IE 506: Machine Learning - Principles and Techniques **Jan-May 2023**

End-term Project Report : Joint feature selection and subspace learning

Team Name: Vikings *Team Members: 200260040,200260045*

Abstract

Dealing with data with large feature space comes along with the curse of dimensionality. Hence, there is a need for dimensionality reduction. Dimensionality reduction can be done by feature selection or subspace learning. In this project, we attempt to understand and investigate a novel framework for subspace learning which attempts to perform subspace learning and feature selection simultaneously by using a sparsity inducing objective function.

1 Introduction

Dimensionality reduction algorithms can be roughly classified into two categories: Feature selection and subspace learning. Feature selection involves selecting the relevant features and eliminating the rest while in subspace learning we attempt to learn a subspace by taking combinations of features. Examples of well-known subspace learning algorithms are Principal Component analysis, Linear Discriminant Analysis etc. It has been shown that such algorithms can be incorporated into a common framework by using a graph embedding approach. There also exist variations of the above algorithms which can be used to learn sparse sub spaces. However in most of these cases, the selected features where different for different samples (depending on the pattern of sparsity). In this framework we attempt to learn sub spaces which are sparse along features i.e have feature value as 0 for all samples using the properties of the $l_{2,1}$ norm.

In the remaining part of this project, we look into relevant literature before briefly introducing the graph embedding framework and looking at an objective function for joint feature selection and subspace learning (FSSL) based on it. We then give details of the experiments followed by concluding remarks.

2 Literature Survey

Our project is based on the algorithm proposed by the paper [2]. It draws inspiration from subspace learning algorithms proposed by PCA paper and LDA paper as well as sparse subspace learning algorithms [4]. The proposed algorithm is based on a graph embedding framework proposed by [3]. In this paper, a general view for subspace learning has been provided using the concept of graph embeddings wherein we construct a graph using data points connected with each other by a weights represented as an adjacency matrix. They have shown equivalence of this view with well known linear subspace learning algorithms like PCA and LDA. Convergence guarantees of the proposed algorithms and relevant mathematical background has been given in [1].

3 Methods and Approaches

3.1 Work done before mid-term project review

3.1.1 Notations

We denote the data matrix $\mathbf{X}$=$[\mathbf{x}_1, \mathbf{x}_2 ... \mathbf{x}_n] \in \mathbb{R}^{d \times n}$. The aim is to learn a projection matrix $\mathbf{A} \in \mathbb{R}^{d \times m}$ thus projecting the data to an m-dimensional subspace. The ith row of $\mathbf{A}$ is denoted by $\mathbf{a}^i$ and the jth column of $\mathbf{A}$ is denoted by $\mathbf{a}_j$. The Frobenius norm of $\mathbf{A}$ is defined as $||A||_F = \sqrt{\sum_i^d ||\mathbf{a}^i||_2^2}$, the $l_{2,1}$ norm is defined as $||\mathbf{A}||_{2,1} = \sum_i^d ||\mathbf{a}^i||_2$ and the $l_{1,1}$ norm is defined as $||A||_{1,1} = \sum_i^d ||\mathbf{a}^i||_1$

3.1.2 Graph embedding view of subspace learning

In the graph embedding view, for linear dimensionality reduction the optimal $\mathbf{A}$ is given by the following optimisation problem:

$$\min_{\mathbf{A}} tr(\mathbf{A}^T \mathbf{X} \mathbf{L} \mathbf{X}^T \mathbf{A}) \tag{1}$$

$$\text{where } \mathbf{A}^T \mathbf{X} \mathbf{D} \mathbf{X}^T A = \mathbb{I} \tag{2}$$

To do feature selection it is desirable to have some rows of the projection matrix be all zeros. Here $D_{ii} = \sum_j W_{ij}$ is a diagonal matrix and $\mathbf{L}$=$\mathbf{D}$-$\mathbf{W}$ is called the graph Laplacian. This problem is solved using generalized eigen-decomposition $\mathbf{X}\mathbf{L}\mathbf{X}^T A = \Lambda \mathbf{X}\mathbf{D}\mathbf{X}^T A$ where Λ is a diagonal matrix of eigen values. For the linear Graph embedding framework the different choices for $\mathbf{W}$ lead to different linear dimensionality methods like PCA, LDA, LPP etc. For LDA, if there are c classes and the kth class has n_k samples such that $n_1 + n_2 + ... n_k = n$, then

$$W_{ij} = \begin{cases} \frac{1}{n_k} & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ belong to kth class} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

