|  |
| --- |
| **Assignment 1** |

**Tutors:** (1. Yu Yao; 2. Xuefeng Li)

**Group members:** (1.Yue Shang, 490104363, ysha5406; 2.Zhecheng, 490329299, zzho7727; 3.Hao Huang, 490498480, hhua6750)

**Abstract**

Non-negative matrix factorization (NMF) explores the non-negativity property of data and has received considerable attention in many fields, such as text mining, hyper-spectral imaging, and gene expression clustering. It decomposes a nonnegative data matrix into the product of two lower dimensional non-negative factor matrices by minimizing the Euclidean distance between their product and the original data matrix.[3] Since NMF only allows additive, non-subtractive combinations, it obtains a natural parts-based representation of the data.[3] Many data contain noises and outliers. Thus, a robust version of NMF is needed.[1] In this paper, we will implement L1-norm NMF algorithms and L2-norm NMF and analyse the robustness of them when the dataset is contaminated by large magnitude noise or corruption. This report is divided into seven sections. Section 1 introduces the main idea and applications of NMF. Section 2 shows the advantages and disadvantages of relevant methods. Section 3 explains the choice of algorithms and choice of noises. Section 4 discusses the details of the experiment, compares different algorithms and evaluates the results. The last section concludes the experiment and gives a suggestion for future work.

**1 Introduction**

The standard NMF method and its variants are often used as a feature extraction technology in various industries, especially for the analysis of high-dimensional data. [4] Applied in image processing (face, medicine, etc.), audio data processing or text mining and decomposition. [4]The main idea is that NMF may be interpreted as factorizing a nonnegative data matrix into two nonnegative matrices[5]By reducing the dimension of the matrix describing the problem, the data stored by the matrix can be processed efficiently, thereby reducing the storage space. There are many analysis methods using matrix decomposition to solve practical problems, such as principal components analysis (PCA), independent component analysis (ICA), singular value decomposition (SVD), vector quantization (VQ), etc. [2]. NMF is different from other methods because it adds non-negative restrictions. These constraints lead to a parts-based representation because they allow only additive, not subtractive, combinations.[2] The NMF approach can be formulated as follows.

Given a nonnegative data matrix whose columns contain data samples, the data decomposition task can be described by factoring into two lower dimensional non-negative data matrix and of size and , The product of and is an approximate estimate of . The mathematical expression is

(1)

The columns of are usually called basis vectors and the rows of are called decomposition (or encoding) coefficients. Thus, the original data are represented as linear combinations of these basis vectors.[4]

To measure the quality of the approximation factorization, a cost function between and needs to be optimized subject to nonnegativity constraints on and. This is done by minimizing the least-squares cost function given by [5]

(2)

The minimization of (1) yields the following multiplicative update rules at th iteration [5]:

and

(3)

There are many different types of non-negative matrix factorizations that use different cost functions to measure the difference between V and WH, or regularization of W and/or H matrices. [8]

In order to make NMF perform better in image processing and other applications, we need to study the robustness of it and its variants. In the following paragraphs, we will compare different models of NMF, and use 𝐿2-norm NMF, 𝐿1-norm NMF to demonstrate the different robustness of the algorithms, and then we analyze how to determine the number of features K. The results will be represented in section 4.

**2 Related work**

NMF is that it is prone to outliers, which is one of the most important disadvantages. Many people have proposed related algorithms. The goal is to improve the robustness of traditional NMF, such as Zhang et al. [17] assumed that the dataset contains both Laplace distributed noise and Gaussian distributed noise and proposed an L1-norm regularized Robust NMF (RNMF-L). Since NMF is essentially a summation of the squared L2-norm of the errors, the large magnitude errors dominate the objective function and cause NMF to be non-robust. To solve this problem, Kong et al. [1] proposed the L2,1-norm-based NMF (L2,1 -NMF ) which minimizes the L2,1 -norm of the error matrix, Guan et al.[3] proposed the Truncated Cauchy NMF, etc.

We briefly review the main ideas and analyse their advantages and disadvantages of 1-NMF and 2-NMF in this part.

