CS7461 Machine Learning: Assignment 3

Matthias Grundmann
grundman@cs.tum.edu

1 Introduction

In this assignment I will examine how well different cluster and dimension reduction algorithms compare and how much impact the application of those as a preprocessing step has to the accuracy of classification with a neural network. I will reuse my datasets from assignment one, that I will briefly review.

1.1 Used datasets

I used two datasets for evaluation of the two cluster and four dimension reduction algorithms, one set of images of hand written digits and one set of cell nucleus properties to enhance breast tumor diagnosis.

1.1.1 Hand written digits

Recognizing hand written digits or characters enables the development of user friendly input devices. An especially need for this arises by the wide spread and use of mobile devices nowadays. Although the successful integration of small keyboards in these devices reduced the need of recognition, it limits their size, a problem that cannot be solved without focusing on other ways of user input.
The variation of human written digits or characters makes the reliable recognition a very hard task. Regional variations of the interpretation of digits make it nearly impossible to build reference digits, e.g. based on a line representation. For example the digit '7' is written in Northern America very similar to a European '1'. I used a subset of the MNIST database [1], which is itself a subset of the NIST database. The original MNIST database consists of 60,000 labeled training and 10,000 labeled test images. I reduced the database by factor 20 to the first 3,000 training and first 500 test images to keep computation time manageable for my machine.[1] Each image is 28x28 pixels big and contains 256 intensity levels, for an example see figure 1. The digit is centered with its center of mass in the middle of each image. The MNIST database mixes the data of Special Database 1 (SD-1) and Special Database 3 (SD-3), which where collected among Census Bureau employees and high school students respectively. The digits in the SD-3 are much cleaner that these found in SD-1 as indicated by [6].

Each image is converted into a 784 dimensional feature vector by simply concatenating the rows of each images. Because the digits are formed by dashes on a white background, the feature vectors are sparse. The vectors are normalized to zero mean with standard deviation one

1.1.2 Breast tumor diagnosis

The most important non-invasive techniques to detect breast cancer are physician examination and mammography. While the former depends highly on the abilities of the physician, the latter is known to have a relatively high error rate (false-positive rate about 7 %, i.e. cancer detected in the absence of cancer and false-negative rate of at leat 10 percent). To detect the cancer reliable often a full biopsy is applied afterwards, which is a full invasive surgery. Therefore a reliable non-invasive technique is preferred, e.g. by analyzing a small amount of tissue from the tumor. By extracting cell

[1] Processing the whole dataset caused always out-of-memory exceptions in MATLAB

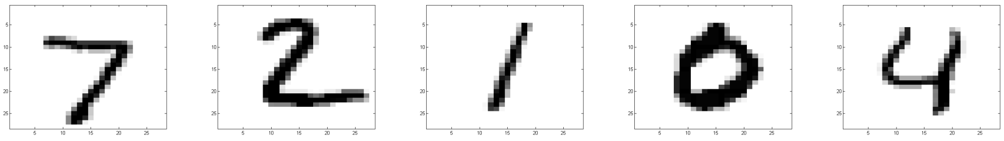

Figure 1: Example images of hand written digits (7,2,1,0,4)

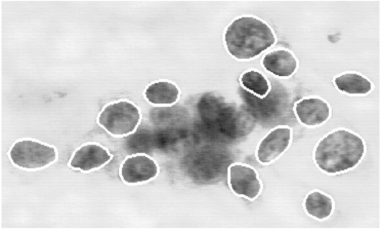

Figure 2: Snakes fitted around cell nucleus

properties from images of these tissues a recognition rate of 97% [8] could be achieved.
I am using the already labeled extracted data, obtained from the UCI Machine Learning Repository [3], more specific the 'New prognostic database', consisting of 569 samples. Each sample is described by the mean, the standard deviation and the mean of the three largest values of 10 attributes, leading to a total number of 30 attributes $\in \mathbb{R}$. The attributes were obtained by minimizing the energy functions of deformable splines, called snakes, to calculate the boundary of a cell nucleus. Details are given in [8] and an example picture is shown in figure 2.

