COMP5328 Assignment1

Meng Zhou SID: 460313182
Zehui Yao SID: 470128664
Jie Lin SID: 490110892

September 2019

Abstract

Non-negative Matrix Factorization is a commonly used Dictionary Learning method to perform dimension reduction, feature extraction and data recovery. In this study, we examine the effectiveness of three different Non-negative Matrix Factorization algorithms (L2 Norm, L1 Norm, L1 Norm Regularization) in handling data with a large magnitude noise or corruption by evaluating their performance on ORL dataset and Extended YaleB dataset in which we manually add some large magnitude noise. In this report, we will firstly introduce the main idea of Non-negative Matrix Factorization in section 1 followed by some related works conducted by other people in section 2. Then we will explain the detailed theory and mathematical formulation of the three Non-negative Matrix Factorization algorithms as well as the noise we added into the dataset in section 3. In section 4, the details and comparison results of the experiments we conducted will be discussed and analysed with insight to their advantages. Finally, we will make a conclusion of the comparison result in section 5 and discuss the future work that could be done.

1 Introduction

NMF, i.e. Non-negative Matrix Factorization[5], is a group of linear algebra algorithms that aim to factorize an original non-negative matrix V into two separate matrices W and H where both of them are also non-negative. As a subclass of Dictionary Learning, Non-negative Matrix Factorization could be applied on various tasks such as dimension reduction, feature extraction and data recovery. And in many real-life applications, some data is only meaningful when it is non-negative (e.g. image, price, weight ...) and in these cases non-negative matrix factorization algorithms are more logical and representative than the other traditional matrix decomposition methods.

Just like other machine learning algorithm, an NMF algorithm $\mathcal{A}$ consists of five key components:

- The input training data is denoted as $S = \{x_1, ..., x_n\}$ where $x_1, ..., x_n$ are set of data that has been encoded into non-negative matrices and the dataset size is n. In the actual implementation, we will first transform the whole dataset into a single matrix V.

- The predefined hypothesis class Y is a set of possible combination of W and H.
- The objective function F that needs to be optimized by the optimizer.
- The optimisation method.
- The final output hypothesis which corresponds to two resulting matrices W and H.

The overall learning algorithm can be formulated as a mathematical mapping:

$$\mathcal{A} = S \in \mathcal{X}^n \mapsto h_S \in Y \tag{1}$$

where $\mathcal{X}$ is the training data domain. Note that all single data entry in the training dataset should have the same dimensions (e.g. width and height for image). In our study, the channel of input images is 1, i.e. all the images are grey-scale.

In this study, we will focus on exploring the applications of NMF on image processing. First, it is intuitive that NMF can be used to compress image data. For an image dataset of big size, the algorithm allows user to transform the original input into a dense vector with a smaller dimension and comprehensive features[3]. The obvious advantage is that the algorithm can help reduce the size of image tensors and make them easier to be further processed (e.g. classification). Secondly, some of the NMF algorithm such as L1 Norm Regularization[7] can be used to remove the noise from the data. In the setting of image, the noise could be different light conditions and occlusion.

In our work, we intend to evaluate the performance and robustness of three different NMF algorithms (L2 Norm, L1 Norm, L1 Norm Regularization) when the image data is contaminated with large magnitude noise, as most data is by nature contaminated with some noise. As the non-negative property grants NMF algorithm great interpretation, the study of NMF algorithms' robustness against contaminated data could lead to the development of better performance in Dictionary Learning.

2 Related Work

To cope with the challenges of high dimensional, negative values, non-sparse and noise, several different variants of Non-negative Matrix Factorisation method have been proposed other than the three methods we are using in our study. In the case where data is missing in some specific location and the position of the noise in data is known, a variant of NMF called Weighted Non-negative Matrix Factorisation (WNMF)[4] could be used. In WNMF algorithm, a matrix W with the same dimension as the data V will be constructed to represent the importance of data, where the value of W in a specific location is set to be 0 if data is missing at that point or corrupted with noise. And by including the weight matrix W in the cost function $\sum_{ij} W_{ij}(V_{ij} - (WH)_{ij})^2$, the effect of missing data and noise will be removed, as it will not contribute to the reconstruction