**2.1 NMF**

Traditional NMF assumes that noise obeys a Gaussian distribution and derives the following squared 𝐿2-norm based objective function: , It is commonly known that NMF can be solved by using the multiplicative update rule (MUR). Because of the nice mathematical property of squared 2-norm and the efficiency of MUR, NMF has been extended for various applications.[3] During the constraints, the 2-norm constraint plays an important part. This constraint has great potential to feature extraction and data representation, and it has been widely discussed in different areas. In face recognition, the 2-norm-based regularization method is employed to get more discriminative representations [6] the 1-NMF and 2 norms are used to enhance the group sparsity and overcome the problem of weak extrapolating ability in traditional NMF. In addition, the 2-norm can be combined with other norms for particular applications [6] However, NMF and its extensions are non-robust because the 2-norm is sensitive to outliers [3] [7].

**2.2 1-Norm Based NMF**

Lam et al.[3]assumed that noise is independent and identically distributed from Laplace distribution and proposed 1-NMF as follows: , this method has better robustness than traditional NMF. Due to the sparsity, the 1- NMF will ignore the unimportant features, so it is not sensitive to outliers. However, since the 1- norm based loss function is non-smooth, the optimization algorithm in is not scalable on large-scale datasets, and its optimization is expensive. Although L1-NMF is not as sensitive to noise as NMF, its decomposition point is related to the dimensionality of the data, the robustness of L1-NMF is not sufficient enough. Manhattan NMF (MahNMF) solves this problem by approximating the loss function of 1-NMF with a smooth function and minimizing the approximated loss function using Nesterov’s method. [3].

**3 Methods**

Since the purpose of this experiment mainly was to realize two NMF algorithms and to compare their robustness, we adopted NMF and 1-NMF to process the dataset. In addition, the image added different proportions of noise to simulate the data damage or pollution by adverse factors in real working and living environment for comparing the robustness between NMF and 1-NMF.

**3.1 Datasets**

In this experiment, the dataset consisted of ORL dataset and the Extended YaleB Dataset. The ORL dataset included 400 face images from 40 different subjects based on different times of shooting, lighting conditions, facial expressions, and facial details. Extended YaleB dataset included 2414 face images from 38 subjects based on 9 postures and 64 lighting conditions. All face images cropped and scaled to pixels and pixels [12][13].

**3.2 Noise Selection**

In the real work and living environment, the unfavourable factors in the image acquisition and transmission process are the main sources of digital image noise. For example, in image acquisition, higher sensor temperatures may produce more severe noise. During the transmission, lightning interference with the transmission channel causes some images to be damaged[16]. In order to simulate the noise during image acquisition and transmission, we mainly use Gaussian noise and salt and pepper noise.

**3.2.1 Gaussian Noise**

Gaussian noise is a common digital image noise that the probability density function obeys the normal distribution. It can effectively reflect the sensor and electronic circuit noise caused by poor lighting or temperature during the acquisition of digital images [10]

**3.2.2 Salt and Pepper Noise**

Salt and pepper noise are a kind of impulse noise to consist of sparse black and white pixels. It has types of noise(salt noise, pepper noise, and mixed noise between salt and pepper). The salt noise is some abnormally bright pixels in the image (The value of pixel=255). In contrast, the pepper noise is some abnormally dark pixels in the image (The value of pixel=0). It can effectively simulate dead pixels, converter errors, and sudden and sharp interference of image signals caused by status errors in transmission [11]

**3.3 Algorithm Details**

Given a set of face images convert into original non-negative matrix and a factorization rank , the NMF algorithm can decompose the original non-negative matrix into a non-negative matrix in dimension and a non-negative matrix in dimension. Generally, in order to reduce the amount of computation (using few basis vectors to describe a large amount of data), the choice of is less than and , so matrix and the matrix will be less than the original matrix . Through the matrix multiplication, the product of matrix and the matrix only approximate to the original matrix [9]Sparse and Unique Nonnegative Matrix Factorization Through Data Pre-processing. We could use the formula (4) to express:

(4)

In the formula (1), the matrix was a basis vector matrix which contained features extracted from the original matrix . The matrix is a coefficient matrix which contains the weights.

**3.3.1 NMF Algorithm**

NMF use the squares loss to realize. It was based on the square of Euclidean distance to measure the error between the original matrix and the predicted matrix ( ) [9] We can use the formula (5) to express:

(5)

For minimizing the error between the original matrix and the predicted matrix , we need to seek the appropriate matrix and matrix . Lee and Seung found that multiplicative algorithm was easier and better than the gradient algorithm for NMF optimization [14]. Therefore, based on the multiplicative update rule, we can gain the iterative updating formula (6)(7) [15].

