Lecture 11. Kernel PCA and Multidimensional Scaling (MDS)

Prof. Alan Yuille

Spring 2014

Outline

1. Kernel PCA
2. Multidimensional Scaling (MDS)

1 Kernel PCA

The kernel trick can be applied to PCA. The basic idea is that we go to feature space from original data space $x \rightarrow \phi(x)$. In this lecture, we assume that $\frac{1}{N}\sum_{k=1}^{N}\phi(x_k) = 0$. (As in PCA lecture, we assume $\frac{1}{N}\sum_{k=1}^{N} x_k = 0$). This can be easily done by subtraction.

Remember that the output of PCA is the projection of the data $\{x_i : i = 1, ..., N\}$ onto the subspace defined by $(e_1, ..., e_M)$. This depends only on dot product, which suggests that we can use the kernel trick if we replace x by $\phi(x)$. But there are difficulties in computing the eigenvalues of the new covariance matrix $C = \frac{1}{N}\sum_{k=1}^{N}\phi(x_k)\phi^T(x_k)$. The feature space is in most cases extremely high-dimensional, so it is almost impossible in practice to compute the feature vectors, and furthermore, the covariance matrix. How do we compute the eigenvalues and eigenvectors then?

For the ordinary PCA, we have the following claim:

The eigenvectors with non-zero eigenvalues can be expressed in form $e = \sum_{j=1}^{N}\alpha_j x_j$. The eigenvectors with zero eigenvalues are of form e, s.t. $x_j \cdot e = 0, j = 1, ..., N$.

Proof:

Suppose $x_j \cdot e = 0, j = 1, ..., N$. Then

$$C \cdot e = \frac{1}{N}\sum_{j=1}^{N} x_j < x_j \cdot e > = 0 \cdot e = 0.$$

Hence e is an eigenvector with zero eigenvalues. This proves the last sentence. The remaining eigenvectors must be orthogonal to the zero eigenvectors. Hence they are of form

$$e = \sum_{j=1}^{N} \alpha_j x_j.$$

That concludes the proof.

Now replace x by $\phi(x)$, we get

$$C = \frac{1}{N} \sum_{k=1}^{N} \phi(x_k)\phi^T(x_k).$$

Based on the above claim, the non-zero eigenvectors e are of the form

$$e = \sum_{j=1}^{N} \alpha_j \phi(x_j).$$

Substituting

$$C \cdot e = \lambda e$$

Then we get

$$\frac{1}{N} \sum_{k=1}^{N} \phi(x_k) \sum_{j=1}^{N} \alpha_j < \phi(x_k), \phi(x_j) >= \lambda \sum_{j=1}^{N} \alpha_j \phi(x_j).$$

Equivalently, we can switch the two sumation index on the left-hand size

$$\frac{1}{N} \sum_{j=1}^{N} \phi(x_j) \sum_{k=1}^{N} \alpha_k < \phi(x_j), \phi(x_k) >= \lambda \sum_{j=1}^{N} \alpha_j \phi(x_j).$$

Equating coefficients of $\phi(x_j)$

$$\frac{1}{N} \sum_{k=1}^{N} < \phi(x_k), \phi(x_j) > \alpha_k = \lambda \alpha_j.$$

Then we can see that, for each j

$$\frac{1}{N} \sum_{k=1}^{N} K(x_k, x_j)\alpha_k = \lambda \alpha_j.$$

So α and λ are eigenvector and eigenvalue of matrix $\frac{1}{N}K$, where K is the kernel function. Let μ denote the $\mu - th$ eigenvalues, we have

$$\frac{1}{N} K \alpha^\mu = \lambda^\mu \alpha^\mu$$

After computing α^μ and λ^μ, we can easily derive e^μ

$$e^\mu = \sum_{j=1}^{N} \alpha_j^\mu \phi(x_j).$$

Given a new datapoint x, the projection is

$$e^\mu \cdot \phi(x) = \sum_{j=1}^{N} \alpha_j^\mu K(x_j, x).$$

The above argument tells us that in order to compute eigenvector e of the covariance matrix C, we can compute the eigenvector α of the kernel matrix K. And also the two matrices have the same rank and the same non-zero eigenvalues.

