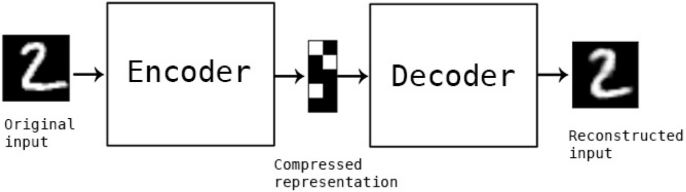
$$\min_\Theta \ell_\Theta(f(X), X) + \beta \sum_{i=1}^{L} \|W_i\|_F^2$$

- L_2 loss $\ell(f(X), X) = \|f(X) - X\|_F^2$
- L_1 loss $\ell(f(X), X) = \|f(X) - X\|_1$

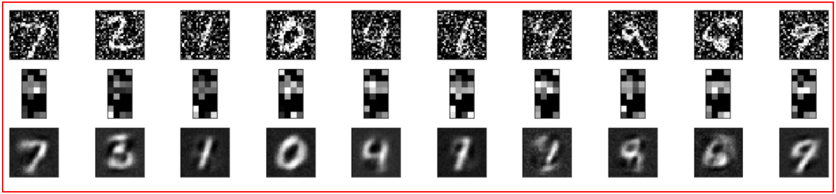
Autoencoders



Compressed representation



MNIST

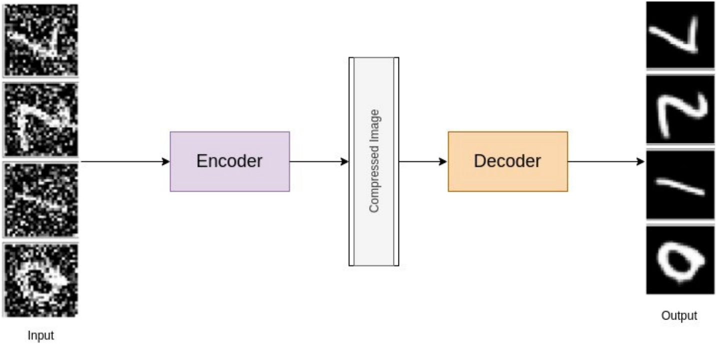


Denoising Autoencoders

noise is added to the input of the network

N : random noise, e.g. $N_{ij} \sim N(0, \sigma^2)$

$$\min_{\Theta} \ell_{\Theta}(f(X+N), X) + \beta \sum_{i=1}^{L} \|W_i\|_F^2$$



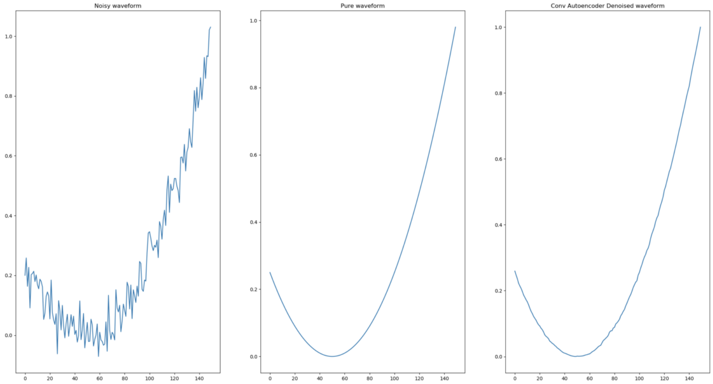
Denoising Autoencoders

length 150 signal $x[n] = n^2$ reshaped into 15×10 image

`Conv2D(3,3,128)-ReLU-Conv2D(32)-ReLU-Conv2DTranspose(32)-ReLU-Conv2DTranspose(128)-Conv2D(1)`

trainable parameters: 85,569 (hugely overparameterized!)