(6)

(7)

The optimization process is divided into four steps. **Step 1**: to generate a matrix W; **Step 2**: after fixing matrix W, to interactively updated the matrix H based on formula (4) for the convergence of its result; **Step 3**: after fixing matrix H, to interactively update the matrix W based on formula (3) for the convergence of its result. **Step 4:** repeat step 2 and step 3. Until the error value was unchanged or to meet the allowable error value, NMF optimization completed.

**3.3.2 1-NMF Algorithm**

1-NMF uses the absolute loss function to calculate the sum of the absolute difference between the original matrix and the predicted matrix  ( ). We can use the formula (8) to express:

(8)

To minimizing , we are based on Kong, Ding, and Huang to realize the optimization of 1-NMF. Under their study, will be replaced by . At the same time, is set to a small number. Then, we can gain the iterative updating formula (9), (10)[1].

(9)

(10)

The matrix is based on the formula (11):

(11)

In addition, is an Hadamard product. The optimization process of *L*1-NMF is similar to NMF algorithm.

**3.4 Evaluate Metrics**

We adopt three evaluation metrics to compare the performance and robustness between NMF and 1-NMF algorithms.

**3.4.1 Relative Reconstruction Errors (RRE)**

Given the original images  and the contaminated images . Using the basis vector matrix and the coefficient matrix denote the factorization results on . We can use the formula(12) to express:

(12)

Generally, the smaller RRE showed that the reconstructed image is similar to the original images, so it indicates that the performance and robustness of the NMF algorithm is better.

**3.4.2 Average Accuracy (ACC)**

Using the basis vector matrix and the coefficient matrix denote the factorization results on V. Then, to perfume some clustering algorithms. The ACC can help us match the true class label with the predicted clustering label. When the large ACC Value means the better clustering performance. It can be defined by the formula (13):

(13)

**3.4.3 Normalized Mutual Information (NMI)**

NMI mainly evaluates clustering quality mainly through information points. The large NMI value means better clustering performance. It can be defined by the formula (14) :

(14)

**4 Experiment**

In the experiment of Extended YaleB Dataset and ORL Dataset, the proportion of training data was 90%. The least error value was set to 0.0001. When the NMF algorithm satisfied the least error, the algorithm stopped. In addition, we set the number of iterations to be 2000 times for the NMF algorithm. All the results were averaged based on three identical experiments.

**4.1 The Experiment in Extended YaleB Dataset**

Before comparing the robustness between NMF and L1-NMF, we separately added different proportions of gaussian noise and salt and pepper noise to Extended YaleB dataset. The result could be shown by Figure 1:



(a)



(b)



(c)



(d)

Figure 1: (a)Part of Extended YaleB Images by Gaussian Noise; (b)(c)(d) Part of Extended YaleB Images with 2%, 5%, 10%, 15%, 20%, 25%, 30%, 35% of Salt Noise, Pepper Noise, and Salt and Pepper Noise with r = 50% (r denoting the proportion of white 255 pixel).

**4.1.1 The Evaluated Results** **in Gaussian Noise (YaleB)**

After testing the Extended YaleB Dataset, Table 1 shows the results between NMF and -NMF in Gaussian noise. Under Table 1, we can find that the RRE score of -NMF is smaller than the RRE score of NMF, which means the -NMF superior to NMF for image reconstruction based on Gaussian Noise. Although the performance of NMF is higher than the performance of NMF in ACC and NMI scores, it only represents the effect of clustering. As we know, Extended YaleB Dataset has 64 lighting conditions. Dark conditions can have a huge impact on the clustering process, so we believe that the ACC and NMI scores can only be used as the alternative evaluate metrics of robustness. Based on the RRE score, -NMF is more robust than NMF in Gaussian noise.

Table 1: The results of performance between NMF algorithm and -NMF algorithm in Extended YaleB Dataset with Gaussian Noise

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Gaussian | Loss Error | RRE | ACC | NMI |
| 2-NMF | 83694.66 | 0.40547 | 0.16515 | 0.21824 |
| -NMF | 56215.27 | **0.27234** | 0.08989 | 0.09627 |

**4.1.2 The Results of Image Reconstruction in Gaussian Noise (YaleB)**

Figure 2 shows the image reconstruction results in Gaussian noise. Based on Figure 2, we can find that the overall image reconstruction of -NMF is better than the NMF. At the low proportion of Gaussian noise, their performances of image reconstruction are similar, but the NMF has higher reduction accuracy than *L*1-NMF. At the high proportion of Gaussian noise, *L*1-NMF is obviously superior to NMF.



(a)



(b)



(c)

Figure 2: (a)The images were contaminated by Gaussian noise; (b)The images reconstruction of NMF; (c)The images reconstruction of -NMF.

