Deep Learning Basics
Lecture 7: Factor Analysis

Princeton University COS 495
Instructor: Yingyu Liang
Supervised v.s. Unsupervised
Math formulation for supervised learning

• Given training data \{ (x_i, y_i) : 1 \leq i \leq n \} i.i.d. from distribution \( D \)
• Find \( y = f(x) \in H \) that minimizes \( \hat{L}(f) = \frac{1}{n} \sum_{i=1}^{n} l(f, x_i, y_i) \)
• s.t. the expected loss is small
  \[ L(f) = \mathbb{E}_{(x,y) \sim D}[l(f, x, y)] \]
Unsupervised learning

- Given training data \( \{x_i: 1 \leq i \leq n\} \) i.i.d. from distribution \( D \)
- Extract some “structure” from the data

- Do not have a general framework
- Typical unsupervised tasks:
  - Summarization: clustering, dimension reduction
  - Learning probabilistic models: latent variable model, density estimation
Principal Component Analysis (PCA)
High dimensional data

• Example 1: images

Dimension: $300 \times 300 = 90,000$
High dimensional data

• Example 2: documents

• Features:
  • Unigram (count of each word): thousands
  • Bigram (co-occurrence contextual information): millions

• Netflix survey: 480189 users x 17770 movies

<table>
<thead>
<tr>
<th></th>
<th>Movie 1</th>
<th>Movie 2</th>
<th>Movie 3</th>
<th>Movie 4</th>
<th>Movie 5</th>
<th>Movie 6</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>User 1</td>
<td>5</td>
<td>?</td>
<td>?</td>
<td>1</td>
<td>3</td>
<td>?</td>
<td></td>
</tr>
<tr>
<td>User 2</td>
<td>?</td>
<td>?</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>User 3</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>?</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Example from Nina Balcan
Principal Component Analysis (PCA)

• Data analysis point of view: dimension reduction technique on a given set of high dimensional data \( \{x_i: 1 \leq i \leq n\} \)

• Math point of view: eigen-decomposition of the covariance (or singular value decomposition of the data)

• Classic, commonly used tool
Principal Component Analysis (PCA)

• Extract hidden lower dimensional structure of the data
  • Try to capture the variance structure as much as possible

• Computation: solved by singular value decomposition (SVD)
Principal Component Analysis (PCA)

• **Definition:** an orthogonal projection or transformation of the data into a (typically lower dimensional) subspace so that the variance of the projected data is maximized.

Figure from isomorphismes @stackexchange
Principal Component Analysis (PCA)

- An illustration of the projection to 1 dim
- Pay attention to the variance of the projected points

Figure from amoeba@stackexchange
Principal Component Analysis (PCA)

- Principal Components (PC) are directions that capture most of the variance in the data.

- First PC: direction of greatest variability in data.
  - Data points are most spread out when projected on the first PC compared to any other direction.

- Second PC: next direction of greatest variability, orthogonal to first PC.

- Third PC: next direction of greatest variability, orthogonal to first and second PC’s.

- ...
Math formulation

• Suppose the data are centered: \( \sum_{i=1}^{n} x_i = 0 \)

• Then their projections on any direction \( v \) are centered: \( \sum_{i=1}^{n} v^T x_i = 0 \)

• First PC: maximize the variance of the projections

\[
\max_v \sum_{i=1}^{n} (v^T x_i)^2, \quad \text{s.t.} \quad v^T v = 1
\]

equivalent to

\[
\max_v v^T X X^T v, \quad \text{s.t.} \quad v^T v = 1
\]

where the columns of \( X \) are the data points
Math formulation

• First PC:

$$\max_v v^T X X^T v, \quad \text{s.t. } v^T v = 1$$

where the columns of $X$ are the data points

• Solved by Lagrangian: exists $\lambda$, so that

$$\max_v v^T X X^T v - \lambda v^T v$$

$$\frac{\partial}{\partial v} = 0 \rightarrow (X X^T - \lambda I) v = 0 \rightarrow X X^T v = \lambda v$$
Computation: Eigen-decomposition

• First PC: $XX^Tv = \lambda v$

• $XX^T$ : covariance matrix
• $v$ : eigen-vector of the covariance matrix
• First PC: first eigen-vector of the covariance matrix

