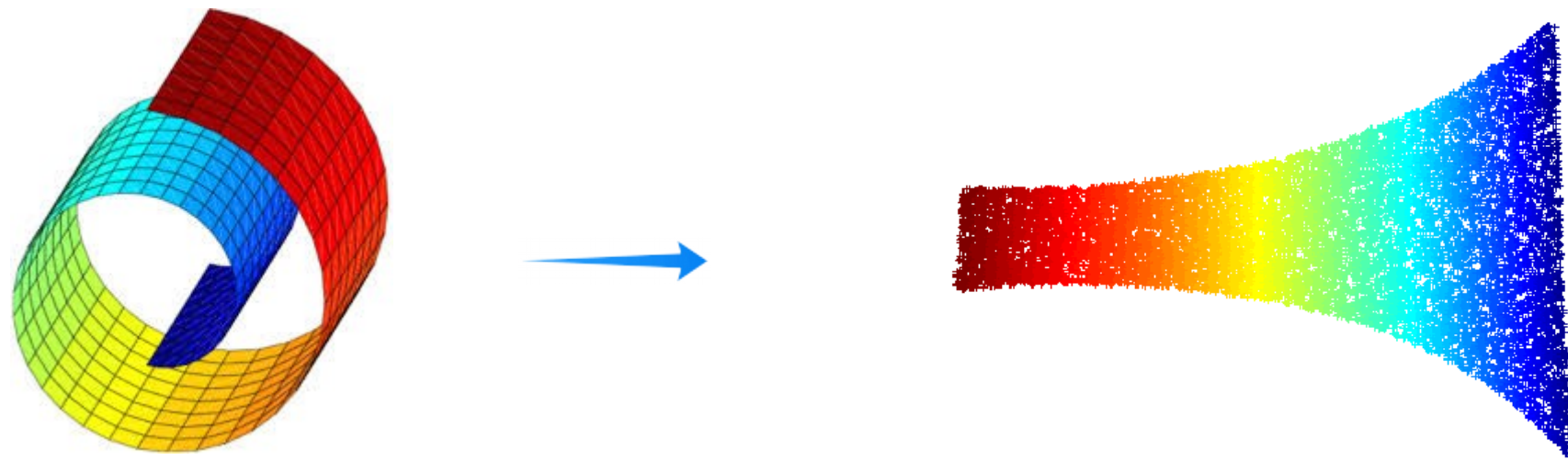


NonLinear Dimensionality Reduction in Pattern Recognition Applications



- Elaboration: Katsileros Petros
- Supervision: Nikolaos Pitsianis, Nikos Sismanis



Introduction

- Introduction about Dimensionality Reduction
- Presentation sections
 - Why we need information compression
 - Dimensionality Reduction Algorithms (PCA, SVD, ISOMAP, Laplassian Eigenmaps, LLE)
 - The Locally Linear Embeddings Algorithm (LLE)
 - **Surpass LLE limitations with two new variations of LLE algorithm**
 - Experiments using native LLE and the two new methods
 - Datasets: MNIST, SVHN, Arcene
 - **Results visualization and explanation**
 - Future work
 - Conclusion



Why we need information compression

- Example: Image with size [480,640]
 - That is equal to a vector with size: $480 \times 640 = 307200$.
 - For a “small” dataset, $N = 100K$ we have a matrix of size: 100.000×307200
- Very large Computational complexity
- Very large memory requirements
- A large amount of these pixels are just Noise, affecting negative machine learning or image processing algorithms.
- Why dont we just simulate the Human brain ...
- How? Using keypoints into each image
 - Features like SIFT, HoG, PFH etc ...
 - Find out which pixels have meaningful information
- Dimensionality reduction techniques are able to extract d (ex. $8 < d < 256$, from 480×640) meaningful key points



Dimensionality Reduction Algorithms

- Linear:

- **Principal Component Analysis (PCA)**

- Minimize a mean square error equation (sum of eigenvalues)
 - Produces statistical independent features

- Multi Dimensional Scaling (MDS)

- **Singular Value Decomposition (SVD)**

- Solving the equation: $X = U_r \Lambda^{\left(\frac{1}{2}\right)} V_r^H$

- Non-Linear:

- Isometric Mapping (ISOMAP)

- Laplassian-Eigenmaps

- Locally Linear Embeddings



NonLinear Dimensionality Reduction – Locally Linear Embeddings

- Step-1: Find K-Nearest Neighbors for each data (Adjacency matrix)

- Executed in CUDA

- Step-2: Find Weights matrix (W)