**4.1.3 The Evaluated Results** **in** **Salt and Pepper Noise (YaleB)**

Table 2 shows the performance between NMF and -NMF in the different proportions of 2%, 5%, 10%, 15%, 20%, 25%, 30%, and 35% Salt and Pepper noise. Based on Table 2, we can find that NMF and -NMF exhibit different performances under different proportions and types of Salt and Pepper noise.

In the different proportions of Salt and Pepper noise, the RRE scores of NMF perform better under low proportions Salt and Pepper noise. In contrast, the high proportions of Salt and Pepper noise are more suitable for NMF.

In the different types of Salt and Pepper noise, the RRE scores show the performance of NMF is useful to most of the proportions of Pepper noise (2%, 5%, 10%, 15%, 20%, and 25%). In the Salt noise, the results are opposite because the 1-NMF is better than NMF at most proportional conditions (10%, 15%, 20%, 25%, and 35%). Under the ACC and NMI scores, the overall result of NMF is better than the 1-NMF.

Table 2: The results of performance between NMF algorithm and -NMF algorithm in Extended YaleB Dataset with 2%, 5%, 10%, 15%, 20%, 25%, 30%, 35% of Salt and Pepper Noise.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| NMF | | | | | | | | |
| **Salt and Pepper** | 2% | 5% | 10% | 15% | 20% | 25% | 30% | 35% |
| Loss Error | 39476.17 | 41237.71 | 45287.81 | 50971.93 | 57775.02 | 65459.93 | 73682.97 | 82349.35 |
| RRE | **0.19124** | **0.19978** | **0.21940** | **0.24694** | 0.27990 | 0.31713 | 0.35697 | 0.39895 |
| ACC | 0.22356 | 0.23267 | 0.23198 | 0.22190 | 0.23005 | 0.19884 | 0.18752 | 0.17109 |
| NMI | 0.31246 | 0.33211 | 0.31381 | 0.29558 | 0.30143 | 0.25868 | 0.24502 | 0.22015 |
| -NMF | | | | | | | | |
| **Salt and Pepper** | 2% | 5% | 10% | 15% | 20% | 25% | 30% | 35% |
| Loss Error | 55937.87 | 56577.86 | 57217.15 | 57188.26 | 57624.16 | 57704.96 | 58346.65 | 59204.17 |
| RRE | 0.27099 | 0.27410 | 0.27719 | 0.27705 | **0.27916** | **0.27955** | **0.28266** | **0.28682** |
| ACC | 0.09058 | 0.08451 | 0.08299 | 0.08865 | 0.08506 | 0.08465 | 0.08244 | 0.08230 |
| NMI | 0.10082 | 0.09019 | 0.07497 | 0.08371 | 0.08064 | 0.08112 | 0.07385 | 0.07524 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| NMF | | | | | | | | | | | | | | | |
| **Salt** | 2% | | 5% | | 10% | | | 15% | 20% | | 25% | | 30% | | 35% |
| Loss Error | 40588.08 | | 45863.33 | | 59804.17 | | | 77274.08 | 96351.22 | | 116025.27 | | 135885.54 | | 155627.38 |
| RRE | **0.19663** | | **0.22219** | | 0.28973 | | | 0.37436 | 0.46679 | | 0.56210 | | 0.65832 | | 0.75396 |
| ACC | 0.22922 | | 0.22480 | | 0.22867 | | | 0.20229 | 0.18917 | | 0.17564 | | 0.16363 | | 0.14167 |
| NMI | 0.31562 | | 0.29578 | | 0.30190 | | | 0.27681 | 0.26523 | | 0.24204 | | 0.21045 | | 0.17968 |
| -NMF | | | | | | | | | | | | | | | |
| **Salt** | | 2% | | 5% | | 10% | 15% | | | 20% | 25% | 30% | | 35% | |
| Loss Error | | 55861.59 | | 56074.97 | | 56936.22 | 57732.81 | | | 59610.70 | 61871.24 | 65628.46 | | 72119.41 | |
| RRE | | 0.27062 | | 0.27166 | | **0.27583** | **0.27969** | | | **0.28879** | **0.29974** | **0.31794** | | **0.34939** | |
| ACC | | 0.08824 | | 0.08561 | | 0.08741 | 0.08644 | | | 0.08547 | 0.07650 | 0.07926 | | 0.08188 | |
| NMI | | 0.09667 | | 0.08443 | | 0.08078 | 0.08353 | | | 0.07559 | 0.06165 | 0.06811 | | 0.07551 | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| NMF | | | | | | | | |
| **Pepper** | 2% | 5% | 10% | 15% | 20% | 25% | 30% | 35% |
| Loss Error | 39354.61 | 40266.49 | 44631.29 | 50069.20 | 56631.81 | 63985.88 | 71831.39 | 80256.24 |
| RRE | **0.19065** | **0.19507** | **0.21622** | **0.24256** | **0.27436** | **0.30999** | 0.34799 | 0.38881 |
| ACC | 0.23626 | 0.23101 | 0.23792 | 0.23101 | 0.22742 | 0.21762 | 0.20312 | 0.19774 |
| NMI | 0.33115 | 0.31954 | 0.31492 | 0.29437 | 0.30446 | 0.28770 | 0.28042 | 0.26422 |
| -NMF | | | | | | | | |
| **Pepper** | 2% | 5% | 10% | 15% | 20% | 25% | 30% | 35% |
| Loss Error | 56357.05 | 56766.51 | 57140.70 | 58362.92 | 61047.63 | 65001.18 | 68500.93 | 76518.10 |
| RRE | 0.27302 | 0.27501 | 0.27682 | 0.28274 | 0.29575 | 0.31490 | **0.33186** | **0.37069** |
| ACC | 0.08741 | 0.08754 | 0.08782 | 0.08934 | 0.08796 | 0.09086 | 0.08879 | 0.09072 |
| NMI | 0.09150 | 0.08997 | 0.09312 | 0.09969 | 0.09250 | 0.09002 | 0.09607 | 0.10009 |