According to [8] the 10 extracted attributes are:

Radius obtained by averaging the length of radial line segments from the centroid to the individual snake points

Perimeter Sum over the total distance of the snake points

Area Number of pixels in the interior of the snake and adding $\frac{1}{2}$ of the perimeter pixels

Compactness $\frac{\text{perimeter}^2}{\text{area}}$

Smoothness Difference between length of a radial line and the mean length of the lines surrounding it

Concavity Draw chords between non-adjacent snake points and measure distance to object boundary

Concave Points Counts the number of contour concavities

Symmetry Similar to relation between major and minor axis

Fractal Dimension see explanation and figure 6 in [**?**]

Texture Variance of the intensity levels in the interior of the snake

I partitioned the dataset according to the rule of thumb given in [7] in $\frac{2}{3}$ training data (the first 380 samples) and $\frac{1}{3}$ test data (the remaining 189 samples).

1.1.3 Evaluation method

The beauty of the hand written digit dataset lies in the fact that it is possible to actually show images of cluster centers and base-vectors of the subspaces we are projecting on. It is also possible to show pictures of the dimension reduced data, in terms of a linear combination of the base-vectors of the subspaces we are projecting on. That makes it easier to understand what the results we obtain actually mean, and whether a reconstruction of the data in the lower dimensional spaces is possible. On the hand for the cancer dataset we have interpret the numerical values and can not display the

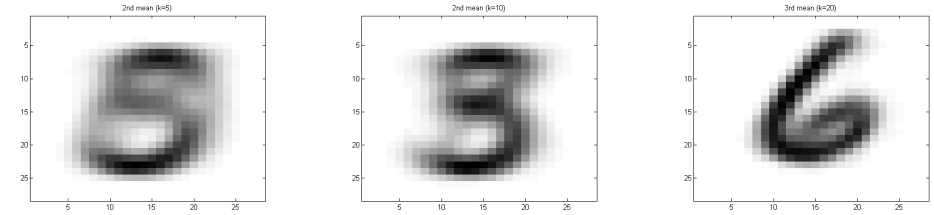

Figure 3: Example of cluster centers for EM-algorithm for k=5,10 and 20

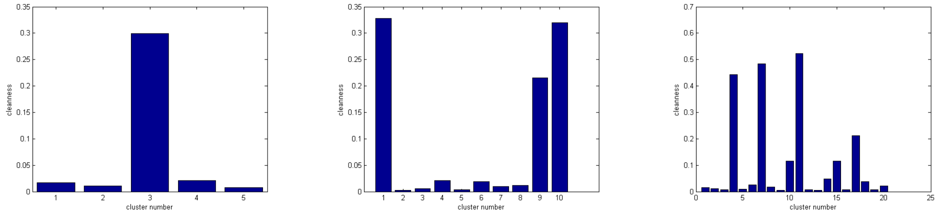

Figure 4: Cleanness of cluster centers for EM-algorithm for k=5,10 and 20

data in similar way. That why I focus for the digit dataset on the visual evaluation and for the cancer set on the numerical evaluation, e.g. in terms of reconstruction error.

2 Problem setup and Results

2.1 Clustering algorithms

Cause both algorithms, k-means as well as the EM-algorithm, select randomly the first k cluster centers, I run each algorithm three times and took the best result. I defined 'best' in the case of the EM-algorithm as the highest value of the maximized expected log likelihood that was obtained among these runs. In the case of the k-means algorithm, I computed for each cluster center the mean distance from each point belonging to this cluster to the cluster center. The variance of these mean distances is a measure of how equally sized the clusters are and is zero in the case of clusters with equal size. That is why I took the cluster result with the minimal variance. In both cases the distance is defined by the standard Euclidean metric. In the case of the digit dataset, I used the obvious value $k = 10$ and one with lower dimension ($k = 5$) and higher dimension($k = 20$). In the case of the cancer dataset, I used the obvious value ($k = 2$) and omitted higher values to keep the analysis short.

The EM-algorithm is called with random mean, but known standard deviation of value one. The standard deviation is known due to the normalization of the data to zero mean and standard deviation one. Figure 3 shows sample cluster centers obtained by the EM-algorithm for the digit data-set for different choices of k.

The samples show, that a choice of $k = 5$ leads to blurred cluster centers due to the fact, that more than one digit is clustered to the same center. Choices of $k = 10$ and $k = 20$ seem to lead to clusters, that capture real digits. But does these centers really model the original data? Figure 4 shows how 'clean' the clusters are for different choices of k, i.e. the percentage of the most frequent label w.r.t. the cluster size. Optimally we would expect a 'cleanness' of 1 for all cluster centers, i.e. the clusters represent the digits of the original data. The results are poor: only 1 or 3 cluster centers represent a actual digit of the dataset with a cleanness ranging from 30-40 %. The increase of k from 10 to 20 does not change this result significantly. The results seem to support the claim in the introduction, that is hard to build a reference digit due to the high variation in hand-written digits.