Hence, the projection of the data onto the eigenvector requires only knowing the kernel $K(x_i, x_j)$ and the α^μ. We don't need to compute $\phi(x)$. The knowledge of the kernel is used twice:

(1). when computing λ^μ and α^μ.

(2). when computing the projection of new datapoints.

2 Multidimensional scaling (MDS)

2.1 Idea of MDS

MDS is a linear projection method similar to PCA. But it has the attractive property that it does not require knowing the features $\vec{x}$ of the datapoints. Instead it only needs to know the distances between two datapoints. This is attractive for problems where it is hard to decide what features to use – e.g., for representing a picture– but easier to decide if two pictures are similar. This also makes it suitable for nonlinear dimension reduction because MDS depends on the distance on the manifold.

The basic idea of MDS is to project the data to preserve the distances $|\vec{x}_i - \vec{x}_j|$ between the datapoints. In other words, we project $\vec{x}$ to a lower-dimensional vector $\vec{y}$ so that $|\vec{y}_i - \vec{y}_j| \approx |\vec{x}_i - \vec{x}_j|$.

This projection constraint is imposed on the dot products $\vec{x}_i \cdot \vec{x}_j \approx \vec{y}_i \cdot \vec{y}_j$, which will imply the result.

Key Result: we only need to know $\Delta_{ij} = |\vec{x}_i - \vec{x}_j|$ in order to calculate $\vec{x}_i \cdot \vec{x}_j$ (i.e. we only need to know the distances between points and not their positions $\vec{x}$).

Result.

$$\vec{x}_i \cdot \vec{x}_j = \frac{1}{2N} \sum_k \Delta_{ik}^2 + \frac{1}{2N} \sum_l \Delta_{lj}^2 - \frac{1}{2N^2} \sum_{lk} \Delta_{lk}^2 - \frac{1}{2} \Delta_{ij}^2,$$

provided $\sum_i \vec{x}_i = 0$ (which we can always enforce).

Proof.
$\Delta_{ij}^2 = |\vec{x}_i|^2 + |\vec{x}_j|^2 - 2\vec{x}_i \cdot \vec{x}_j$. Let $T = \sum_i |\vec{x}_i|^2$. (Note that $\sum_i \vec{x}_i \cdot \vec{x}_j = 0 = \sum_j \vec{x}_i \cdot \vec{x}_j$). Then

$$\sum_k \Delta_{ik}^2 = N|\vec{x}_i|^2 + T, \sum_l \Delta_{lj}^2 N|\vec{x}_j|^2 + T, \sum_{lk} \Delta_{lk}^2 = 2NT.$$

The result follows by subtraction.

Now we proceed by defining the Gram matrix:

$$G_{ij} = \vec{x}_i \cdot \vec{x}_j = \sum_k \Delta_{ik}^2 + \frac{1}{2N} \sum_l \Delta_{lj}^2 - \frac{1}{2N^2} \sum_{lk} \Delta_{lk}^2 - \frac{1}{2}\Delta_{ij}^2.$$

We minimize the error function

$$Err(y) = \sum_{ij} (G_{ij} - \vec{y}_i \cdot \vec{y}_j)^2.$$

Here $\vec{x}$ lies in D-dimensional space, G is an $N \times N$ matrix (N is the number of datapoints), $\vec{y}$ lies in a d-dimensional space, with $d << D$ and $d < N$.