**4.1.4 The Results of Image Reconstruction in Salt and Pepper Noise (YaleB)**

Figure 3 shows the image reconstruction results in Salt and Pepper noise. Based on Figure 3, the result of NMF is close to the original image than the result of NMF with a low proportion of noise. When the proportion of noise is high, the result of -NMF is clearer than the result of NMF.

  

(a) (b) (c)

Figure 3: (a)The images were contaminated by 5% and 30% Salt and Pepper noise; (b)The images reconstruction of NMF; (c)The images reconstruction of -NMF.

Figure 4 shows the image reconstruction results in Salt noise and Pepper noise. In Figure 4, the image reconstruction of -NMF is better than the result of NMF in Salt noise. Under the Pepper Noise, although -NMF effectively restores the image, the feature of the face has been partially changed. Compared with -NMF, the image reconstruction of NMF is better.

  

(a1) (a2) (a3)

  

(b1) (b2) (b3)

Figure 4 : (a1) The images were contaminated by 20% of Salt noise; (a2) The NMF images reconstruction for Salt Noise; (a3) The -NMF images reconstruction for Salt Noise; (b1)The images were contaminated by 20% of Pepper noise; (b2) The NMF images reconstruction for Pepper Noise; (b3) The -NMF images reconstruction for Pepper Noise.

**4.2 The Experiment in ORL Dataset**

In the ORL dataset, the procedure is similar to the experiment of the Extended YaleB Dataset. The ORL dataset was polluted by different proportions of Gaussian Noise and Salt and Pepper noise. Figure 5 shows the result:



(a)



(b)



(c)



(d)

Figure 5: (a)Part of Extended YaleB Images by Gaussian Noise; (b)(c)(d)Part of Extended YaleB Images with 2%, 5%, 10%, 15%, 20%, 25%, 30%, 35% of Salt Noise, Pepper Noise, and Salt and Pepper Noise.

**4.2.1 The Evaluated Results** **in Gaussian Noise** **(ORL)**

When we add Gaussian noise to the ORL dataset, as shown in Table 3, we find that the RRE score of NMF (0.158) is lower than RRE score of L1-NMF (0.296). In addition, the accuracy and NMI of NMF is higher than L1-NMF, which means that NMF performs better when we add Gaussian noise to the picture. Therefore, NMF is more robust in this situation.

Table 3: The results of performance between 2-NMF algorithm and -NMF algorithm in ORL Dataset with Gaussian Noise

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Gaussian** | Loss Error | RRE | ACC | NMI |
| 2-NMF | 39557.74 | **0.15815** | 0.7225 | 0.83623 |
| -NMF | 36923.73 | 0.29616 | 0.19 | 0.34723 |

**4.2.2 The Evaluated Results** **of** **in Salt and Pepper Noise (ORL)**

Table 4 shows the performance between NMF and -NMF in the different proportions of 2%, 5%, 10%, 15%, 20%, 25%, 30%, and 35% Salt and Pepper noise. Based on Table 4, we can find that NMF and -NMF exhibit different performances under different proportions and types of Salt and Pepper noise.