loss. However, WNMF could be only applied when the position of the noise and missing data is known in advance. In the case where the location is unknown, WNMF will become ineffective. And to mitigate the influence of outliers and errors in the data, L2,1 Norm NMF[8] could be applied. Different from the normal NMF cost function which could be easily dominated by the outliers, L2,1 Norm NMF use L2,1 Norm to regularise both the Reconstruction error and coefficient matrix. $||V - WH||_{2,1} + \lambda||H||_{2,1}$. As the L2, 1 Norm is not squared like L2 Norm, the impact of outliers could be significantly reduced.

3 Methods

In this section, we will first give formal formulations of different non-negative matrix factorisation algorithms then introduce the actual implementation details.

3.1 Preprocessing

We first encode and apply normalization to all the images, i.e. put them into the numerical values within $[0, 1]$. Then we flatten all the image data, i.e. change the tensor shape from 2 dimension to 1 dimension. Moreover, the dimension of the original image is reduced by resizing the image down by a fixed proportion. In our experiments, the proportion is chosen to be 5, which means the height and width of the images is scaled down to one fifth of its original size.

3.2 Frobenius-Norm NMF

We first implement an Frobenius-Norm Based NMF algorithm. It includes proper preprocessing, the cost function design as well as the optimization steps.

3.2.1 Cost Function

In an NMF algorithm, the cost function is a guidance for the optimizer to follow, that is, the optimizer aims to find some W and H that minimizes the predefined cost function $F(V, W, H)$. In this case, the function is in the Frobenius form, formally defined as:

$$F(V, W, H) \ = \ \|V - WH\|_F^2 \tag{2}$$

According to the definition of the Frobenius Norm, i.e. $\|X\|_F = \sqrt{\mathrm{trace}(X^T X)} = \sqrt{\sum_{i=1}^{d} \sum_{j=1}^{n} X_{i,j}^2}$, the above function can be rewritten as:

$$F(V, W, H) \ = \ \sum_{i=1}^{d} \sum_{j=1}^{n} V_{i,j}^2 \tag{3}$$

where $V_{i,j}$ resents the jth flatten image vector element in the ith original image in the given dataset.

3.2.2 Optimization

The optimizer of the NMF algorithm ideally aims to find the optimal W and H that minimize the cost function.

$$W^*, H^* \;=\; \arg \min_{W \in \mathcal{W}, H \in \mathcal{H}} \|V - WH\|_F^2 \tag{4}$$

In the actual implementation, we use multiplicative update rules to optimize the hypothesis. The cost function is not convex towards to both W and H but convex towards to either of them when fixing the other one. Therefore, we optimize the function each time only with respect to one variable, i.e. either W or H.

When fixing W, we use the partial derivative of H to update the cost. It can be written as:

$$\frac{\partial \|V - WH\|_F^2}{\partial H} = -2W^T H + 2W^T WH \tag{5}$$

Similarly, we can find the partial derivative of the cost function with respect to W.

With both partial derivatives, we can optimize the objective step by step with the multiplicative update rules. We first initialize both component matrices W and H in some approaches, for example randomly sampling elements from a normal distribution. Assume in the k-th iteration, W and H have been updated to W^k and H^k. We first fix $W = W^k$ and update the value of H.

$$H_{i,j}^{k+1} \;=\; H_{i,j}^{k} \frac{\left({W^k}^T V\right)_{i,j}}{\left({W^k}^T W^k H^k\right)_{i,j}} \tag{6}$$

Then fix $H = H^{k+1}$ and update the value of W.

$$W_{i,j}^{k+1} \;=\; W_{i,j}^{k} \frac{\left(V {H^{k+1}}^T\right)_{i,j}}{\left(W^k H^{k+1} {H^{k+1}}^T\right)_{i,j}} \tag{7}$$

We can theoretically prove that the above multiplicative update rules will make the cost $\|V - WH\|_F^2$ non-increasing. Also, the cost will always be non-negative. Therefore, after sufficient iterations, the cost change between two adjacent training iterations will be lower than certain constant threshold ϵ, which can be used as a condition to terminate the optimization procedure. Besides, we can set the maximum training steps in advance. When the training step reaches the pre-defined limit, we can stop the optimization.