For LPP the weight matrix is:

$$W_{ij} = \begin{cases} d(x_i, x_j), & \text{if } \mathbf{x}_j \in \mathcal{N}_k(\mathbf{x_i}) \text{ or } \mathbf{x}_i \in \mathcal{N}_k(\mathbf{x_j}) \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Here, $\mathcal{N}_{\|}(x_i)$ is the set of k nearest neighbors of $\mathbf{x_i}$ and $d(\mathbf{x_i}, \mathbf{x_j})$ is a measure of the similarity between the 2 vectors $\mathbf{x_i}$ and $\mathbf{x_j}$.

3.1.3 Joint Feature selection and Subspace learning

We want row sparsity in the projection matrix. This motivates us to minimise the $l_{2,1}$ norm which promotes row sparsity. Hence, the formulation becomes:

$$\min_A \ ||A||_{2,1} + tr(A^T X L X^T A) \tag{5}$$

where the constraint $\mathbf{A}^T \mathbf{X} \mathbf{D} \mathbf{X}^T A = \mathbb{I}$ holds as before. Since, the constraint is not convex, optimising directly is difficult. Hence, we look for an equivalent formulation. For this formulation, the following theorem is useful:

Theorem 3.1 *Let* $\mathbf{Y} \in \mathbb{R}^{n \times m}$ *be a matrix of which each column is an eigenvector of the eigen-problem* $\mathbf{W}y = \lambda \mathbf{D}y$. *If there exists a matrix*$A \in \mathbb{R}^{d \times m}$ *such that* $\mathbf{X^T A} = \mathbf{Y}$, *then each column of* $\mathbf{A}$*is an eigenvector of eigen-problem* $\mathbf{XWX^T}a = \lambda \mathbf{XDX^T}a$.

A proof of this theorem is given in (cite). Using this theorem, we can solve the eigen-problem $\mathbf{XWX^T A} = \mathbf{\Lambda XDX^T A}$ to get A as follows:

- Solve the eigen problem $\mathbf{WY} = \mathbf{\Lambda DY}$ to obtain $\mathbf{Y}$
- Find A which satisfies $\mathbf{X^T A} = \mathbf{Y}$

Since $\mathbf{X^T A} = \mathbf{Y}$ is a linear system, it can have infinitely many solutions, one solution or no solution. Hence depending on this, we can formulate the problem as shown below.

3.1.4 Algorithm 1

When the linear system has infinitely many solutions the proposed formulation is:

$$\min_{\mathbf{A}} ||\mathbf{A}||_{2,1} \tag{6}$$

where $\mathbf{X^T A} = \mathbf{Y}$ The lagrangian of the above objective is:

$$L(\mathbf{A}) = ||\mathbf{A}||_{2,1} - tr(\mathbf{\Gamma^T}(\mathbf{X^T A} - \mathbf{Y})) \tag{7}$$

Minimising the lagrangian by setting the derivative with respect to $\mathbf{A}$ to zero implies:

$$\frac{\partial L(\mathbf{A})}{\partial \mathbf{A}} = \mathbf{GA} - \mathbf{X\Gamma} = 0 \tag{8}$$

Here, $\mathbf{G}$ is a diagonal matrix where:

$$g_{ii} = \begin{cases} 0 & \text{if } \mathbf{a^i} = \mathbf{0} \\ \frac{1}{||\mathbf{a^i}||_2} & \text{otherwise} \end{cases} \tag{9}$$