- Minimize cost function: $\arg \min E_w = \sum_{i=1}^N \|X_i - \sum_{j=1}^N W(i, j) X_j\|^2$

- $E = \|\vec{x} - \sum_j w_j \vec{x}_j\|^2 = \|\sum_j w_j (\vec{x} - \vec{x}_j)\|^2 = \sum_{jk} w_j w_k C_{jk}$

- Gram matrix: $C_{jk} = (\vec{x} - \vec{x}_j) \cdot (\vec{x} - \vec{x}_k)$

- Solve the system: $\sum_j C_{jk} \cdot w_k = 1$



NonLinear Dimensionality Reduction

– Locally Linear Embeddings

- Step-3: Find the embedding coordinates, using the weights matrix W

- Minimize cost function: $\arg \min E_y = \sum_{i=1}^N \|Y_i - \sum_{j=1}^N W(i, j) Y_j\|^2$

- $E_y = \|(I - W)Y\|^2 = Y^T M Y$

- Find eigen-values of the square $[N \times N]$ **sparse** matrix

- $M = (I - W)^T (I - W)$

- Step-4: Keep the final embedded coordinates

- Discard the λ_0 eigenvalue (equal to zero)

- Keep the rest d eigen-values. The eigen-vectors are the final embedded d coordinates



LLE Limitations

- Eigen-value decomposition step has complexity $k \cdot O(N^2)$
 - For sparse matrix M, solving with Lanczos algorithm
- We must run LLE algorithm with both train and test datasets as input data
 - Dimensionality reduction on train and separate on test produces spaces with different base vectors
 - We cannot find any relation between the low dimensional test and train data

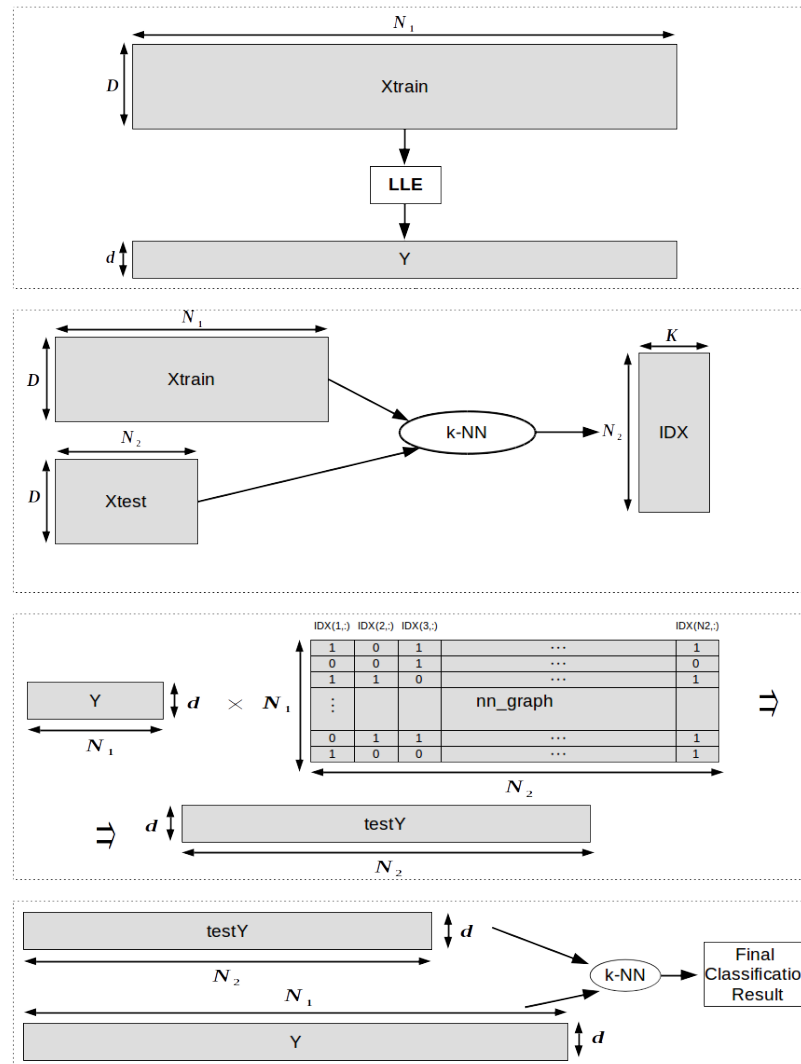


Method-1: LLE with Test-Projection

- Apply LLE dimensionality reduction on Train dataset $[D \times N_1]$
 - Result: matrix Y of size $[d \times N_1]$
- K-NN between Test dataset $[D \times N_2]$ and Train dataset
 - Result: matrix IDX of size $[N_2 \times K]$
- Create the adjacency zero matrix nn_graph of size $[N_1 \times N_2]$
- For every column in nn_graph set $IDX(\text{column}, 1:K)$ elements equal to ones
- Execute the matrix multiplication: $Y[d \times N_1] \times nn_graph[N_1 \times N_2]$
 - Result: matrix $testY$ of size $[d \times N_2]$
- Final step, execute classification algorithm (k-NN) between dimensional reduced test ($testY$) and train (Y) datasets