3.3 L1 Norm Regularization NMF

With the intention to improve robustness, L1 Norm Regularization NMF is specially designed to remove the unknown large addictive noise. Using the L1 Norm Regularization, the basis matrix, coefficient matrix could be learned to approximate the clean

data while predicting the location of the noises. In the following parts of this section, we will discuss the design of its cost function, optimization process and advantages on contaminated data.

3.3.1 Cost Function

To properly define the existence of noise in the data, the observed corrupted data is denoted as V and the noises E could be formulated as $V = \hat{V} + E$ where $\hat{V}$ stands for the clean data. And the objective of L1 Norm Regularization NMF is to represent the clean data $\hat{V}$ using the basis matrix and coefficient matrix. Thus the noise could be injected using the following equation $X \approx WH + E$. In this way, the cost function for L1 Norm Regularization NMF is defined as follows.

$$F(V, W, H, E) = ||V - WH - E||_F^2 + \lambda \sum_j [||E_{.j}||_0]^2 \tag{8}$$

where the second term is used to constrain the sparseness of E and λ is the factor to control the ratio of the constrain. However, this objective function is hard to optimize because of the presence of L0 Norm in the second term. For easier optimization, the above cost function could be converted into the following format with the adaption of L1 Norm.

$$F(V, W, H, E) = ||V - [W, I, -I]\begin{pmatrix} H \\ \binom{E^p}{E^n} \end{pmatrix}||_F^2 + \lambda \sum_j [||E^p_{.j}||_1 + ||E^n_{.j}||_1]^2 \tag{9}$$

where $E^p = \frac{|E|+E}{2}$ and $E^n = \frac{|E|-E}{2}$.

3.3.2 Optimization

At the beginning, the value of matrices W, H and E could be randomly sampled with the constraint that $X - E > 0$. In our implementation, the value of matrices W, H and E are randomly sampled using standard normal distribution with a mean of 0 and a variance of 1. And the matrices are forced to be non-negative by taking the absolute value of the randomly sampled values. The condition $X - E > 0$ is satisfied by setting the upper bound of E to be the value of X in the corresponding position. Similar to the optimization process of the normal NMF algorithm, since the cost function is conditioned on W, H and E, the optimization for W, H and E has to be conducted by fixing the others. And with the form of L1 Norm in the modified cost function, H and E could be updated together. The equations for optimization step are provided below:

$$W_{ij} = W_{ij} \frac{(\hat{V}H^T)_{ij}}{(WHH^T)_{ij}} \tag{10}$$

where $\hat{X} = X - E$

$$\tilde{H} = \begin{pmatrix} H \\ \binom{E^p}{E^n} \end{pmatrix} \tag{11}$$

$$\tilde{H_{ij}} = max(0, \tilde{H_{ij}} - \frac{\tilde{H_{ij}}(\tilde{W}^T\tilde{W}\tilde{H})_{ij}}{(S\tilde{H})_{ij}} + \frac{\tilde{H_{ij}}(\tilde{W}^T\tilde{V})_{ij}}{(S\tilde{H})_{ij}} \tag{12}$$

where $\tilde{V} = \binom{V}{0_{1\times n}}$ and $\tilde{W} = \binom{W,I,-I}{0_{1\times k},\sqrt{\lambda}e_{1\times m},\sqrt{\lambda}e_{1\times m}}$. ($m$ denotes the amount of data and n dimension of a single data)

$$S_{ij} = |(\tilde{W}^T\tilde{W})_{ij}| \tag{13}$$

3.3.3 Advantages

The advantages of L1 Norm Regularization NMF is its ability to process the noise without knowledge about the location of it. After optimization on the corrupted data, the clean data could be reconstructed using the basis matrix and coefficient matrix. Furthermore, the location of the noise is predicted by the matrix E and the author of this algorithm suggests the matrix E could be used with other denoising approach such as WNMF to further reduce the noise in the data.