Figure 5 shows sample cluster centers obtained by the k-means algorithm for the digit data-set for different choices of k. The results are similar to the ones obtained by the EM-algorithm. Surprisingly

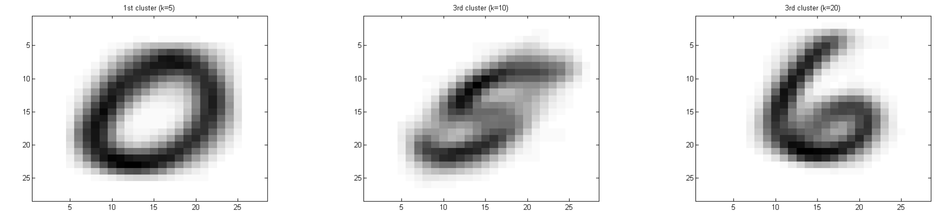

Figure 5: Example of cluster centers for k-means for k=5,10 and 20

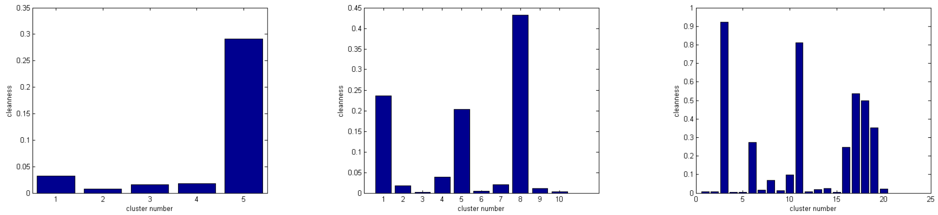

Figure 6: Cleanness of cluster centers for k-means algorithm for k=5,10 and 20

k-means seem to find more clusters that represent digits than EM does for larger choices of k, as figure 6 indicates.

In order to analyze how well the cluster centers are distributed among the data it is valuable to look at the distribution of the mean distances of the points to the their closest center shown in the left figure 7 for the k-means algorithm. The clusters have the similar mean radius, i.e. they cover the original data similarly and stable. The advantage of the EM-algorithm over the k-mean algorithm is, that is does not assign 'hard' memberships of the data to the clusters but uses 'soft' memberships representing the probability of being a member of a particular cluster. That arises the question whether the linear combination of the cluster centers with coefficients given by the 'soft' assignments can reconstruct the original data. An example is given in figure 7. The original digit '9', shown in the middle of the figure can be reconstructed pretty well, it is blurred a bit because it is the mixture of Gaussians.

Clustering the cancer dataset with both algorithms lead to clusters described in table 1 for four selected attributes. The results of k-means and the EM algorithm are the same, leading to two distinct clusters.

The clusters model the original data very well. 99.2% of the first cluster are malignant cancer cells, while only 18.7% of the second set are malignant cancer cells. Figure 8 shows boxplots for the attributes 'radius' and 'symmetry' for the cancer dataset for the original data. It can be seen, that the mean values of the attribute correspond very well to the coordinates of the two calculated clusters. We can conclude that malignant cancer-cells are in general larger and have more symmetry, and that a small subset of the attributes already models the original data very well. The right of figure 8 also shows the reconstruction error (sum of squared distances) obtained by the EM-algorithm. It

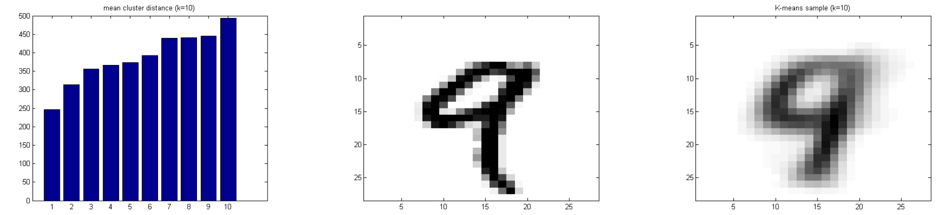

Figure 7: Left: Mean distance for k-mean, Right: Original digit '9' and reconstructed one by EM