train with 56000 samples



Principal Component Analysis and Neural Networks

- two-layer neural network $f(x) = \phi(x^T W_1) W_2$
  linear activation $\phi(u) = u$
- data matrix $X \quad : n \times d$
- layer 1 weights $W_1 \quad : d \times m$
- layer 2 weights $W_2 \quad : m \times d$
  autoencoder : targets $Y = X$
- training problem

$$\min_{W_1, W_2} \|X W_1 W_2 - X\|_F^2 + \beta \|W_1\|_F^2 + \beta \|W_2\|_F^2$$

Principal Component Analysis and Autoencoder Networks

layer 1 weights W_1 $\quad : d \times m$

layer 2 weights W_2 $\quad : m \times d$

- no regularization ($\beta = 0$)

$$= \min_{W_1, W_2} \|XW_1W_2 - X\|_F^2$$
$$= \min_{W : \mathrm{rank}(W) \leq m} \|XW - X\|_F^2$$

- Optimal solution is $V_{1:m}V_{1:m}^T$
  where $X = U\Sigma V^T$ is the SVD of X and $V_{1:m}$ are the top k right singular vectors
- PCA = linear activation neural autoencoder

Principal Component Analysis and Autoencoder Networks

- Back to the more general regularized case

$$= \min_{W_1} \min_{W_2} \frac{1}{2} \| \sum_{j=1}^{m} X W_{1j} W_{2j}^T - Y \|_F^2 + \beta \sum_{j=1}^{m} \|W_{1j}\|_2^2 + \|W_{2j}\|_2^2$$

- rescale $W_{1j} \leftarrow W_{1j}/\alpha_j$, $W_{2j} \leftarrow W_{2j}\alpha_j$

$$\min_{W_1} \min_{W_2} \min_{\{\alpha_j\}_{j=1}^m} \| \sum_{j=1}^{m} X W_{1j} W_{2j}^T - Y \|_F^2 / 2 + \beta \sum_{j=1}^{m} \frac{\|W_{1j}\|_2^2}{\alpha_j^2} + \alpha_j^2 \|W_{2j}\|_2^2$$

$$= \min_{W_1} \min_{W_2} \frac{1}{2} \| \sum_{j=1}^{m} X W_{1j} W_{2j}^T - Y \|_F^2 + 2\beta \sum_{j=1}^{m} \|W_{1j}\|_2 \|W_{2j}\|_2$$

Principal Component Analysis and Autoencoder Networks

$$= \min_{W_1} \min_{W_2} \frac{1}{2} \| \sum_{j=1}^{m} X W_{1j} W_{2j}^T - Y \|_F^2 + 2\beta \sum_{j=1}^{m} \|W_{1j}\|_2 \|W_{2j}\|_2$$

- substitute $W_{1j} \leftarrow W_{1j} / \|W_{1j}\|_2$ and $W_{2j} \leftarrow W_{2j} / \|W_{1j}\|_2$

$$= \min_{\|W_{1j}\|_2 = 1, \forall j} \min_{W_2} \frac{1}{2} \| \sum_{j=1}^{m} X W_{1j} W_{2j}^T - Y \|_F^2 + 2\beta \sum_{j=1}^{m} \|W_{2j}\|_2$$

- group ℓ_1 penalty on the second layer weights
- convex problem when we fix W_1

Dual Neural Network Problem

- take the dual of the W_2 minimization problem while W_1 is fixed

$$\min_{\|W_{1j}\|_2=1,\forall j} \min_{W_2} \frac{1}{2}\|\sum_{j=1}^{m} XW_{1j}W_{2j}^T - Y\|_F^2 + 2\beta\sum_{j=1}^{m}\|W_{2j}\|_2$$

$$= \min_{\|W_{1j}\|_2=1,\forall j} \max_{V} -\frac{1}{2}\|V - Y\|_F^2 \text{ s.t. } \|V^TXW_{1j}\|_2 \leq \beta \ \forall \text{ j}$$