3.4 L1 Norm NMF

To enhance the robustness against outliers or noise, L1-Norm NMF is introduced to compare the performance with Frobenius-Norm NMF and L1-Norm Regularization NMF. The author claims it has the most robust formulation of error function, but the optimization process is computationally expensive than NMF methods relatively. In the following chapters, we will discuss its cost function and two optimization process as well as the advantages and disadvantages.

3.4.1 Cost Function

The cost function of L1-Norm is the absolute sum of the vectors in a space, formally defined as:[2][6]

$$M(V, W, H) \ = \ \|V - WH\|_1 \tag{14}$$

According to the definition of the L1-Norm, the above function can be rewritten as:

$$M(V, W, H) \ = \ \sum_{i=1}^{d}\sum_{j=1}^{n}|(V - WH)_{i,j}| \tag{15}$$

where $V_{i,j}$ resents the jth flatten image vector element in the ith original image in the given dataset. To reduce the computational cost, we approximate the cost function as:

$$J(W, H) \ = \ |(V - WH)_{i,j}| \approx ((V - WH)_{i,j}^2 + \epsilon^2)^{\frac{1}{2}} \tag{16}$$

where ϵ is a small number. Formally, L1-Norm is formulated as:

$$min_{W,H} J(W, H) \ s.t. \ W \leq 0, H \leq 0 \tag{17}$$

3.4.2 Optimization

1. Addictive update
When using addictive updating method, we first compute the gradients with respect to W and H, and then update them simultaneously with a small learning rate α.

$$gW = -\sigma(X - WH^T)H \tag{18}$$

$$gH = -(\sigma(X - WH^T))^T W \tag{19}$$

where

$$\sigma(y) = \begin{cases} 1 & \text{if } y > 0 \\ 0 & \text{if } y = 0 \\ -1 & \text{if } y < 0 \end{cases} \tag{20}$$

Update W and H

$$W \Leftarrow W - \alpha * gW \tag{21}$$

$$H \Leftarrow H - \alpha * gH \tag{22}$$

2. Multiplicative update
Another updating rule is multiplicative updating, derived from L2,1-Norm NMF[2]. The cost function is convex with respect to either W or H but not to both. We should update one of them while another remains fixed. Updating rules are shown below. (the notation $*$ is the elementwise product between two matrices).
Fix H, update W

$$W_{jk} \Leftarrow W_{jk} \frac{[(X * Q)H^T]_{jk}}{[(WH) * QH^T]_{jk}} \tag{23}$$

Fix W, update$H$

$$H_{jk} \Leftarrow H_{jk} \frac{[W^T(X * Q)]_{ki}}{[W^T((WH) * Q)]_{ki}} \tag{24}$$

where Q is a matrix given by

$$Q_{ij} = ((X - WH)_{ij}^2 + \epsilon^2)^{\frac{-1}{2}} \tag{25}$$

3.4.3 Advantages and disadvantages

The pros of L1-Norm NMF is its robustness against noise, while the cons of L1-Norm is its computational complexity. When using multiplicative updating rule, the converge speed of minimizing objective function is low, which means it takes exponential more training iterations to reach expected error compared with using L2NMF. Also, the cost function of L1-Norm NMF is not convex at the point of 0. Deploying simple $\sigma()$ function can solve this problem, but it also requires carefully choosing a suitable learning rate α in order to converge.

3.5 Noise

To evaluate the robustness of the three NMF algorithms we described above, we manually add three kinds of noise into the clean data. The details for each kind of noise will be discussed in the remaining parts of this section.

3.5.1 Gaussian Noise

Gaussian Noise is one kind of common natural noise which generally interferes pixels of an image[1]. It acts as an ideal approximation of noise in real world. The Gaussian Noise model can be described as below:

$$P(g) = \sqrt{\frac{1}{2\pi\sigma^2}e}^{-\frac{(g-\mu)^2}{2\sigma^2}} \tag{26}$$

In our implementation, we added a random Gaussian Noise on every pixel on the image. In the experiment, we set the mean of distribution μ to 0 and the standard deviation σ to 0.1.