Algorithm	Cluster	Radius	Compactness	Concavity	Symmetry
k-means	1	12.56	496.06	0.091	0.033
EM	1	12.56	496.06	0.091	0.033
k-means	2	19.38	1185.9	0.148	0.1
EM	2	19.38	1185.9	0.148	0.1

Table 1: Cluster results for cancer dataset

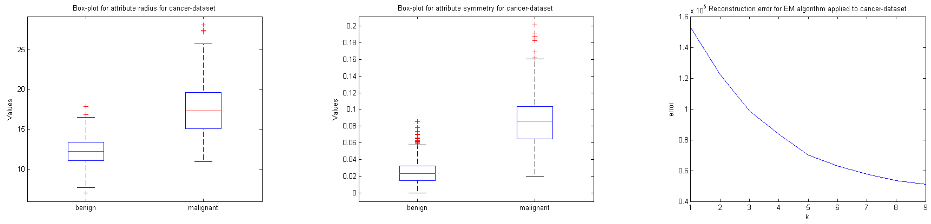

Figure 8: Left: Boxplots of the attributes radius and symmetry. Right: Reconstruction error for EM-algorithm

decreases exponentially with increasing k, but does not necessarily models the original data better. It is more likely that outliers as seen in the boxplots can be better represented. Furthermore the reconstruction error alone does not seem to be a valuable analysis tool. The reconstruction error is large (in the magnitude of 10^5) also the analysis of the clusters showed, that they result models the data reasonable well. Solely relying on the error does not give insights which k should be chosen.

2.2 Dimension reduction algorithms

To compare the results of Principal Component Analysis (PCA), Independent Component Analysis ICA and Randomized Projections (RP), I applied first PCA to both datasets and projected the data on a subspace that captures $v \in \{100, 99.9, 99, 97.5, 95, 90, 75, 33\}$ percent of the total variance of the data. The resulting number of corresponding principal components were used to set the dimension for ICA and RP. As usual the data was normalized to zero mean and standard deviation one.

2.2.1 PCA and ICA for Digit Dataset

Figure 9 shows the distribution of the sorted eigenvalues of the covariance matrix for digit dataset. The dimensions that capture $\lambda\%$ of the variance are also highlighted. They correspond to the dimensions $\{784, 472, 329, 228, 154, 91, 37, 7\}$. The figure shows, that dimension that are necessary to capture λ % of the data decreases very fast in accord with the decrease of the eigenvalues.[2] I assume this is because the digits fill only a small subset of the entire 28×28 area, in average the fraction of non-white pixels is 19.1% which corresponds to 150 pixel positions that carry information. The figure shows that the first 154 principal components already carry 95% of the data total variation, so the assumption is justified. Figure 10 shows three examples of the the digit '9' projected on the first principal components that carry 33, 95 and 99.9 % of the data. It can be seen, that 33% are probably to few principal components, while 95 % already give satisfactory results. The gain of using 3 times as many vectors to achieve the results for 99.9 % seems marginal. The visual result confirms our assumption above. It is worth to take a look at the base vectors that are generated by the PCA, also known as eigen-images. Figure 11 shows the 1st, 10th and 500th principal component. The first one seems to model large areas, like the dashes that occur in the digit 1, 7, 8 or 9. The 10th principal component models details around the large area, while the 500th looks like random noise at its border and is mostly zero (zero corresponds to the gray color) especially where the centroid of the digits are. The fact that it is mostly zero is a very good reason to discard it, and shows that the variation of the data is limited and the application of PCA is justified.

[2] Cause the covariance matrix is symmetric and positive definite the eigenvalues are all positive

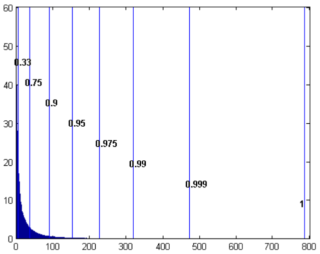

Figure 9: Distribution of eigenvectors and number of principal components for specific variances

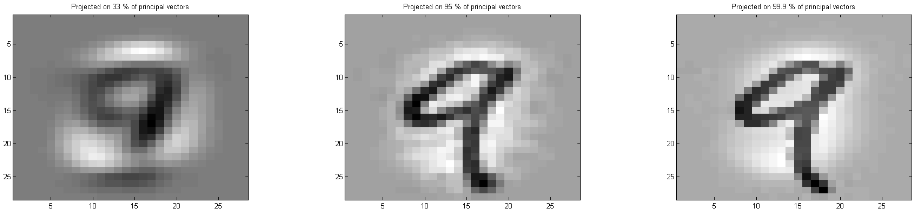

Figure 10: Examples of the digit '9' projected on 33, 95 and 99.9 % of the principal components