When we add the mixed noise of salt and pepper, in most cases, the RRE of NMF is smaller, which shows that NMF is resistant to salt and pepper when the noise ratio is small (2%-30%) The noise capability is stronger. At a ratio of 35%, L1-NMF performs better. It is speculated that L1-NMF performs better in a larger proportion of noise.

When adding salt noise alone, NMF performs better at a smaller noise ratio, which is 2%-15%. In L1-NMF, the RRE is lower, and the accuracy is higher when the noise ratio is larger, which is between 20%-35%. It shows that for pure salt noise, NMF has better robustness under small-range noise interference, while L1-NMF has better robustness under large-scale noise interference.

When pepper noise is added alone, the experimental results are similar to the previous ones. NMF has lower RRE in the noise ratio of 2%-20%, while L1-NMF has lower RRE in the noise range of 25%-35%. Combining these three situations, we conclude that for the two types of noise, salt, and pepper, we need to illustrate the robustness of L1-NMF and NMF according to the noise ratio. L2 is often achieved when the noise ratio is small. The better indicators (lower RRE scores, higher ACC, and NMI scores) indicate that it has better robustness and stronger anti-interference ability in this case. And L1-NMF tends to show better robustness when the noise ratio is high.

Table 4: The results of performance between NMF algorithm and -NMF algorithm in ORL Dataset with 2%, 5%, 10%, 15%, 20%, 25%, 30%, 35% of Salt and Pepper

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| NMF | | | | | | | | |
| **Salt and Pepper** | 2% | 5% | 10% | 15% | 20% | 25% | 30% | 35% |
| Loss Error | 40849.33 | 43731.98 | 48748.41 | 54157.87 | 59769.13 | 65270.04 | 71191.08 | 76389.61 |
| RRE | **0.16332** | **0.17484** | **0.19490** | **0.21652** | **0.23896** | **0.26095** | **0.28463** | 0.30541 |
| ACC | 0.69 | 0.76 | 0.72 | 0.6725 | 0.6375 | 0.5975 | 0.54 | 0.515 |
| NMI | 0.82725 | 0.86835 | 0.85299 | 0.81581 | 0.79830 | 0.74710 | 0.71498 | 0.66554 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| -NMF | | | | | | | | |
| **Salt and Pepper** | 2% | 5% | 10% | 15% | 20% | 25% | 30% | 35% |
| Loss Error | 37000.40 | 36972.30 | 36936.59 | 36986.36 | 36958.63 | 37016.98 | 37059.41 | 37097.78 |
| RRE | 0.29677 | 0.29655 | 0.29626 | 0.29666 | 0.29644 | 0.29690 | 0.29725 | 0.29755 |
| ACC | 0.19 | 0.1975 | 0.1975 | 0.1875 | 0.195 | 0.175 | 0.175 | 0.1875 |
| NMI | 0.36324 | 0.34267 | 0.37434 | 0.35088 | 0.34537 | 0.33299 | 0.35420 | 0.33810 |
|  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| NMF | | | | | | | | |
| **Salt** | 2% | 5% | 10% | 15% | 20% | 25% | 30% | 35% |
| Loss Error | 41575.08 | 46600.97 | 57743.01 | 70581.76 | 84520.04 | 98536.34 | 112111.32 | 125709.89 |
| RRE | **0.16622** | **0.18631** | **0.23086** | **0.28219** | 0.33792 | 0.39396 | 0.44823 | 0.50260 |
| ACC | 0.715 | 0.6975 | 0.73 | 0.6625 | 0.5525 | 0.5225 | 0.4675 | 0.465 |
| NMI | 0.83567 | 0.82706 | 0.82652 | 0.80184 | 0.72751 | 0.72785 | 0.67720 | 0.63602 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| -NMF | | | | | | | | |
| **Salt** | 2% | 5% | 10% | 15% | 20% | 25% | 30% | 35% |
| Loss Error | 37009.17 | 37114.33 | 37539.02 | 38206.89 | 39343.72 | 40988.04 | 43565.98 | 47159.48 |
| RRE | 0.29684 | 0.29769 | 0.30109 | 0.30645 | **0.31557** | **0.32876** | **0.34943** | **0.37826** |
| ACC | 0.1975 | 0.1875 | 0.1875 | 0.2 | 0.19 | 0.1825 | 0.1975 | 0.1875 |
| NMI | 0.35322 | 0.35907 | 0.33497 | 0.37279 | 0.35343 | 0.35932 | 0.36052 | 0.36635 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| NMF | | | | | | | | |
| **Pepper** | 2% | 5% | 10% | 15% | 20% | 25% | 30% | 35% |
| Loss Error | 40866.66 | 44371.37 | 52036.75 | 61350.11 | 71599.48 | 82364.45 | 93311.49 | 103943.56 |
| RRE | **0.16338** | **0.17740** | **0.20804** | **0.24528** | **0.28626** | 0.32930 | 0.37307 | 0.41557 |
| ACC | 0.7275 | 0.735 | 0.66 | 0.6775 | 0.63 | 0.61 | 0.5575 | 0.4525 |
| NMI | 0.85792 | 0.84280 | 0.81784 | 0.81333 | 0.79200 | 0.77402 | 0.75195 | 0.66271 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| -NMF | | | | | | | | |
| **Pepper** | 2% | 5% | 10% | 15% | 20% | 25% | 30% | 35% |
| Loss Error | 37007.33 | 37044.38 | 37296.82 | 37980.70 | 39185.98 | 41008.79 | 43720.65 | 47808.01 |
| RRE | 0.29683 | 0.29712 | 0.29915 | 0.30463 | 0.31430 | **0.32892** | **0.35067** | **0.38346** |
| ACC | 0.1975 | 0.17 | 0.2 | 0.185 | 0.1925 | 0.2075 | 0.205 | 0.2 |
| NMI | 0.34952 | 0.340814 | 0.36267 | 0.36526 | 0.37798 | 0.39130 | 0.36880 | 0.37392 |