Assuming $\mathbf{G}$ is invertible, if we multiply both sides of equation (8) by $\mathbf{X^T G^{-1}}$ and using the constraint $\mathbf{X^T A} = \mathbf{Y}$ we get:

$$\mathbf{\Gamma} = (\mathbf{X^T G^{-1} X})^{-1} \mathbf{Y} \tag{10}$$

If $\mathbf{G}$ is not invertible, we can add a smoothing term ϵ to g_{ii}. If a decreasing value of ϵ is used the algorithm converges to a global solution. Hence we get (by substituting equation 10 into 8)

$$\mathbf{GA} - \mathbf{X}(\mathbf{X^T G^{-1} X})^{-1} \mathbf{Y} = 0 \tag{11}$$

$$\Rightarrow \mathbf{A} = \mathbf{G^{-1}X}(\mathbf{X^T G^{-1} X})^{-1} \mathbf{Y} \tag{12}$$

In the code, we initialise $\mathbf{G} = \mathbb{I}$, then we compute $\mathbf{A}$ based on equation (12) and then using this value we update the value of $\mathbf{G}$ using equation (9). The updated value of $\mathbf{G}$ is then used to update $\mathbf{A}$ (eqn.12) and this process is repeated until convergence. The convergence of this algorithm was proved in (nie et al).

Situation 1 and its code was covered in pre-midsem review

3.2 Work done after mid-term project review

After the mid-term review, we looked into the case when linear system has a unique solution or no solutions.

3.2.1 Algorithm 2

When the linear system has a single unique solution or no solution then also it is a solution of (6). However, it does not have row sparsity. Hence the reformulated problem is:

$$\min_{\mathbf{A}} ||\mathbf{A}||_{2,1} \tag{13}$$

such that $||\mathbf{X^T A} - \mathbf{Y}||_F^2 \leq \delta$ or equivalently:

$$\min_{\mathbf{A}} ||\mathbf{A}||_{2,1} + \mu||\mathbf{X^T A} - \mathbf{Y}||_F^2 \tag{14}$$

As before if we set the derivative to zero we get:

$$A = 2\mu(\mathbf{G} + 2\mu\mathbf{X}\mathbf{X^T})^{-1}\mathbf{X}\mathbf{Y} \tag{15}$$

We can simplify this further using the Woodbury Matrix identity:

$$(\mathbf{A} + \mathbf{UCV})^{-1} = \mathbf{A^{-1}} - \mathbf{A^{-1}U}(\mathbf{C^{-1}} + \mathbf{VA^{-1}U})^{-1}\mathbf{VA^{-1}} \tag{16}$$

from which we can obtain:

$$\mathbf{A} = \mathbf{G^{-1}X}(\mathbf{X^T G^{-1}X} + \frac{\mathbf{1}}{\mu}\mathbb{I})^{-1}\mathbf{Y} \tag{17}$$

3.2.2 Using other norms for the objective function

Instead of using the $l_{2,1}$ norm we can use other norms like $l_{1,1}$ norm or frobenius norm. We investigate the effects of using these norms in the ipython notebook submission. In such cases, for both algorithm 1 and 2 the only change appears in the form of $\mathbf{G}$ (while differentiating the norm). For Frobenius norm the form of $\mathbf{G}$ becomes:

$$g_{ii} = \begin{cases} 0 & \text{if } ||\mathbf{A}||_\mathbf{F} = \mathbf{0} \\ \frac{1}{||\mathbf{A}||_F} & \text{otherwise} \end{cases} \tag{18}$$

For the l_{11} norm the form of $\mathbf{G}$ becomes:

$$g_{ij} = 1 \text{ for all i and j} \tag{19}$$

Note that for l_{11} norm $\mathbf{G}$ is not a diagonal matrix.

4 Data set Details

For the experiments the following dataset where used:

- Yale Face subset: A set of 165 images of 15 subjects. There are 11 images per subject, one for each of the following facial expressions or configurations: 'happy', 'normal', 'surprised', 'sad', 'sleepy', 'wink', 'centerlight', 'leftlight', 'rightlight', 'glasses', 'noglasses'. The data is resized to 32x32 grey images. Each face is represented as a 1024-dimensional vector.
- MNIST-784 dataset: The MNIST database of handwritten digits with 784 features (from 0 to 9). It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image with dimensions 28x28. Each image is represented as a 784-dimensional vector. For the purposes of our code we test on data from 2 classes '1' and '9'.