3.5.2 Laplace Noise

Similar to Gaussian noise in distribution, Laplace noise has sharper peak and fatter tails. (The difference between Gaussian noise and Laplace Noise is illustrated in Figure 1) It is widely used in Differential Privacy. We deploy Laplace noise in the experiment to blur the identical features of human faces. The formula shown below represents the Laplace Noise. We set distribution peak μ to 0 and the exponential decay λ to 0.1. Similarly, a random Laplace noise is added to every pixel in the image data in our experiment.

$$f(x|\mu, b) = \frac{1}{2b}exp(-\frac{|x-\mu|}{b}) \tag{27}$$

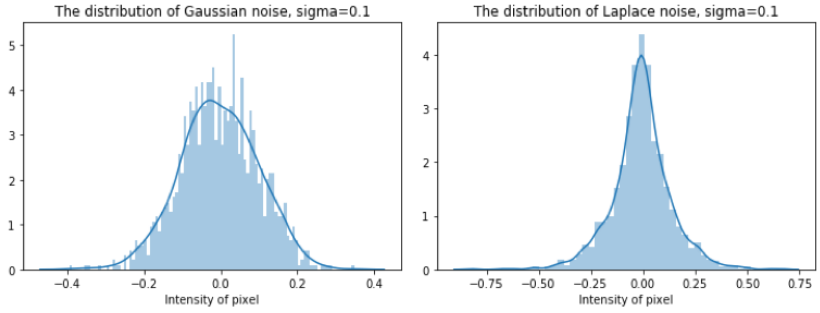

Figure 1: The distribution of Gaussian and Laplace Noise

3.5.3 Block Noise

Also, we use blocks with fixed size at random position to simulate artificial noise, such as glasses and scarf. In our implementation, the value of the block is set to be 1e-6. And we set the size to $\frac{h}{8}$. (h represents the number of vertical pixel of given image) An example of Block Noise is provided Figure 2 below.

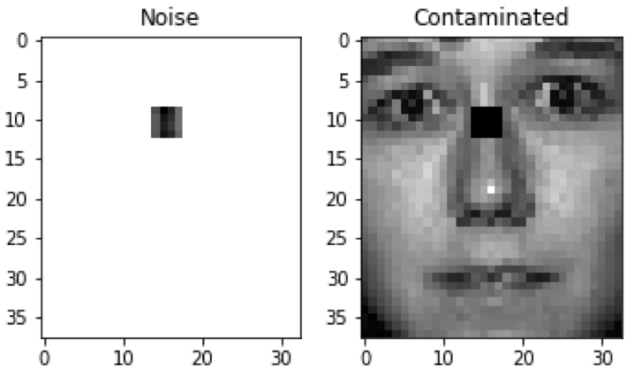

Figure 2: Block Noise Example

4 Experiments and Discussion

In this study, we conduct a series of experiments to prove the validity of the implementation of NMF algorithms. Besides we compare the performance of the three different NMF algorithms under fair comparison.

ORL and YaleB datasets are used in our experiments. With reduce constant = 5, image samples in the ORL dataset loaded into vectors with 396 elements while the YaleB sample vector has 1254 elements. In the experiment, we add random noise into the clean data in three different forms. To evaluate the model performance, we will compare the reconstructed matrix with the original images.

4.1 Metrics

Relative Reconstruction Errors REE criteria describes the similarity of between the reconstructed matrix and the image matrix that has been processed with noise. It is defined as following:

$$\mathrm{RRE} = \frac{\|\hat{V} - WH\|_F^2}{\|\hat{V}\|_F^2} \tag{28}$$

where $\hat{V}$ denotes the clean data. In the evaluation, a lower RRE indicates higher similarity between $\hat{V}$ and the reconstructed matrix WH.