2.2 MDS minimization

To minimize $Err(y)$ we do spectral decomposition of the Gram matrix:

$$G = \sum_a \lambda_a \vec{v}^a \vec{v}^a$$

In coordinates: $G_{ij} = \sum_a \lambda_a v_i^a v_j^a$, where λ_a is the a^{th} eigenvalue of G, $\vec{v}^a$ its eigenvector, and v_i^a is the i^{th} component of the eigenvector $\vec{v}^a$. We use the convention that $\lambda_1 \geq ... \geq \lambda_N$ and, as always, the eigenvectors are orthogonal $\vec{v}^a \cdot \vec{v}^b = 0$ if $a \neq b$ ($|\vec{v}^a| = 1$).

Result. The optimal minimization of $Err(y)$ is obtained by setting the vectors $\vec{y}_i$ to be have N components $y_i^a = \sqrt{\lambda_a} v_i^a$. I.e. $\vec{y}_i = (\sqrt{\lambda_1} v_i^1, ..., \sqrt{\lambda_N} v_i^N)$.

Proof. $\vec{y}_i \cdot \vec{y}_j = \sum_a \sqrt{\lambda_a} v_i^a \sqrt{\lambda_a} v_j^a = \sum_a \lambda_a \vec{v}_i^a \vec{v}_j^a = G_{ij}$. I.e. we get $Err(y) = 0$ if we express $y_i^a = \sqrt{\lambda_a} v_i^a$.

But this does not reduce the dimension (even if it gives zero error). We can reduce the dimension, while raising the error, by only keeping the first d components. I.e. set $\vec{y}_i = (\sqrt{\lambda_1} v_i^1, ..., \sqrt{\lambda_d} v_i^d)$ (recall that we ordered the eigenvalues $\lambda_1 \geq ... \geq \lambda_N$, so we are picking the biggest d eigenvalues).

In this case, $G_{ij} \neq \vec{y}_i \cdot \vec{y}_j^T$. Instead, we compute that

$$G_{ij} - \vec{y}_i \cdot \vec{y}_j = \sum_{a=d+1}^{N} \lambda_a v_i^a v_j^a$$

It follows that the error $Err(y) = \sum_{a=d+1}^{N} \lambda_a^2$. This justifies keeping the first d components (corresponding to the biggest d eigenvalues).

Hence we project to d-dimension provided that $\sum_{a=d+1}^{N} \lambda_a^2$ is small.

2.3 Relations between PCA and MDS

Both are linear projections. Both involve projections specified by the biggest d eigenvalues/eigenvectors of a matrix which depends on the data.

PCA uses the eigenvalues/eigenvectors of the covariance $\sum_{i=1}^{N} \vec{x}_i \vec{x}_i^T$. MDS uses eigenvalues/eigenvectors of the Gram matrix $\sum \vec{x}_i \cdot \vec{x}_j$.

Result. The covariance and the Gram matrix have the same eigenvalues and their eigenvectors are related.

Proof. This result was discussed in an earlier lecture (in the PCA lecture in the section which discussed SVD). To obtain it, we defined the matrix X with components x_{ia}. Then the covariance is expressed as the matrix XX^T and the Gram matrix as X^TX. If e is an eigenvector of XX^T with eigenvalue λ – i.e. $XX^Te = \lambda e$ – then X^Te is an eigenvector of X^TX with eigenvalue λ – because $X^TXX^Te = X^T(XX^Te) = X^T\lambda e = \lambda X^Te$. Similarly we can relate the eigenvectors/eigenvalues of X^TX to those of XX^T.

In summary, the error criteria of PCA and MDS depend on the same eigenvalues and dimension reduction occurs by rejecting the coordinate directions corresponding to the smallest eigenvalues. But the difference is the PCA and MDS project using the eigenvectors of different, but related, matrix (correlation or Gram).

The truncation conditino for PCA and MDS are similar

$$\frac{\sum_{i=1}^{d} \lambda_i^2}{\sum_{i=1}^{N} \lambda_i^2} > threshold.$$

Note that to do PCA, we have to know the data $\vec{x}_i$, but to do MDS, we only need to know Δ_{ij}. In some applications, it is possible to specify Δ_{ij} but not $\vec{x}_i$.