5 Experiments

We investigated the FSSL algorithm using both LDA and LPP type adjacency matrices for both the datasets. We compare the perfomance of FSSL(LDA) with other subspace learning algorithms like PCA and LDA (using scikit-learn) using accuracy. We selected p number of samples from each class as the training set and the rest of the samples as the test set. For dataset 1, p=5,6,7 and for dataset 2, p=10,20,30. The training set is randomly chosen and hence the experiment is repeated 20 times to calculate average accuracy for FSSL-LDA. The training set was used to learn a subspace after which recognition was done by a 1-Nearest Neighbor classifier. The baseline model is 1-Nearest Neighbor. μ can be selected by tuning and the smoothing term ϵ is set to 0.01. For PCA, the number of components was set to capture 99% variance. We also studied the effect of value of μ on accuracy. We also look into the case where l_{11} and Frobenius norm were used and their effect on accuracy. All tests were done in Python on a 11th Gen Intel(R) Core(TM) i7-11370H @ 3.30GHz, Windows 11 Machine.

6 Results

- For Yale Face database we obtained low accuracy on baseline as well as PCA, LDA and FSSL-LDA. The accuracy was $\approx 16\%$. Possible causes were due to lower samples per class in the data (1 sample per class).
- For MNIST dataset, we obtained high accuracy of $\approx 98\%$ in baseline, as well as PCA,LDA and FSSL-LDA. FSSL-LDA gave competitive results in spite of having the subspace with lowest dimensionality.
- In both cases, it was observed that the value of μ did not have significant correlation with accuracy.
- Using Frobenius norm or $l_{1,1}$ norm in place of $l_{2,1}$ did not have significant differences in accuracy in both cases.

7 Future Work

The above frameworks can be explored in kernel space. We would also like to explore the frameworks on other forms of large data.

8 Conclusion

In summary we looked at a novel algorithm which attempts to perform joint feature selection and subspace learning. We saw the motivation behind this was to maintain the physical importance of features and also saw why we needed to induce row sparsity in the projection matrix. We also saw the significance of choosing $l_{2,1}$ norm and what are the implications when we replaced it with $l_{1,1}$ and Frobenius norm. The resulting framework gave competitive results to well known subspace algorithms like PCA and LDA for a lower dimensional subspace. We also saw that the the value of mu did not significantly affect accuracy.