Average Accuracy Each image dataset contains images of different persons. We notice that the number of involved people n_p is greatly less than the total number of image sample n. We employ some clustering algorithms (i.e. K-means) on the reconstructed matrix with the num_clusters equal to n_p and get the prediction. The average accuracy

can be formally expressed as:

$$\mathrm{ACC}(Y, Y_{pred}) = \frac{1}{n}\sum_{i=1}^{n} 1\{Y_{pred}(i) == Y(i)\} \tag{29}$$

Normalized Mutual Information NMI is another acceptable metric, defined as:

$$\mathrm{NMI}(Y, Y_{pred}) = \frac{2 * I(Y, Y_{pred})}{H(Y) + H(Y_{pred})} \tag{30}$$

where $I(.,.)$ represents the mutual information and $H(.)$ is entropy.

4.2 Evaluations

In this section, we will discuss the experiments we conducted to evaluate the relative significance between each algorithm factor (i.e. number of components k and noise) and the algorithm performance.

4.2.1 Influence of Number of Components

We further examined the influence of the number of components k i.e. the reduced dimension of NMF algorithm on the performance of NMF algorithms. To properly evaluate its impact on performance, we trained the three NMF algorithms we mentioned before on the ORL dataset for 300 iterations. The images in the ORL dataset are re-scaled by a factor of 5 and the same Gaussian noise over the entire image is added with a strength factor of 0.1. And we trained each NMF algorithm with a range of number of components value from 20 to 120 with a step size of 5. Through observing the result, we found that the Relative Reconstruction Errors will steadily decrease as the number of components in the NMF algorithm increases. A diagram is provided below to illustrate the effect.

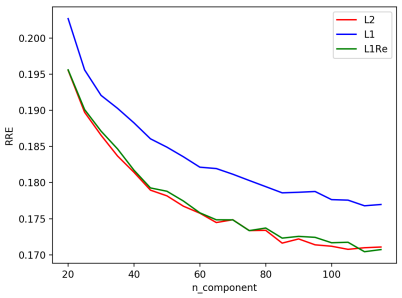

Figure 3: The influence of number of components on RRE.

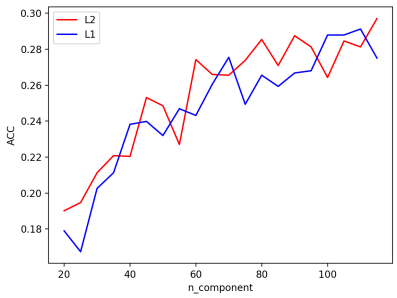

Figure 4: The influence of number of components on ACC.

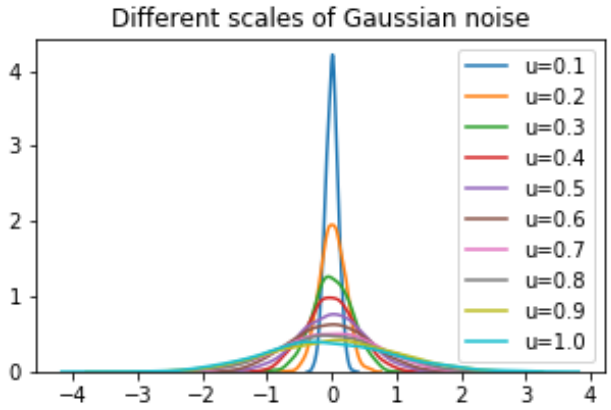

Figure 5: Different standard deviation σ of Gaussian Distribution.

As the resolution on ORL dataset is too low, the same experiment setting conducted on ORL dataset does not show any correlation between number of components and Average Accuracy and Normalized Mutual Information. Thus we switched to YaleB dataset to evaluate the influence of number of components on Average Accuracy and Normalized Mutual Information matrices where K-means is used as the classifier. (Due to the huge amount of data in YaleB and inefficient update of L1 Norm Regularisation NMF, only the L2 Norm and L1 Norm NMF algorithm is tested) As a result, when the number of components is increased in the NMF models, the Average Accuracy and Normalized Mutual Information will both gradually increase. A diagram is provided below to illustrate the effect of number of components on the Average Accuracy.

In general, we can see that as the number of components in the NMF algorithm increases, the performance of the algorithm will also increase. However, as a trade-off, the compression rate of the NMF will decrease in regard of the dimension reduction task.