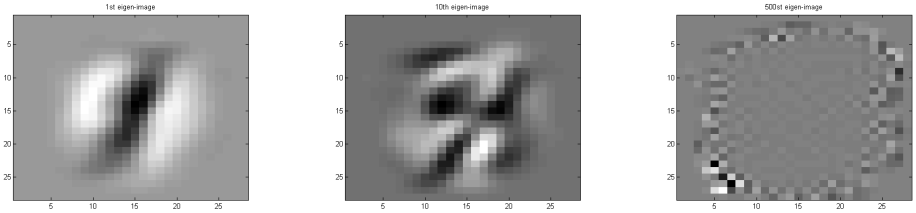

Figure 11: Examples of the 1st, 10th and 500th principal component

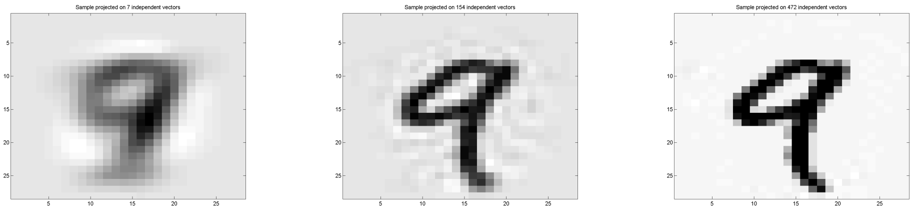

Figure 12: Examples of the digit '9' projected on 7, 154 and 472 independent signals

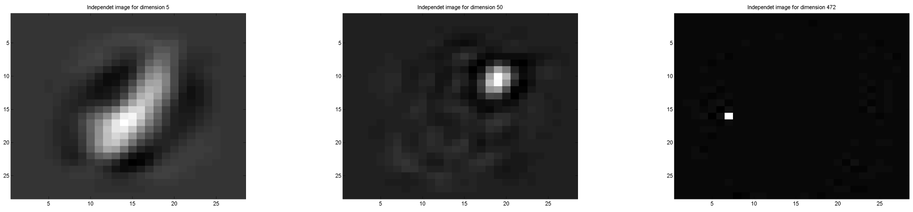

Figure 13: Examples of the independent signals for 5, 50 and 472 subspace dimensions

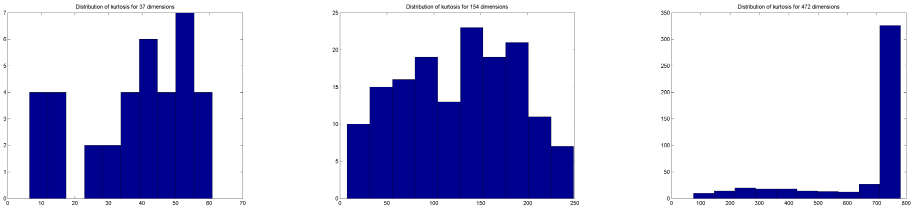

Figure 14: Histogram of kurtosis of obtained signals by ICA for 37, 154 and 472 subspace dimensions

Figure 12 shows the reconstruction of the digit '9' for 7, 154 and 472 dimensions, i.e. the same dimensions we used for samples for the PCA above. There does not exist a significant difference between both dimension reduction algorithms, ICA is able to model the surrounding of the digits better (which in white and in case of PCA gray, which corresponds to zero). A difference to PCA are the obtained base vectors that are shown in figure 13. Once the number of mixing signals is selected ICA computes those signals. There are no 'first' signals like in PCA that carry more variance than others, instead all signals are equally important. While in low dimensional space the mixing signals look like blurry blob-like structures very similar to PCA, with higher dimensions the blob get smaller and sharper and decrease to one sharp pixel (or peak when considered as a probability distribution). This is because ICA computes independent source signals that are non-gaussian. Non-gaussian leads to high peaks, and the independence to vectors that have their peak at different positions. A interpretation of this result is that ICA represents the original data by the non-white pixels in a space where each non-white pixels forms a block. The reconstruction by this method seems more intuitive than by PCA. The obtained base vectors or also known as signals in the ICA literature are nearly perfect independent. The covariance matrix of the obtained signals differs from the identity by values ranging from $1.88 \cdot 10^{-16}$ for 5 dimension to $6 \cdot 10^{-10}$ for all 784 dimensions. The analysis of the kurtosis confirms the visual results and is given in figure 14. The kurtosis is always $\gg 0$ and increases with more dimensions significantly.[3] A significant difference between PCA and ICA is the run-time. While ICA needs to construct the base vectors for each dimension from ground up, we simply computed PCA once and select number of base-vectors that seem appropriate. The time ICA needs to execute seems to be not very dependent on the dimension we chose, it ranges from 320s for 5 dimensions to 1043s for all 784 dimensions. In contrast PCA needs only 18.9s to execute.