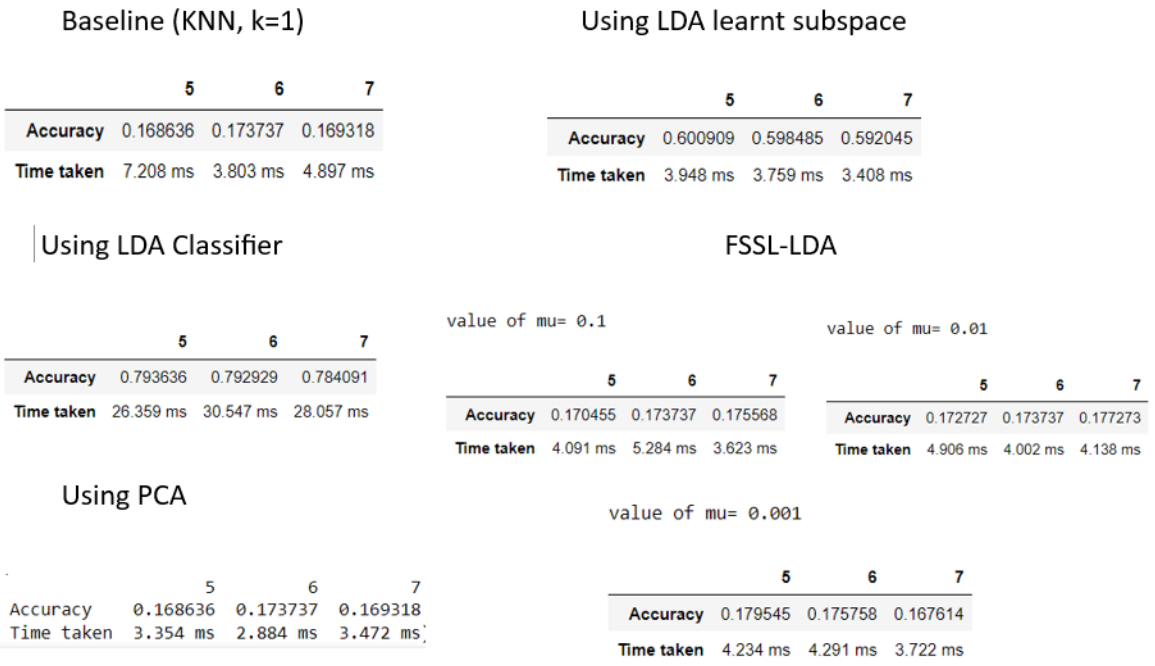

Figure 1: Figure describing accuracies across PCA, LDA, baseline and FSSL-LDA for Yale Face subset

References

[1] Xiaofei He and Partha Niyogi. Locality preserving projections. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16. MIT Press, 2003.

[2] Zhenhui Li Quanquan Gu and Jiawei Han. Joint feature selection and subspace learning. In *Twenty-Second International Joint Conference on Artificial Intelligence*.

[3] Shuicheng Yan, Dong Xu, Benyu Zhang, Hong-jiang Zhang, Qiang Yang, and Stephen Lin. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):40–51, 2007.

[4] Hui Zou, Trevor Hastie, and Robert Tibshirani. Sparse principal component analysis. *Policy*, pages 1–30, 05 2004.

Baseline (KNN, k=1)

	10	20	30
Accuracy	0.967013	0.976725	0.9818
Time taken	824.212 ms	820.323 ms	864.609 ms

PCA

	10	20	30
Accuracy	0.967455	0.977124	0.982098
Time taken	719.809 ms	736.033 ms	720.307 ms

FSSL (LDA)

	10	20	30
Accuracy	0.97001	0.980696	0.985391
Time taken	250.23 ms	257.454 ms	261.497 ms

Figure 2: Figure describing accuracies across PCA, baseline and FSSL-LDA for MNIST dataset

