

Classification of handwritten digits using supervised locally linear embedding algorithm and support vector machine

Olga Kouropteva *, Oleg Okun, Matti Pietikäinen

Machine Vision Group, Infotech Oulu and
Department of Electrical and Information Engineering
P.O.Box 4500, FIN-90014 University of Oulu
FINLAND

Email: {kouropte, oleg, mkp}@ee.oulu.fi

Abstract. The locally linear embedding (LLE) algorithm is an unsupervised technique recently proposed for nonlinear dimensionality reduction. In this paper, we describe its supervised variant (SLLE). This is a conceptually new method, where class membership information is used to map overlapping high dimensional data into disjoint clusters in the embedded space. In experiments, we combined it with support vector machine (SVM) for classifying handwritten digits from the MNIST database.

1 Introduction

Dimensionality reduction is an important preprocessing step before classifying multidimensional data. The locally linear embedding (LLE) algorithm [3] [4] has been recently proposed for this purpose. Its attractive properties are: 1) only two parameters to be set, 2) optimizations not involving local minima, 3) preservation of local geometry of high dimensional data in the embedded space, 4) a single global coordinate system of the embedded space.

In this paper, we describe a supervised variant of LLE, called the supervised locally linear embedding (SLLE) algorithm [2]. Unlike LLE, SLLE projects high dimensional data to the embedded space using class membership relations. This allows obtaining well-separated clusters in the embedded space (if the dimensionality of the embedded space is less by one than the number of classes

*Olga Kouropteva is grateful to the Infotech Oulu Graduate School (Finland) for a financial support.

[2]); moreover, each cluster is represented by only one point in the embedded space.

To test SLLE, we coupled it with support vector machine (SVM) [5], since SVM has provided excellent results in many tasks. SVM performs classification by mapping data into a high dimensional space where classes are linearly separable by hyperplanes. The combination of SLLE and SVM was applied for recognizing handwritten digits of the MNIST database [1]. Our results bring open questions for further research, which are also briefly discussed.

2 Locally linear embedding algorithm

As an input, LLE takes a matrix \mathbf{X} of size $D \times N$ consisting of N columns representing feature vectors. Its output is a matrix \mathbf{Y} of size $d \times N$ ($d \ll D$), where columns are coordinates of the feature vectors in the embedded space. Further, the term *point* will stand for a vector either in R^D or R^d space, depending on the context.

The LLE algorithm contains the following three steps:

Step 1. K nearest neighbors are found for each point X_i in R^D , $i = 1, 2, \dots, N$ by using Euclidean distances to measure similarity. Then the proximity matrix \mathbf{A} of size $K \times N$ is built and its i -th column holds indices of K points, which are the nearest to X_i (A_{1i} corresponds to the highest proximity).

Step 2. Assigning a weight to every pair of neighboring points. The weights representing contributions to the reconstruction of a given point from its nearest neighbors can be found by solving the optimization task [3]:

$$\epsilon(\mathbf{W}) = \sum_{i=1}^N \left\| X_i - \sum_{j=1}^N W_{ij} X_j \right\|^2, \quad (1)$$

subject to constraints $W_{ij} = 0$, if X_i and X_j are not neighbors and $\sum_{j=1}^N W_{ij} = 1$. The latter condition enforces rotation, translation, and scaling invariance of X_i and its neighbors.

Step 3. Computing embedded coordinates Y_i for each X_i . Projections are found by minimizing the embedding cost function for the fixed weights [3]:

$$\delta(\mathbf{Y}) = \sum_{i=1}^N \left\| Y_i - \sum_{j=1}^N W_{ij} Y_j \right\|^2, \quad (2)$$

under the following constraints: $\frac{1}{N} \sum_{i=1}^N Y_i Y_i'$ (normalized unit covariance) and $\sum_{i=1}^N Y_i = 0$ (translation-invariant embedding), which provide a unique solution. To find the matrix \mathbf{Y} , a new matrix $\mathbf{M} = (\mathbf{I} - \mathbf{W})'(\mathbf{I} - \mathbf{W})$ is constructed and its d bottom eigenvectors starting from the second one are computed. These eigenvectors span rows of \mathbf{Y} .

3 Supervised locally linear embedding algorithm

The essence of SLLE consists of the following. The whole data set Ψ is divided into subsets $\Psi_1, \Psi_2, \dots, \Psi_m$ so that $\Psi = \Psi_1 \cup \Psi_2 \cup \dots \cup \Psi_m$ and $\Psi_i \cap \Psi_j = \emptyset, \forall i \neq j$. Each Ψ_i holds the data of one class only and m is the total number of classes known a priori. Let each Ψ_i be associated with its own data matrix Ξ_i , which contains a set of N_i D -dimensional feature vectors.