2.2.2 Randomized projection for Digit Dataset

In order to construct randomized projections I use the following technique. Assume our we have n samples of our data, each has d dimensions. We arrange our data column-wise in a data matrix $D \in \mathbb{R}^{d \times n}$. Our goal is to reduce each of our n samples to $k << d$ dimensions. That corresponds to a projections matrix $A \in \mathbb{R}^{k \times d}$, with the rows of A containing the base vectors of the subspace we project on. The question arises: What makes a good projection matrix? While PCA and ICA construct A w.r.t. to optimize certain goals described above, the randomized projection technique just generates A randomly. It is preferable to be A an orthonormal matrix, so that the projection is although not isometric but at least norm-decreasing. That ensures that the data is not arbitrarily rescaled in the new subspace, which could decrease performance of the neural network training otherwise. To generate a random orthonormal matrix we first generate A randomly and apply afterwards the QR decomposition to $A^T = QR$, with $Q \in \mathbb{R}^{d \times d}$ and $R \in \mathbb{R}^{d \times k}$ is a upper triangular matrix. The nice property of this decomposition is, that cause R is a upper triangular matrix, the span of the first k columns of Q equals the span of A^T, so the first k rows of Q^T form an orthonormal basis for the span created by the rows of A, i.e. the subspace we would like to project on. Please note that the first k rows of Q^T can be computed very fast and efficiently by the reduced QR decomposition.[4]. After we obtained several random orthonormal projection matrices $B \in \mathbb{R}^{k \times d}$ by this procedure, we would like to select the best one w.r.t. the reconstruction error of our data. The reconstruction error

[3] this corresponds to the reduction of the blobs to pixels

[4] command qr(A',0) in MATLAB

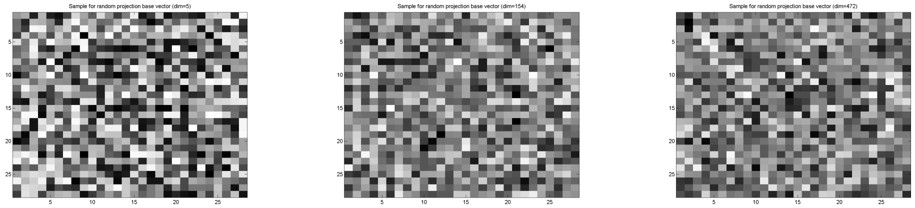

Figure 15: Example of randomized base vector for the Digit-Set for dimensions=5,154 and 472

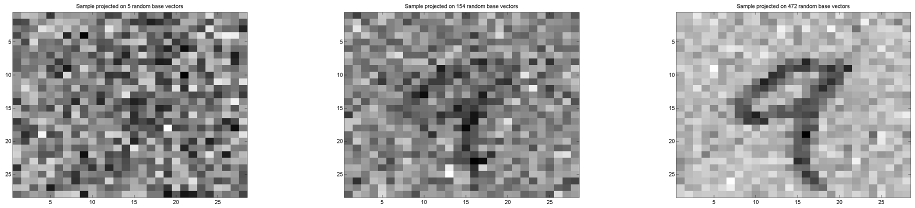

Figure 16: Example of reconstruction of digit '9' for dimensions=5,154 and 472

is given by the difference between of the projected data and the original data. In matrix notation is this the sum of the row-wise norm of $D - B^T BD$.[5]

It turned out that in general many runs are necessary to obtain results, that might reconstruct the original data. I choose 2000 runs for each dimension, but probably a much higher number should be chosen to give better results. The computational time for RP is prohibitive high and magnitudes larger than the time needed for ICA and RP(it takes hours to execute) Figure 16 shows samples of the random found base vectors, and it can be clearly seen that not much structure is present. With more dimensions the original data can be reconstructed as figure **??** shows. This is not very surprising, cause the base vector may be random but still orthonnormal. Because the span of 784 arbitrary random orthonormal vectors is still the whole space, it could reconstruct 28 × 28 grayscale image perfectly.