Method-1: LLE with Test-Projection

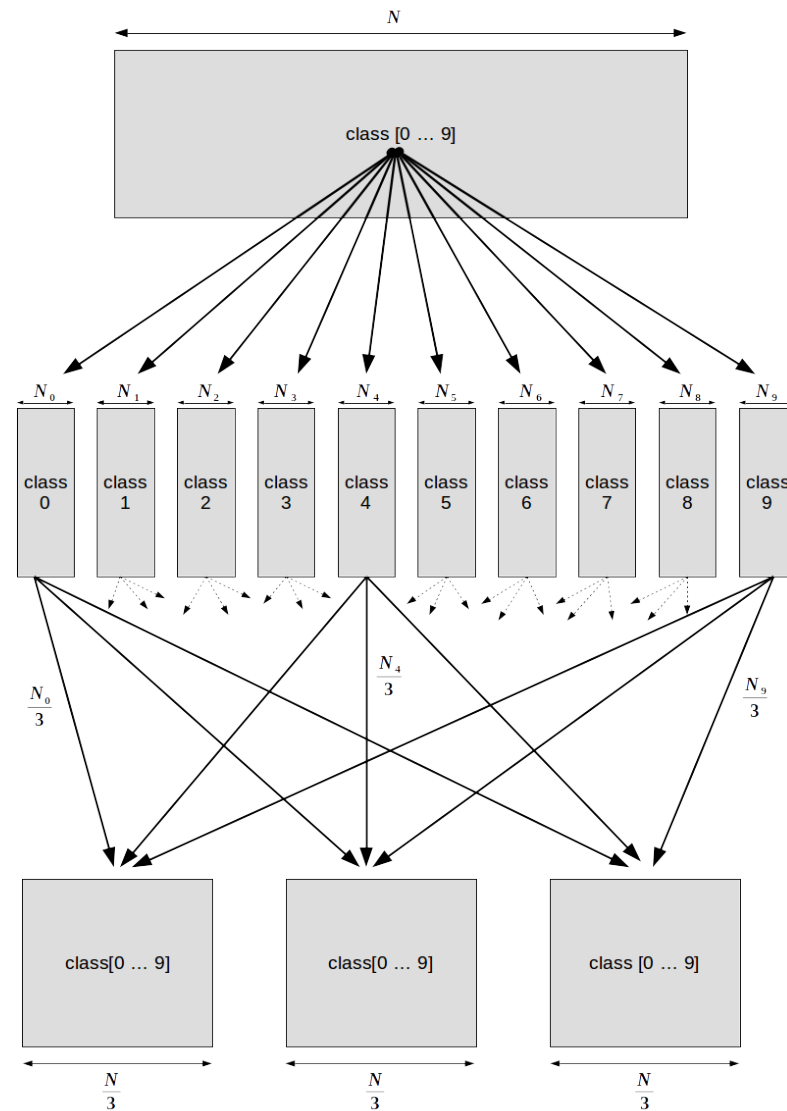


Method-2: LLE with SubSpace Voting

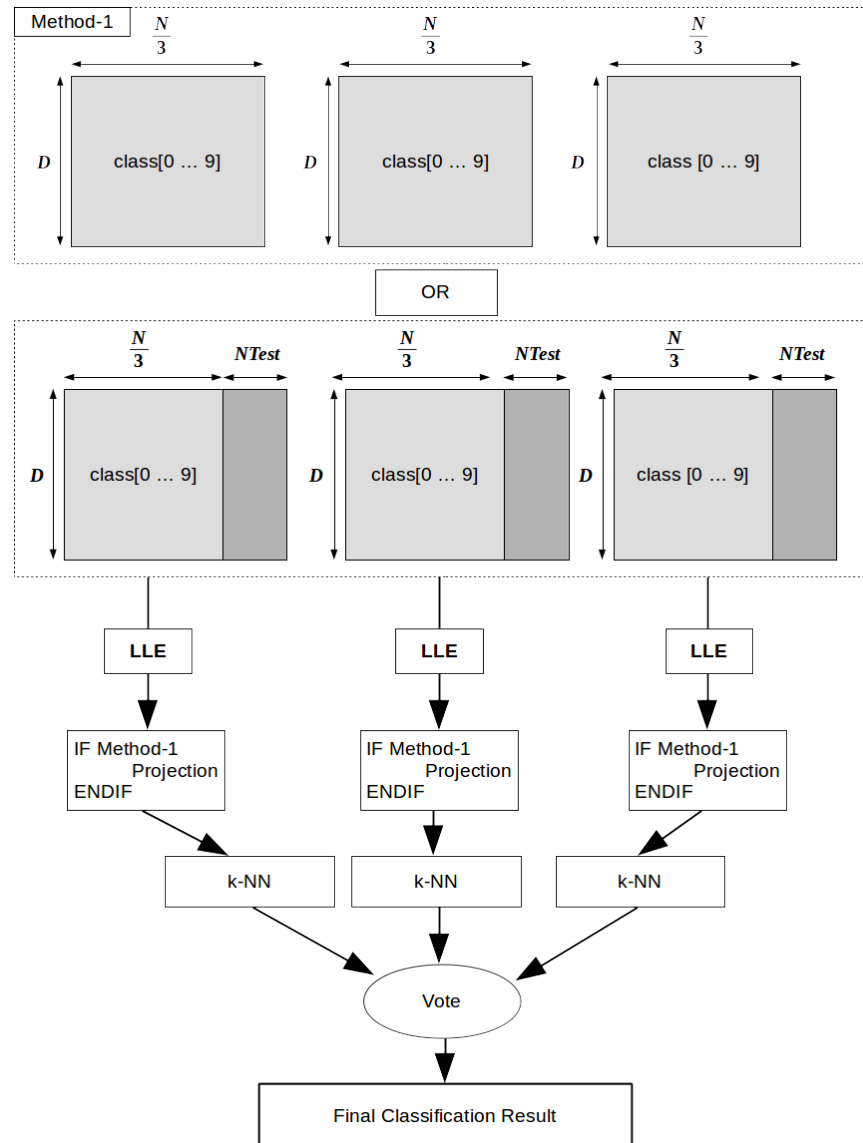
- Split Train dataset into sub-sets
 - Sub-sets MUST contain equal information for every class
- Execute dimensions reduction using LLE on every sub-set
 - Method-1 can be used, in order to avoid including Test data into each sub-dataset
- Execute classification process (k-NN) for every Test data
- The final classification result depends from the majority class after each sub-space voting



Method-2: LLE with SubSpace Voting



Method-2: LLE with SubSpace Voting



Experiments

- Datasets: MNIST, SVHN, ARCENE
 - MNIST: Gray scale images of handwritten digits.
 - 60K Train data, 10K Test data, $D=[28 \times 28]$
 - SVHN: Google street view house numbers $[32 \times 32]$ RGB images.
 - 73257 Train data, 26032 Test data, $D=[32 \times 32]$ (531131 additional, somewhat less difficult Train data)
 - Arcene: Cancer dataset with a large number of predefined features for each patient.
 - 200 Train data, 700 Test data, $D = 10K$.
- Classification algorithm: k-NearestNeighbors
- Classification metric: Mean average % error



MNIST Experiments

- LLE parameters k, d. Method-2 parameter: batch_size
 - 1st-Exp: K = [6, 7, 8, 9, 10, 12, 16, 20, 24, 32, 64], d = [10, 16, 20, 24, 32, 40, 52, 64, 96, 128, 256], subSet_size=[60000,30000,20000] of Train dataset
 - Classification error using k-NN (k=2): **K=12, d=128, subSet_size=60K**, qual to **3.06%**
 - Classification error using k-NN (k=2) without dimensionality reduction: **3.5%**
 - **This is also a VERY important result: K=8, d=10, subSet_size=60K** equal to **3.31%**
 - From the above results, we can say that LLE algorithm can be used as a feature extraction process with great data compression ability.



MNIST Experiments

- Method-2 results are very impressive:
 - K=16, d=256, batch_size=20K. Best classification error equal to **3.27%**.
 - K=10, d=128, batch_size=10K. Best classification error equal to **3.31%**.
 - **Huge reduction both in time and space.** (Step-3 of LLE has $O(N^2)$ complexity)
- Method-1: Not impressive results, but this is the only realistic use of LLE dimensionality reduction in “Real time” application
 - K=8, d=256, batch_size=60K. Best classification error equal to 3.85%.