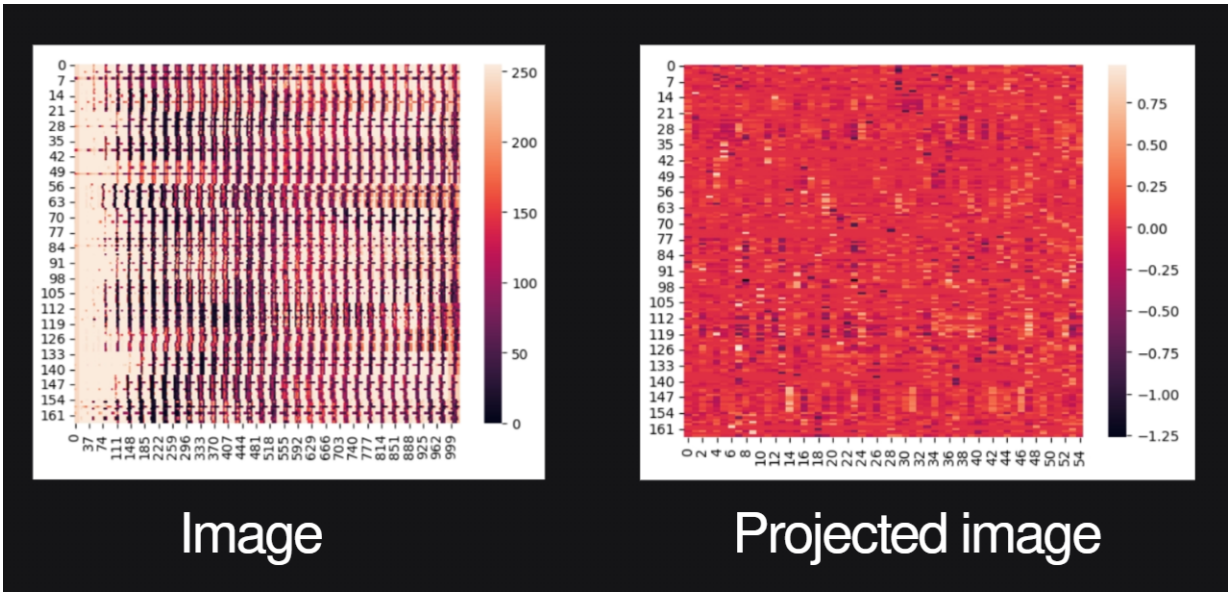

Figure 3: Image and Projected image for Yale Face subset

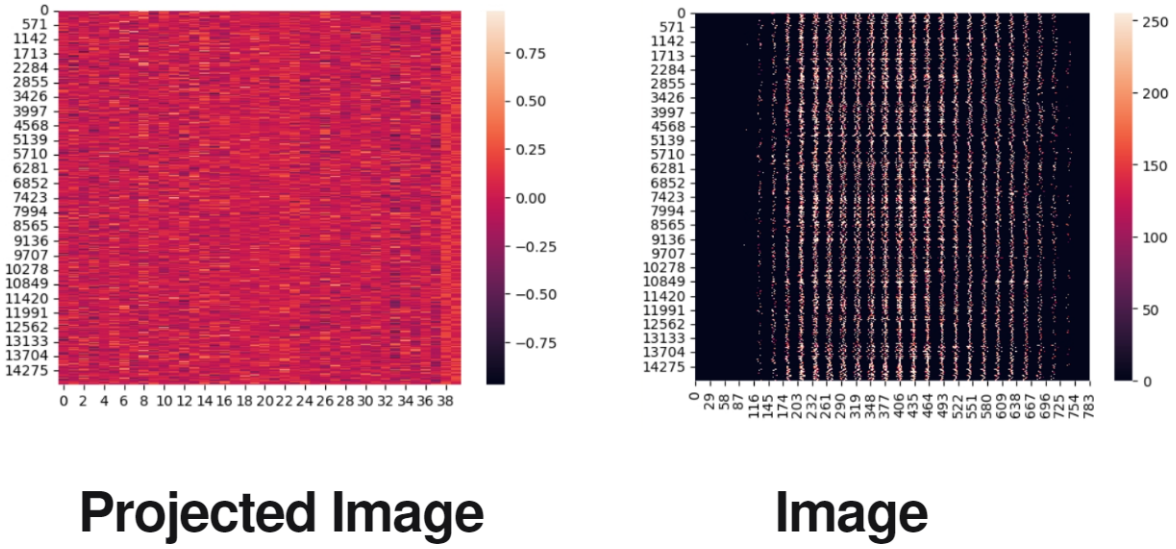

Figure 4: Image and Projected image for MNIST dataset

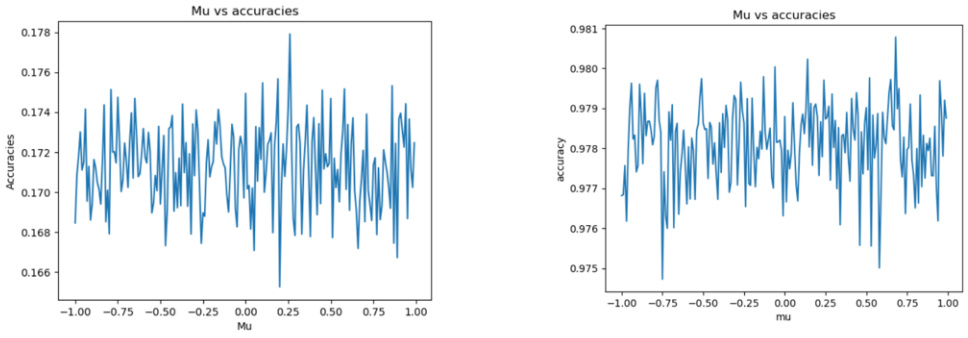

Figure 5: Plots of μ vs accuracies