2.2.3 Dimension reduction for cancer dataset

In the case of the cancer dataset it is not possible to show pictures and to conclude whether the principal components make sense or not. Analysis showed that the obtained principal components have negative entries and comparison to the original values is due to rescaling not possible. However in figure 17, that shows the distribution of the eigenvalues, it can be seen that already one attribute captures the data very well. This is corresponds to the results we obtained from the clustering of the data, that e.g. the radius is already a discriminative feature. Furthermore only one third of the attribute measure something (the last 20 attributes are just the standard deviation and the mean of the three largest values for the first 10 attributes) and nearly half of these attributes are a combination of the others. The ICA of the cancer dataset produced for k=2 two independent signals, with a high kurtosis, as shown in the right figure 17. However it is not possible to relate one of these signals to one of the clusters we obtained above cause the signals are independent, while the clusters we obtained above seem to be related by a scalar multiple of each other. Randomized projections can project the data on different subspaces, but we have seen that in the case of the Digit dataset, that this does not lead to satisfactory results. Because I don't use this dataset for neural network training and it is very hard to describe the results of the RP for this dataset, except plotting a graph showing a decrease of the reproduction error with increasing dimensions there is nothing more to say about the RP for this dataset.

[5] This could also be computed by projecting D on the orthogonal complement $B^\perp$, but in the case of $k << d$ this take longer to compute

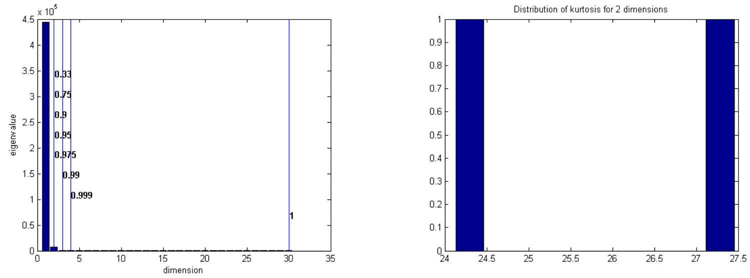

Figure 17: Left: Distribution of eigenvalues of covariance matrix for PCA. Right: Example of kurtosis for dimension 2

2.2.4 Naive dimension reduction

Many different dimension reduction algorithms are possible. I took Harris feature points into consideration for the digit dataset, but it turned out that the image patches are too small to compute them reliable. Instead I used two naive methods. The first just uses the mean and the variance of the intensity of the images patches. Assuming that each digit has another mass or amount of black pixels, this may lead to satisfactory classification results by just two variables! The second method is bilinear downsample by factor 4 to a 7x7 patch. In the case of the cancer dataset, the cluster and dimension reduction results showed that is data is highly redundant and that the radius of the cancer cells is a highly discriminative feature. So I reduced the data from 30 dimensions to 2, by just including the mean and the variation of the radius.

2.3 Neural network training

I have chosen the digit dataset for neural network training, cause the cancer dataset is too easy for classification and does not lead to good comparisons. In my first assignment I showed that the network configuration of 2 layers of each 50 nodes performed best for classification. I examined classification without and with regularization, i.e. inclusion of a regularization term that penalizes huge weights in the nodes. For the dimension reduction algorithms I applied the same projection to the training and the test set. For the cluster algorithms I computed the closest cluster for each test sample. In the case of k-means I used a hard assignment (take closest one), in the case of EM I used soft-assignments (inverse square of distance to each cluster center times probability of the cluster center, normalized w.r.t. to $\updownarrow_1$ norm). The figure 18 compare the training time for the network and figure 19 compares the classification accuracy for the different dimension reduction algorithms and the clustered data. The classification accuracy and training time without any preprocessing is given by the horizontal blue line.[6] The training time is highly volatile, I assume cause of the cache-functions of the operating system. But clearly fewer dimensions make training in general faster. In the case of the clustered data training time is faster then the standard time without regularization, but different values for k does not seem to make much difference.