4.2.2 Influence of Noise Scale

In this section, we compare the robustness of the NMF algorithms mentioned by changing the property of Gaussian noise. We set different value σ to Gaussian noise model to simulate the different intensity of noise. Theoretically, robust NMF algorithms should have the ability to handle with noise with small σ.

To simulate various intensity of Gaussian noise, we set the standard deviation σ from 0.1 to 1 with a step size of 0.1, and evaluate the RRE of three NMF algorithms on ORL dataset. We also set the max iteration to 1000, and randomly sample 20 percent of images on the entire ORL dataset.

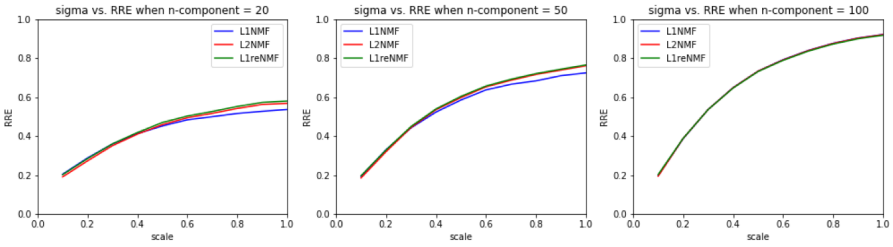

Figure 6: The influence of standard deviation σ on the Relative Reconstruction Error.

As illustrated in Figure 4.2.1, the μ of Gaussian distribution is 0. The distribution of noise gathers at 0 when σ is 0.1 and spreads wider as σ approaches to 1.

With smaller value of n-component, the RRE of L1-Norm increase slower than other NMF algorithms as σ increases. The differences disappear as the value of n-components increases.

4.3 Comparisons between NMF algorithms

noise_type = block	REE	ACC	NMI
Frobenius-Norm NMF	0.171	0.582	0.736
L1 Norm NMF	**0.149**	0.613	0.758
L1 Norm Regularization NMF	0.152	**0.649**	**0.795**
noise_type = gussian	REE	ACC	NMI
Frobenius-Norm NMF	**0.171**	0.582	0.736
L1 Norm NMF	0.178	0.589	0.732
L1 Norm Regularization NMF	0.181	**0.604**	**0.741**

Table 1: Comparison results of baseline algorithms on the ORL dataset.

noise_type = block	REE	ACC	NMI
Frobenius-Norm NMF	**0.167**	**0.302**	0.350
L1 Norm NMF	0.185	0.254	0.339
L1 Norm Regularization NMF	0.182	0.290	**0.368**
noise_type = gussian	REE	ACC	NMI
Frobenius-Norm NMF	**0.173**	0.306	0.378
L1 Norm NMF	0.185	0.290	0.379
L1 Norm Regularization NMF	0.183	**0.322**	**0.381**

Table 2: Comparison results of baseline algorithms on the YaleB dataset.

In this section, we will discuss the experiments we conducted to evaluate the robustness of NMF algorithms by comparing the performance of the three different NMF

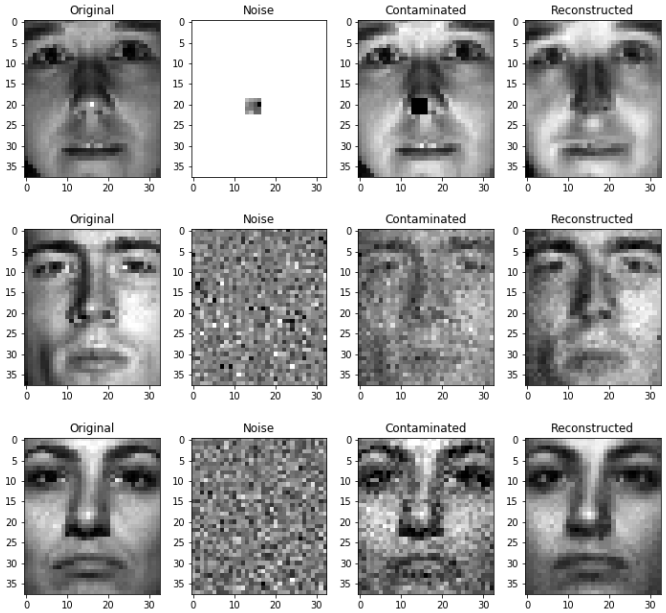

Figure 7: Reconstruction result under different noises

algorithms. Under different configurations, to fairly compare the performance of different NMF algorithms, we need to fix some the factors in one comparison. In our comparative experiments, the noise scale is set to 0.1 and the number of factorization component is set to 100.