**4.3 Determine k-features**

K value represents the abstract features of training dictionary. It has tremendous impact on the performance of NMF and L1-norm algorithm. This section chooses different k values (k = [10, 20, 30, 40, 50, 60, 70, 80]) under Gaussian noise and Salt Pepper 5% noise. For each test, it is conducted based on ORL dataset with 2000 times iteration.

**4.3.1 Gaussian Noise**

![图片包含 图表

描述已自动生成]() ![图表

描述已自动生成]()

（a） (b)

Figure 6: (a): Reconstruct error with L1-norm NMF on Gaussian noise. (b): Reconstruct error with NMF on Gaussian noise.

**4.3.2 Salt and Pepper Noise**

![图片包含 图表

描述已自动生成]() ![图表

描述已自动生成]()

（a） (b)

Figure 7: (a): Reconstruct error with L1-norm NMF on 5% Salt and Pepper noise. (b): Reconstruct error with NMF on 5% Salt and Pepper noise.

Experiment results on Gaussian noise and Salt Pepper noise address that, generally, a larger k value signifying a smaller RRE. While L1-norm NMF and NMF are generating the features of image, they are sensitive on the k value.

**5 Conclusion**

For each dataset, we randomly exemplify 90% images, testing three evaluation metrics on NMF and *L*1-norm NMF algorithms with 3 times. The experiment on Extended YaleB dataset and ORL dataset show that, giving an invariant k (number of features in a dictionary), the value of Relative Reconstruct Error (RRE), Average Accuracy (ACC), and NMI (Normalized Mutual Information) are decreasing with the proportion of noise increasing. This indicates that the reconstruction result of an NMF algorithm is under-performing. The experiment defaults to the value of k as the number of groups of photos. In this case, NMF performs better than *L*1-norm NMF.

Determining the value of k is indispensable in NMF algorithms, a larger k value signifying a smaller RRE. *L*1-norm NMF and NMF are both sensitive on the k value. While *L*1-norm NMF is generating the features of image, it is inclined to distribute bigger weight to similar lighting parts. As a consequence, the dictionary features of human face and reconstruction images are blurred and rough, especially in ORL dataset which has less training images.

As for future research, we plan to explore an NMF algorithm with higher robustness and better performance, since NMF is widely used in various areas, including computer vision, speech analysis, text mining, and etc. Highly robust Algorithms can perform better in these areas.

**6 Indicate the contribution**

Shang: Abstract, Instruction, Related work, Literature review, indicate contributions and future works, analyze the experiment results of ORL dataset, code implementation, entire report layout.

Zhong: Write codes and implementation, write determine K features part and conclusion in the report.

Huang: Methods review, analyze experiment results of YaleB dataset, made related figures and tables, code implementation.

