Computer Vision
Jacobs University Bremen
Fanlin Wang & Fanghang Ji

Course: CO22-320671
Nov, 8, 2017

Homework 2

Problem 2.1

The *predict* function is implemented as follows in opencv:

```
void Fisherfaces::predict(InputArray _src, Ptr<PredictCollector> collector) const {
Mat src = _src.getMat();
// check data alignment just for clearer exception messages
if(_projections.empty()) {
// throw error if no data (or simply return -1?)
String error_message = "This Fisherfaces model is not computed yet. Did you call Fish
CV_Error(Error::StsBadArg, error_message);
} else if(src.total() != (size_t) _eigenvectors.rows) {
String error_message = format("Wrong input image size. Reason: Training and Test imag
CV_Error(Error::StsBadArg, error_message);
}
// project into LDA subspace
Mat q = LDA::subspaceProject(_eigenvectors, _mean, src.reshape(1,1));
// find 1-nearest neighbor
collector->init((int)_projections.size());
for (size_t sampleIdx = 0; sampleIdx < _projections.size(); sampleIdx++) {
        double dist = norm(_projections[sampleIdx], q, NORM_L2);
        int label = _labels.at<int>((int)sampleIdx);
        if (!collector->collect(label, dist))return;
}
}
```

It has two parameters:

1. src: Sample image to get a prediction from.
2. collector: User-defined collector object that accepts all results

subpaceProject function projects all training image(stored in eigenvectors) to the lower dimensional data space. Then for i from 0 to c, where c is the size of the row, it computes *dist* as follows:

```
double dist = norm(_projections[sampleIdx], q, NORM_L2);
```

In opencv, a PCA is applied in front of the LDA in order to reduce the feature vectors to about the size of the image-count. Therefore, the *projections* matrix stores the projections of the original data as PCA.eigenvectors * LDA.eigenvectors:

```
gemm(pca.eigenvectors, lda.eigenvectors(), 1.0, Mat(), 0.0, _eigenvectors, GEMM_1_T);
for(int sampleIdx = 0; sampleIdx < data.rows; sampleIdx++) {
        Mat p = LDA::subspaceProject(_eigenvectors, _mean, data.row(sampleIdx));
        _projections.push_back(p);
}
```

q stores the projections of LDA subspace. The heuristic function used to determine the class of the image is therefore the shortest distance obtained by comparing these two projections.

Now written this in equations:
Let A be the matrix composed of normalized eigenvectors of

$$\Sigma_W^{-1}\Sigma_B y = \lambda y$$

where Σ_W is the single value decomposition of scatter matrix within class, and Σ_B is that of scatter matrix between class. c is the number of classes, and we have $c-1$ eigenvectors for this equation. Therefore,

$$A = \begin{bmatrix} \hat{\alpha_1} & \dots & \hat{\alpha_{c-1}} \end{bmatrix}$$

let μ be an mn by 1 matrix extracted from the grey image:

$$\tilde{\mu} = \mu - \bar{mu} = U_{PCA} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_{l-c} \end{bmatrix}$$

project this to an $l - c$ dim space we get:

$$U_{PCA}^T \tilde{\mu}$$

We further project it into a lower $c - 1$ dim space:

$$\beta_{ij} = A^T U_{PCA}^T (\mu_{ij} - \bar{\mu})$$

$$\beta_t = A^T U_{PCA}^T (t - \bar{\mu})$$

β_t is the projection of an PCA $\cdot$ LDA subspace

$$\gamma_t = A^T (t - \bar{\mu})$$

γt is the projection of LDA subspace. For testing image we predict class as

$$\underset{t=1,\ldots,c}{argmin} ||\gamma_t - \beta_t||^2$$