Assignment #5

Due: 23:55pm Friday 26 April 2019

Upload at: `https://dropbox.cs.princeton.edu/COS324_S2019/HW5`

Problem 1 (1pt)

Use the instructions above to put your name on the assignment when you compile this document.

Problem 2 (24pts)

Imagine that we have N data $\{\boldsymbol{x}_n\}_{n=1}^N$, where $\boldsymbol{x}_n \in \mathbb{R}^D$. We will model them as having a K-dimensional latent continuous representation with $K \ll D$. One way to do this is to imagine a procedure with the following steps:

1. Generate N latent data $\{z_n\}_{n=1}^N$ from a K-dimensional spherical Gaussian, i.e., $z_n \sim \mathcal{N}(0, \mathbb{I}_K)$.
2. Linearly transform the data with a (not necessarily orthonormal) matrix $\boldsymbol{W}^{D\times K}$ into $\boldsymbol{W}z_n$.
3. Add a bit of independent Gaussian noise to this quantity to get $\boldsymbol{x}_n = \boldsymbol{W}z_n + \boldsymbol{\eta}_n$ where $\boldsymbol{\eta}_n \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Lambda})$ and $\boldsymbol{\Lambda}$ is a diagonal positive definite matrix.

A. Write down the marginal distribution of $\boldsymbol{x}_n$.
B. Conditioned on $\boldsymbol{\mu}$, $\boldsymbol{W}$ $\boldsymbol{\Lambda}$, and $\boldsymbol{x}_n$, what is the distribution over z_n?
C. Write out the "complete data log likelihood" for the nth datum — the log of the joint probability of *both* $\boldsymbol{x}_n$ and z_n, conditioned on $\boldsymbol{\mu}$, $\boldsymbol{W}$, and $\boldsymbol{\Lambda}$.
D. Denote the mean and covariance from part (B) as $\boldsymbol{m}_n$ and $\boldsymbol{S}_n$, respectively. Compute the expectation of part (C) under this mean and covariance. This quantity is sometimes called the "expected complete data log likelihood". The *expectation maximization* (EM) algorithm consists of alternating between maximizing this quantity summed over all the data, with respect to $\boldsymbol{W}$ and $\boldsymbol{\Lambda}$, and updating $\boldsymbol{m}_n$ and $\boldsymbol{S}_n$ via the computation from part (B), across all of the data. The EM algorithm is one way to find maximum likelihood estimate in models with latent variables like z_n.

Problem 3 (20pts)

Imagine that you have a matrix $\boldsymbol{X} \in \mathbb{R}^{N\times D}$ that you'd like to model as the (approximate) product of a tall-and-thin matrix $\boldsymbol{U} \in \mathbb{R}^{N\times K}$ and a short-and-fat matrix $\boldsymbol{V}^{\mathrm{T}}$ where $\boldsymbol{V} \in \mathbb{R}^{D\times K}$. Here $K \ll N, D$. We want to find a $\boldsymbol{U}$ and $\boldsymbol{V}$ so that $\boldsymbol{U}\boldsymbol{V}^{\mathrm{T}}$ is close to $\boldsymbol{X}$. There are various ways to measure when two matrices are "close" but we'll choose the squared *Frobenius norm* in this case, which is essentially the sum of the squares across all entries:

$$||\boldsymbol{A}||_{\mathrm{Fro}}^2 = \sum_{i,j} A_{i,j}^2 = \mathrm{trace}(A^{\mathrm{T}}A)\,. \tag{1}$$

Thus we could write our loss function for learning $\boldsymbol{U}$ and $\boldsymbol{V}$ as

$$L(\boldsymbol{U}, \boldsymbol{V}) = \frac{1}{2}||\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^{\mathrm{T}}||_{\mathrm{Fro}}^2\,, \tag{2}$$

and minimize this to find good $\boldsymbol{U}$ and $\boldsymbol{V}$.
A. Would the solution of this minimization (the pair of matrices $\boldsymbol{U}$ and $\boldsymbol{V}$) be unique? Why or why not?
B. One approach to minimizing this loss function is to use coordinate descent and alternate between fixing $\boldsymbol{V}$ and minimizing with respect to $\boldsymbol{U}$, and fixing $\boldsymbol{U}$ and minimizing with respect to $\boldsymbol{V}$. The alternating updates in this case have a nice and familiar form, with the caveat that you'll need to think about the "gradient" with respect to a matrix. Just do the obvious thing and find matrices of the same size as $\boldsymbol{U}$ and $\boldsymbol{V}$ containing the partial derivatives for each matrix element. Then set these to zero and solve for the iterative updates.

Problem 4 (20pts)

Take a greyscale image of your choice and load it into Python as a matrix where the entries in the matrix are the pixel intensities. Compute the SVD of the image matrix.
A. Square the singular values, sort them from largest to smallest and plot their cumulative sum as a function of latent dimension K. Is there interesting structure in the curve, e.g., an elbow?
B. Choose 8 different low-rank truncations and create images of each of the reconstructions. How does the visual quality of the reconstruction relate to the plot from part A?

Problem 5 (35pts)

Go out and grab an image data set like:

- CIFAR-10 or CIFAR-100:
  `http://www.cs.toronto.edu/~kriz/cifar.html`
- MNIST Handwritten Digits:
  `http://yann.lecun.com/exdb/mnist/`
- Small NORB (toys):
  `http://www.cs.nyu.edu/~ylclab/data/norb-v1.0-small/`
- Street View Housing Numbers:
  `http://ufldl.stanford.edu/housenumbers/`
- STL-10:
  `http://cs.stanford.edu/~acoates/stl10/`
- Labeled Faces in the Wild:
  `http://vis-www.cs.umass.edu/lfw/`

Figure out how to load it into your environment and turn it into a set of vectors. Center the data (make it zero mean) and compute the covariance matrix. Compute the eigenvalue decomposition of the covariance using `numpy.linalg` or `scipy.linalg`.
A. Produce a Scree plot, similar to the problem above, showing the cumulative sum of eigenvalues (when they are ordered from largest to smallest). Is there interesting structure in this plot indicating that the data are low dimensional?
B. Take the top 16 eigenvectors and put them back into image space, probably by rescaling them to be in $[0, 1]$, reshaping, and then using `imshow`. Produce a figure with these images as subplots.
C. Take 1000 of the data or so and project them onto the top six eigenvectors. Produce three visualizations, each showing a two-dimensional representation arising from PC1 vs. PC2, PC3 vs. PC4, and PC5 vs. PC6. (These are essentially what I showed in lecture.) To make the visualizations, I suggest looking into the `matplotlib figimage` function which lets you place an image anywhere you want in a figure.

Changelog

- 15 April 2019 – Initial version.