Each Ψ_i is treated separately from others as follows. For each data point $X_i \in \Psi_1$, we seek its K nearest neighbors belonging to Ψ_1 , i.e., X_i and its neighbors are contained in the same class, while *the nearest neighbors are chosen from arbitrary classes in the LLE algorithm*. When applied to all $X_j \in \Psi_1$, this procedure leads to a construction of the matrix \mathbf{A}_1 of size $K \times N_1$, where columns correspond to the points and rows correspond to the nearest neighbors. By repeating the same for Ψ_2, \dots, Ψ_m , matrices $\mathbf{A}_2, \dots, \mathbf{A}_m$ are generated.

To distinguish points belonging to different classes, we add a shift to values of all elements of the matrices starting from \mathbf{A}_2 . The shift for all elements of \mathbf{A}_2 is equal to N_1 , the shift for all elements of \mathbf{A}_3 is $N_1 + N_2$, ..., and the shift for all elements of \mathbf{A}_m is $\sum_{i=1}^{m-1} N_i$. Such a procedure guarantees that no two matrices \mathbf{A}_i and \mathbf{A}_j will make reference to the same point.

Having set elements of all matrices, we concatenate \mathbf{A}_i 's into a single matrix \mathbf{A} whose size is $K \times N$, where now $N = \sum_{i=1}^m N_i$. We also concatenate Ξ_i 's into a single matrix \mathbf{X} whose size is $D \times N$. After that, operations are carried out as in case of LLE starting from Step 2.

At a first glance, it may be difficult to estimate the influence of the changed procedure for the selection of nearest neighbors on the final embedding. The change is small, yet significant, since the matrix \mathbf{A} is used to construct another matrix, \mathbf{W} , which, in turn, determines embedded coordinates. As a result, embeddings obtained with the unsupervised and supervised LLE are different.

4 Experiments

The MNIST database of handwritten digits [1] was used in experiments. It consists of ten different classes of grayscale images (from '0' to '9', each of 28×28 pixels in size) together with their class labels. The training and test sets consist of 60,000 and 10,000 images, respectively. The number of images in the training set varies from 5,842 to 6,742 per class and it is about 1,000 images per class in the test set. Each image was transformed into a D -dimensional feature vector ($D = 28 \times 28 = 784$), where pixels values were used as features.

From the previous work [2] we learnt that for good data separation the dimensionality of the embedded space, d , should be less by one than the number of classes, L . Since we have ten classes, it is obvious that d should be set to 9. Moreover, there is no significant influence of the parameter K on good data separation by SLLE; $K = 18$ was used in the experiments. The meaning of "good separation" is that all points from the same class in the high dimensional space are mapped into one point in the embedded space. Mathematically, it

- 1) Map the training set into d -dimensional space ($d = L - 1$) by using SLLE.
- 2) Train SVM on L d -dimensional vectors, each of which represents a certain class.
- 3) Map the test set by applying the non-parametric generalization [4].
- 4) Use the projection obtained in the previous step and the trained SVM to classify the test data.

Figure 1: A procedure of the application of SLLE+SVM for classification.

Table 1: Confusion matrix in % when classifying in the original space.

Class	'0'	'1'	'2'	'3'	'4'	'5'	'6'	'7'	'8'	'9'
'0'	99.3	0	0.1	0.2	0	0.1	0.1	0	0.2	0
'1'	0	99.4	0.2	0.1	0	0.1	0.1	0.1	0.1	0
'2'	0.7	0.1	97.7	0.1	0.1	0	0.4	0.6	0.4	0
'3'	0	0	0.3	97.7	0	0.5	0	0.5	0.7	0.3
'4'	0.1	0	0.4	0	98.4	0	0.2	0	0	0.9
'5'	0.2	0	0	1.0	0.1	97.8	0.3	0.1	0.2	0.2
'6'	0.5	0.2	0.2	0	0.2	0.5	98.1	0	0.2	0
'7'	0	0.6	0.9	0.1	0.1	0	0	97.5	0.1	0.8
'8'	0.4	0	0.2	0.4	0.3	0.2	0.1	0.4	97.6	0.3
'9'	0.2	0.4	0	0.4	0.9	0.4	0	0.4	0.3	97.0

means that the best solution for the minimization problem (2) was found, i.e., $\delta(\mathbf{Y}) = 0$.

The SVM with a polynomial kernel $[\gamma \langle X_i, X_j \rangle + c]^\xi$ was then used for data classification in the high and low dimensional spaces, where ξ 's were set to 2 and 1 for the original and embedded spaces, respectively, γ 's and c 's were set to 1 in both cases.

We first trained the SVM classifier using 60,000 images in the original space. Then 10,000 images were fed to the classifier and the confusion matrix shown in Table 1 was obtained (the average accuracy of recognition is 98.05%).