The EM-algorithm performs very bad, I assume this is because the soft-assignment of the test-data does not work well. Although a hard-assignment does not improved the accuracy. The accuracy would probably improve by applying EM to training and test data at the same time and calculating the soft assignments, but this is not a realistic test case, cause we would use the test data to create the model. K-means performs very well and improves accuracy with increasing k. Very interesting is the performance of the dimension reduction algorithms. First for some dimensions they perform better than the original data, so the effort to use them is justified. PCA's accuracy increases with higher dimensions, obtains a maximum and decreases rapidly with higher dimensions. It is shown above that the principal components of smaller eigenvalues tend to zero and tend to be noisy, so the more of these principal components are included the worse is the generalization performance cause we start representing also the idiosyncrasies of our training data. ICA's accuracy also increases with higher dimensions, but becomes saturated, i.e. more dimensions do not improve the accuracy. This is because ICA computes independent, non-gaussian signals. While in the case of PCA projection

[6] Please note that these are not the values obtained in assignment 1, cause I preprocessed the data different

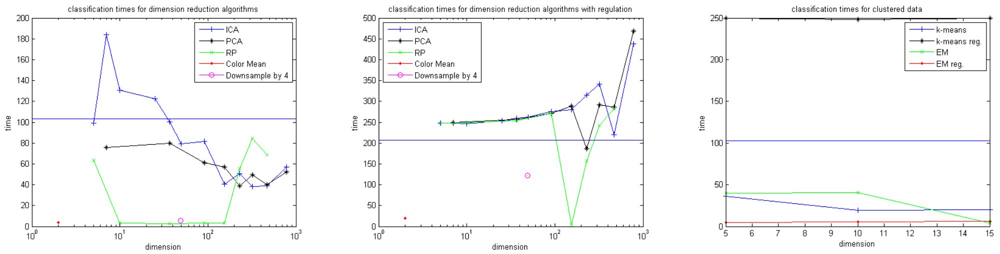

Figure 18: Training times for dimension reduction algorithms without and with regularization and for clustered data

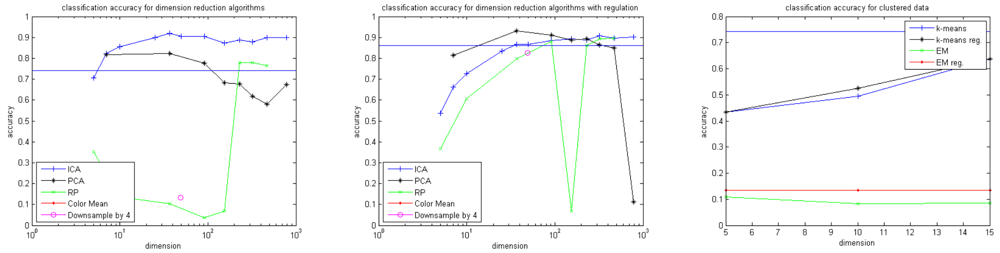

Figure 19: Classification accuracy for dimension reduction algorithms without and with regularization and for clustered data

onto noisy principal components lead to non-zero coefficients, projection on unused independent components lead to coefficients near zero and does not tamper the classification algorithm. The accuracy of randomized projections also increases with more dimensions and is also able to outperform the original data. I showed that only around 20% of the pixels carry information, projection on random orthogonal subspaces should increase performance. But because the construction is completely random and does not follow any algorithm it performs worse than ICA or PCA. The naive dimension reduction algorithm of taking the mean color, performs very bad, but performs at least better than random guessing. Boosting it could be valuable. The down-sample algorithm performs pretty good, it is a bit worse than the original data, but magnitudes faster to train. This was expected cause bilinear downsampling does not destroy many details in the case of digits, because they are often missing small details and just the overall shape represents a digit.

References

[1] http://yann.lecun.com/exdb/mnist/.

[2] http://en.wikipedia.org/wiki/Mammography.

[3] ftp://ftp.ics.uci.edu/pub/machine-learning-databases/breast-cancer-wisconsin/.

[4] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[5] S. S. Keerthi and C.-J. Lin. Asymptotic behaviors of support vector machines with gaussian kernel. *Neural Comput.*, 15(7):1667–1689, 2003.

[6] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. Available from: citeseer.ist.psu.edu/article/lecun98gradientbased.html.

[7] T. M. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[8] W. Street, W. Wolberg, and O. Mangasarian. Nuclear feature extraction for breast tumor diagnosis, 1993. Available from: citeseer.ist.psu.edu/street93nuclear.html.