**References**

[1] Kong, D., Ding, C., & Huang, H. (2011, October 01). Robust nonnegative matrix factorization using L21-norm. Retrieved October 22, 2020, from https://dl.acm.org/doi/10.1145/2063576.2063676

[2] Lee, D. D., & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. Nature, 401(6755), 788-791. doi:10.1038/44565

[3] Guan, N., Liu, T., Zhang, Y., Tao, D., & Davis, L. S. (2019). Truncated Cauchy Non-Negative Matrix Factorization. IEEE Transactions on Pattern Analysis and Machine Intelligence, 41(1), 246-259. doi:10.1109/tpami.2017.2777841

[4] Buciu, Ioan. (2008). Nonnegative Matrix Factorization, A New Tool for Feature Extraction: Theory and Applications. International Journal of Computers, Communications and Control. 3.

[5] Hamza, A., & Brady, D. (2006). Reconstruction of reflectance spectra using robust nonnegative matrix factorization. IEEE Transactions on Signal Processing, 54(9), 3637-3642. doi:10.1109/tsp.2006.879282

[6] Yang, Z., Hu, Y., Liang, N., & Lv, J. (2019). Nonnegative Matrix Factorization with Fixed L2-Norm Constraint. Circuits, Systems, and Signal Processing, 38(7), 3211-3226. doi:10.1007/s00034-018-1012-4

[7] N. Z. Weixiang Liu and Q. You. Nonnegative matrix factorization and its applications in pattern recognition. Chinese Science Bulletin, 51(1):7–18, 2006.

[8] [Inderjit S. Dhillon](https://en.wikipedia.org/w/index.php?title=Inderjit_S._Dhillon&action=edit&redlink=1); [Suvrit Sra](https://en.wikipedia.org/w/index.php?title=Suvrit_Sra&action=edit&redlink=1) (2005). [Generalized Nonnegative Matrix Approximations with Bregman Divergences](https://papers.nips.cc/paper/2757-generalized-nonnegative-matrix-approximations-with-bregman-divergences.pdf) (PDF). [NIPS](https://en.wikipedia.org/wiki/Conference_on_Neural_Information_Processing_Systems).

[9] Gillis, N. (2012, April 11). Sparse and Unique Nonnegative Matrix Factorization Through Data Preprocessing. Retrieved October 21, 2020, from https://arxiv.org/abs/1204.2436

[10] Barbu, T. (2013). Variational Image Denoising Approach with Diffusion Porous Media Flow. Abstract and Applied Analysis, 2013, 1-8. doi:10.1155/2013/856876

[11] Dwivedy, P. (2017). Performance Assessment of Several Filters for Removing Salt and Pepper Noise, Gaussian Noise, Rayleigh Noise and Uniform Noise. International Journal of Engineering and Applied Computer Science, 02(06), 176-180. doi:10.24032/ijeacs/0206/01

[12] ORL Images (examples.orl\_images)¶. (n.d.). Retrieved October 21, 2020, from http://ai.stanford.edu/~marinka/nimfa/nimfa.examples.orl\_images.html

[13] The Extended Yale Face Database B. (n.d.). Retrieved October 21, 2020, from http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html

[14] Hoyer, P. O. (2004, August 04). Non-negative Matrix Factorization with Sparseness Constraints. Retrieved October 22, 2020, from https://jmlr.csail.mit.edu/papers/volume5/hoyer04a/hoyer04a.pdf

[15] Lee, D. D., &amp; Seung, S. (n.d.). Algorithms for Non-negative Matrix Factorization. Retrieved October 22, 2020, from http://www.cs.cmu.edu/~11755/lectures/Lee\_Seung\_NMF.pdf

[16] Cattin, P. (2016, April 19). Image Restoration. Retrieved October 22, 2020, from https://miac.unibas.ch/SIP/06-Restoration.html

[17] L. Zhang, Z. Chen, M. Zheng, and X. He, “Robust Non-negative Matrix Factorization,” Frontiers of Electrical and Electronic Engineer- ing in China, vol. 6, no. 2, pp. 192-200, Feb. 2011.

**Appendix：instructions of code running**

To conduct the experiment, go to main.py and execute this file. There are four python files in this project, where main.py encompassing variants such as training data root, noise type, noise proportion, algorithm type and iterations. Core.py is a functional unit controlling the core processes including adding noise, generating noise images, and reconstructing images via utilizing NMF algorithm. The implementation of NMF algorithm is within the nmf.py. Utils.py includes evaluation function and plot function.