For the ORL dataset, as illustrated in Table 1, when we employ the block noise, the L1 Norm Regularization NMF algorithm provides the best performance, winning the competition in ACC and NMI scores, with 0.649 and 0.795 respectively, while L1 Norm NMF reaches the lowest REE. When we employ the gussian noise, the L1 Norm Regularization NMF algorithm also provides the best performance, winning the competition in ACC and NMI scores while Frobenius-Norm NMF reaches the lowest REE (0.171).

For the YaleB dataset, as illustrated in Table 2, when the block noise is used, the Frobenius-Norm NMF algorithm reaches best scores in REE and ACC while L1 Norm Regularization NMF algorithm reaches the highest NMI. When we employ the gussian noise, the L1 Norm Regularization NMF algorithm provides the best performance and has the best ACC and NMI scores while Frobenius-Norm NMF has the lowest REE score (0.173). (The reconstruction result examples of L1 Norm Regularisation NMF under different noises are provided in Figure 7.)

4.4 Summary

From the experiment result, we can conclude that L1 Norm Regularisation NMF is most robust when applied on the data with noise compared with L2 Norm NMF and L1 Norm NMF. We argue this is caused by the fact that L1 Norm Regularisation NMF adapts the extra matrix E in its cost function to approximate the unknown noise inside the data and can remove the effect of the noise through reconstruction. On the other hand, L2 Norm NMF and L1 Norm NMF does not make extra effort in handling the noise in data as shown in their cost function.

5 Conclusion

In this study, we implement three NMF algorithms based on several different objective functions. In addition to extensive experiments to evaluate the model performance under different configuration, we further examined the performance of these three NMF algorithms in the case where large magnitude noise presents in the data. And through comparison experiments, we found the L1 Norm Regularisation NMF shows better robustness against contaminated data because of its novel design in cost function to approximate and remove unknown noise. In the future, we can further explore the study in the following aspects:

- We can implement and evaluate more NMF algorithms such as Weighted NMF [4] and L2,1 Norm NMF [8]
- We can employ more metrics to better evaluate the algorithms.
- When evaluating the Accuracy matrix, more classifiers could be used such as SVM and MLP.

6 Appendix

To run the code for experiments, please refer to Experiment_Guide.ipynb in the code folder for detailed instructions. Every group member has made equal contribution to this assignment by implementing one NMF algorithm, running experiments and writing allocated parts of the report.

References

[1] A. K. Boyat and B. K. Joshi. A review paper: Noise models in digital images processing. *Signal Image Processing : An International Journal (SIPIJ)*, 2015.

[2] C. D. Deguang Kong and H. Huang. Robust nonnegative matrix factorization using l21-norm. 2016.

[3] D. Guillamet and J. Vitria‘. Non-negative matrix factorization for face recognition. *Conference on Artificial Intelligence*, 2002.

[4] Y.-D. Kim and S. Choi. Weighted nonnegative matrix factorization. *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2009.

[5] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorisation. *Conference on Neural Information Processing Systems*, 2001.

[6] T. Liu and I. Dacheng Tao, Fellow. On the performance of manhattan nonnegative matrix factorization. 2016.

[7] B. Shen, L. Si, R. Ji, and B. Liu. Robust nonnegative matrix factorization via l1 norm regularization. *IEEE International Conference on Image Processing*, 2012.

[8] D. Wang, J.-X. Liu, Y.-L. Gao, J. Yu, C.-H. Zheng, , and Y. Xu. An nmf-l2,1-norm constraint method for characteristic gene selection. 2016.