SVHN Experiments

- Extract HoG features due to noise and light distortions.
 - Split image into sub sections and calculate the gradient of pixel intensity.
- 1st-Exp: $K = [8, 10, 12]$, $d = [16, 20, 32, 64, 96, 128, 164, 196, 256]$, 30K of SVHN Train data-set
 - HoG features parameters: Kernels size: $[2 \times 2]$, $[4 \times 4]$, $[8 \times 8]$ produce features of length: 8100, 1764, 324 .
 - **Best parameters for SVHN dataset: $K=12$, $d=32$, kernel= $[4 \times 4]$.**
- 2nd-Exp: $K=12$, $d=32$, kernel= $[4 \times 4]$. Full SVHN dataset (73257 Train data, 26032 Test data)
 - Classification error with LLE dimensionality reduction is equal to: **16.67%**.
 - Classification without dimensionality reduction ($D=1024$) is equal to: **17.00%**.
 - The above classification results came from k-NN Classification algorithm and $k=8$.



SVHN Experiments

- 3rd-Exp Method-1: LLE parameters: $K=12$, $d=32$, $\text{kernel}=[4 \times 4]$. 42K of SVHN Train dataset
 - Classification error with LLE dimensionality reduction on Train data and projection for Test data is equal to 18.34%
 - Classification error with LLE dimensionality reduction on Train+Test data is equal to 16.67%
 - Classification error without dimensionality reduction is equal to 17.00%
- 4th-Exp Method-2: LLE parameters: $K=12$, $d=32$, $\text{kernel}=[4 \times 4]$. 30K of SVHN Train dataset.
Method-2 parameters: 3 subspaces
 - Classification results for each of 3 subspaces: 20.19%, 19.05%, 19.97%
 - Classification after subspace voting: **18.30%**



SVHN Experiments

- 5th-Exp Method-2: LLE parameters: $K=12$, $d=32$, $\text{kernel}=[4 \times 4]$. 30K of SVHN Train dataset.
Method-2 parameters: 5 subspaces
 - Classification results for each of 5 subspaces: 20.28%, 21.11%, 20.93%, 20.92%, 21.11%
 - Classification error after subspace voting: **18.37%**
 - Classification error for 3 subspaces (4th-Exp) is equal to **18.30%**
- 6th-Exp Method-2: LLE parameters: $K=12$, $d=32$, $\text{kernel}=[4 \times 4]$. 30K of SVHN Train dataset.
Method-2 parameters: 10 subspaces
 - Classification error after subspace voting: **18.58%**
- 7th-Exp Method-2: LLE parameters: $K=12$, $d=32$, $\text{kernel}=[4 \times 4]$. 30K of SVHN Train dataset.
Method-2 parameters: 20 subspaces
 - Classification error after subspace voting: **19.13%**



ARCENE Experiments

- 1st-Exp: $K = [10, 12, 16, 20, 24, 32, 64]$, $d = [10, 16, 20, 24, 32, 40, 52, 64, 96, 128]$
 - Lack of test labels. 150 Train patients and 50 Test patients and 10K features for each one.
 - For every K and d (≤ 96) combinations, classification error after LLE dimensionality reduction is by far better than the $D=10K$ dimensional space
 - Best classification error for parameters $(K=10, d=10)$, $(K=10, d=52)$, $(K=12, d=128)$ equal to **10%**
 - Classification error without dimensionality reduction ($D=10K$) equal to **76%**.
 - **With 10 of 10K features, we gain a boost of 14% classification accuracy.**
 - **Accurate classification result with 90% probability**



Future Work

- Parallelization: Even though k-NN algorithm both in LLE and in Classification process is executed in CUDA, further parallelization can be achieved from sub-set splitting into threads.
- Extend to large datasets and invest the behaviour of subspace voting
- Find the cutting edge between number of subsets-Train dataset size and Test data size
- Maybe a very accurate dimensionality reduction algorithm for Medical image retrieval.



Conclusion

- LLE produces impressive results after huge dimensionality reduction.
- Huge space and time savings at Final Classification Step
- Method-1 can give as real time Classification into the low dimensional Train and Test spaces
- Method-2 achieves dramatically reduction on execution sources. Both space and time.
- Method-1 and Method-2 makes the execution of LLE algorithm available for normal PCs.
- LLE algorithm can be used both as feature extraction and feature selection process.
- LLE has the ability to remove noise-data, producing very accurate classification results



Thank You