Dual Neural Network Problem

- strong duality holds
- we can exchange minimum and maximum[1]

$$\min_{\|W_{1j}\|_2=1, \forall j} \min_{W_2} \frac{1}{2} \| \sum_{j=1}^{m} X W_{1j} W_{2j}^T - Y \|_F^2 + 2\beta \sum_{j=1}^{m} \|W_{2j}\|_2$$

$$= \min_{\|W_{1j}\|_2=1, \forall j} \max_{V} -\frac{1}{2} \|V - Y\|_F^2 \text{ s.t. } \|V^T X W_{1j}\|_2 \leq \beta \ \forall \text{ j}$$

$$= \max_{V} -\frac{1}{2} \|V - Y\|_F^2 \text{ s.t. } \max_{\|W_{1j}\|_2=1, \forall j} \|V^T X W_{1j}\|_2 \leq \beta \ \forall \text{ j}$$

[1] See the paper *Neural Networks are Convex Regularizers* for a proof https://stanford.edu/∼pilanci/papers/NNConvex.pdf

Dual Neural Network Problem

- we obtain a convex dual problem

$$
\min_{\|W_{1j}\|_2=1,\forall j} \min_{W_2} \frac{1}{2}\|\sum_{j=1}^{m} XW_{1j}W_{2j}^T - Y\|_F^2 + 2\beta \sum_{j=1}^{m} \|W_{2j}\|_2
$$

$$
= \max_{V} -\frac{1}{2}\|V - Y\|_F^2 \text{ s.t. } \sigma_{\max}(V^T X) \leq \beta
$$

- simple constrained convex program

Bidual of the Neural Network Problem

- we obtain a convex dual problem

$$
\begin{aligned}
&\min_{\|W_{1j}\|_2=1,\forall j}\ \min_{W_2} \frac{1}{2}\|\sum_{j=1}^m XW_{1j}W_{2j}^T - Y\|_F^2 + 2\beta\sum_{j=1}^m \|W_{2j}\|_2 \\
&= \max_V -\frac{1}{2}\|V-Y\|_F^2 \text{ s.t. } \sigma_{\max}(V^TX) \le \beta \\
&= \min_W \|XW-Y\|_F^2 + \beta\|W\|_*
\end{aligned}
$$

where $\|W\|_* = \sum_{k=1}^m \sigma_k(W)$ is the Nuclear Norm

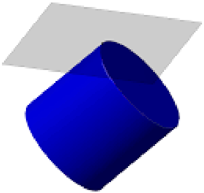
ℓ_1 norm of the singular values of W

- sparse singular values = low rank matrix
- linear activation + weight decay = Nuclear Norm penalty

Nuclear Norm

$$\|W\|_* = \sum_{k=1}^{m} \sigma_k(W) = \|\sigma(W)\|_1$$

- also called Schatten 1-Norm
- for positive semidefinite matrices nuclear norm reduces to the trace norm $\|W\|_* = \sum_{k=1}^{m} \lambda_k(W) = \textbf{Trace}\, W$
- extreme points of the nuclear norm ball $\{W : \|W\|_* \leq 1\}$ are rank-one matrices



- analogous to the extreme points of the ℓ_1 ball which are 1-sparse vectors, i.e., the Dirac delta basis

Robust Principal Component Analysis

- Suppose that the data matrix is the sum of a low rank matrix and a sparse matrix

$$X = \underbrace{W}_{\text{low rank}} + \underbrace{S}_{\text{sparse}}$$

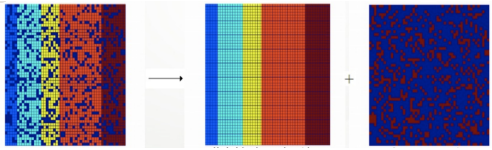
- convex heuristic:

$$\min_{W,S\,:X=S+W} \|W\|_* + \|S\|_1$$

- efficiently solvable

  examples:

  low rank data + outliers



Robust Principal Component Analysis

$$\min_{W,S\,:Y=S+W} \|W\|_* + \|S\|_1$$

examples:

background image + few corrupted pixels