• Top $k$ PC’s: similar argument shows they are the top $k$ eigen-vectors
Computation: Eigen-decomposition

• Top $k$ PC’s: the top $k$ eigen-vectors $XX^T U = \Lambda U$
  where $\Lambda$ is a diagonal matrix
• $U$ are the left singular vectors of $X$

• Recall SVD decomposition theorem:
• An $m \times n$ real matrix $M$ has factorization $M = U\Sigma V^T$ where $U$ is an $m \times m$ orthogonal matrix, $\Sigma$ is a $m \times n$ rectangular diagonal matrix with non-negative real numbers on the diagonal, and $V$ is an $n \times n$ orthogonal matrix.
Equivalent view: low rank approximation

• First PC maximizes variance:

\[ \max_v v^T XX^T v, \quad \text{s.t.} \quad v^T v = 1 \]

• Alternative viewpoint: find vector \( v \) such that the projection yields minimum MSE reconstruction

\[ \min_v \frac{1}{n} \sum_{i=1}^{n} ||x_i - vv^T x_i||^2, \quad \text{s.t.} \quad v^T v = 1 \]
Equivalent view: low rank approximation

- Alternative viewpoint: find vector $\mathbf{v}$ such that the projection yields minimum MSE reconstruction

$$\min_{\mathbf{v}} \frac{1}{n} \sum_{i=1}^{n} ||\mathbf{x}_i - \mathbf{v}\mathbf{v}^T\mathbf{x}_i||^2, \quad s.t. \quad \mathbf{v}^T\mathbf{v} = 1$$

$\text{blue}^2 + \text{green}^2 = \text{black}^2$

$\text{black}^2$ is fixed (it’s just the data)

So, maximizing $\text{blue}^2$ is equivalent to minimizing $\text{green}^2$

Figure from Nina Balcan
Summary

• PCA: orthogonal projection that maximizes variance
• Low rank approximation: orthogonal projection that minimizes error
• Eigen-decomposition/SVD

• All equivalent for centered data
Sparse coding
A latent variable view of PCA

• Let $h_i = v^T x_i$
• Data point viewed as $x_i = vh_i + noise$
A latent variable view of PCA

- Consider top $k$ PC’s $U$
- Let $h_i = U^T x_i$
- Data point viewed as $x_i = Uh_i + noise$
A latent variable view of PCA

- Consider top $k$ PC's $U$
- Let $h_i = U^T x_i$
- Data point viewed as $x_i = Uh_i + noise$

PCA structure assumption: $h$ low dimension. What about other assumptions?
Sparse coding

- Structure assumption: $h$ is sparse, i.e., $|h|_0$ is small
- Dimension of $h$ can be large
Sparse coding

• Latent variable probabilistic model view:

\[ p(x|h) = W h + N \left(0, \frac{1}{\beta} I\right), \text{ } h \text{ is sparse}, \]

• E.g., from Laplacian prior: \[ p(h) = \frac{\lambda}{2} \exp\left(-\frac{\lambda}{2} |h|_1\right) \]
Sparse coding

• Suppose $W$ is known. MLE on $h$ is

$$h^* = \arg \max_h \log p(h|x)$$

$$h^* = \arg \min_h \lambda ||h||_1 + \beta ||x - Wh||_2^2$$

• Suppose both $W, h$ unknown.
  • Typically alternate between updating $W, h$
Sparse coding

• Historical note: study on visual system

Project paper list
Supervised learning

- AlexNet: *ImageNet Classification with Deep Convolutional Neural Networks*
- GoogLeNet: *Going Deeper with Convolutions*
- Residue Network: *Deep Residual Learning for Image Recognition*
Unsupervised learning

• Deep belief networks: *A fast learning algorithm for deep belief nets*

• *Reducing the Dimensionality of Data with Neural Networks*

• Variational autoencoder: *Auto-Encoding Variational Bayes*

• *Generative Adversarial Nets*
Recurrent neural networks

• Long-short term memory

• Memory networks

• Sequence to Sequence Learning with Neural Networks
You choose the paper that interests you!

• Need to consult with TA
  • Heavier responsibility on the student side if customize the project

• Check recent papers in the conferences ICML, NIPS, ICLR
• Check papers by leading researchers: Hinton, Lecun, Bengio, etc
• Explore whether deep learning can be applied to your application

• Not recommend arXiv: too many deep learning papers