Next, we combine SLLE and SVM (see Figure 1) for classification. In this case, the SVM training was very fast since $L \ll N$ and the output of SLLE immediately gave us support vectors. Table 2 represents the confusion matrix obtained for the classification in the embedded space (the average accuracy is equal to 97.06%).

To compare the performances of SVM and SLLE + SVM on a more difficult data, the training and test sets were interchanged, i.e. the number of images in the training set was 6 times less than that in the test set. Moreover, the number of classes was reduced to five ('1', '3', '7', '8' and '9'). These digits are the most difficult to recognize due to their shape similarity. SVM was applied to the data lying in the original space and in the 4D embedded space obtained by SLLE, respectively. In the latter case the unseen data points were mapped into the embedded space by the non-parametric generalization algorithm [4].

Table 2: Confusion matrix in % when classifying in the embedded space.

Class	'0'	'1'	'2'	'3'	'4'	'5'	'6'	'7'	'8'	'9'
'0'	99.3	0.6	0	0	0	0	0	0.1	0	0
'1'	0	99.7	0.2	0.1	0	0	0	0	0	0
'2'	0.6	1.3	97.3	0	0	0	0.1	0.7	0.1	0
'3'	0	2.4	0.1	96.2	0	0.4	0	0.6	0.3	0
'4'	0.1	1.3	0	0	97.0	0	0.3	0.1	0	1.1
'5'	0.1	3.0	0	0.3	0	96.3	0.1	0.1	0	0
'6'	0.2	1.2	0	0	0.1	0.1	98.4	0	0	0
'7'	0	1.9	0.5	0	0.1	0	0	97.1	0	0.4
'8'	0.2	4.4	0.2	0.4	0	0.1	0	0.2	94.2	0.3
'9'	0.2	3.2	0	0.2	0.8	0	0	0.5	0.1	95.0

The achieved accuracies were 96.59% and 95.86%, respectively.

Despite of these unfavorable circumstances, the combination SLLE + SVM has its merits. SLLE arranges all points of the same class lying in the original space into one point of the embedded space, i.e. a unique representation is obtained for each class in the embedded space. This fact allows a cheaper way to train SVM, since only one vector of dimensionality d per class is needed for this stage. In our experiments when SVM was trained on the D -dimensional data, 7749 and 957 support vectors were found for ten and five class' problems, respectively, whereas these numbers are 10 and 5 after preprocessing with SLLE. Moreover, in the latter case the data are linearly separable, i.e. only a linear SVM is needed, which means that a problem-dependent kernel tuning is avoided.

One can see from the results of the two series of experiments that when the recognition task becomes more difficult and the number of support vectors significantly decreases, the accuracy of SVM falls more rapidly ($98.05\% - 96.59\% = 1.46\%$) than that of SLLE + SVM ($97.06\% - 95.86\% = 1.2\%$). Furthermore, the difference between the accuracies of SVM and SLLE + SVM becomes smaller: 0.99% and 0.73% for 10 and for 5 classes, respectively. However, more research on different data sets would be needed to confirm these observations.

5 Conclusion

The original LLE assumes that data lie on one manifold embedded in a high dimensional space. In this paper, we extended the concept of LLE to the case of multiple disjoint manifolds by proposing a supervised variant of LLE. It can be used as a preprocessing step before classification in order to obtain a compact representation for classes while reducing dimensionality. SLLE allows computing support vectors for SVM without training SVM on the huge amount of data. By comparing the performance of SVM and SLLE + SVM for recognition of handwritten digits, SLLE + SVM failed to overcome SVM

alone operating in the high dimensional space. Perhaps, this indicates that the effect known as the curse of dimensionality did not manifest itself in the large extent for the task considered. It also implies that further research is definitely needed to establish how much SLLE (mapping + generalization) contributes to the classification accuracy.

LLE (and moreover, SLLE) is a new technique and its efficient implementation for large-scale problems ($N \approx 10^5$ and higher) did not yet come into existence. This is concerning both the nearest neighbor search and eigendecomposition. If an efficient solution for these two problems will be found, the time spent on dimensionality reduction can be significantly shortened.

References

- [1] The MNIST database. <http://yann.lecun.com/exdb/mnist/index.html>.
- [2] O. Kouropteva, O. Okun, A. Hadid, M. Soriano, S. Marcos, and M. Pietikäinen. Beyond locally linear embedding algorithm. Technical Report MVG-01-2002, Machine Vision Group, University of Oulu, 2002.
- [3] S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.
- [4] L.K. Saul and S.T. Roweis. Think globally, fit locally: unsupervised learning of nonlinear manifolds. Technical Report MS CIS-02-18, University of Pennsylvania, 2002.
- [5] V. Vapnik. *The nature of statistical learning theory*. Springer Verlag, 1995.