Assignment 1

Tutors: Siyu Xu
Group members: Kaiyue Wang (ID: 530402325, UniKey: kwan5221), Peng Huang (ID: 530203700, UniKey: phua0038), Shoudi Huang (ID: 500478204, Unikey: shua9875)

Abstract

In this study, We evaluated the robustness and performance of L_2-NMF, Cauchy-NMF, and Huber-NMF algorithms on ORL and Extended YaleB (CroppedYaleB) datasets contaminated with Gaussian, salt and pepper, and block occlusion noises. Our results indicate that Huber-NMF is exceptionally robust across most noise types. Both Cauchy-NMF and L_2-NMF offer similar performance under most conditions, but L_2-NMF proves superior in handling Gaussian Noise. These findings emphasize the significance of algorithm selection based on specific noise conditions, underscoring its crucial role in image processing applications.

1 Introduction

Non-negative matrix factorization (NMF) [7], as one of the dictionary learning techniques, essentially decomposes a non-negative matrix into the product of two non-negative matrices. It is widely used to extract meaningful features.

In recent years, many scholars have proposed various NMF algorithms, such as $L_{2,1}$-NMF [5], Cauchy-NMF [8], CIM-NMF, Huber-NMF [3], etc. Some of the new NMFs claim to have improved the performance of traditional NMF algorithms.

The evolution of machine learning technology has significantly increased the demand for training datasets. However, during the preparation of these datasets, noise inevitably arises due to natural or human related factors. This noise, especially in datasets with a large number of features (high-dimensional) can significantly slow down the computational efficiency and increase the risk of overfitting.

Therefore, our primary objective is to evaluate the performance of different NMF algorithms on datasets contaminated by various types of noise.

This report will experimentally verify the performance of Cauchy-NMF, Huber-NMF, and L_2-NMF when the dataset is contaminated by large magnitude noise or corruption. We introduced three types of noise contamination in both the ORL dataset[1] and the Extended YaleB (CroppedYaleB) dataset[2] and fed them into the algorithms for training. We evaluated using three metrics with the aim to compare the performance and robustness of different NMF algorithms.

By studying the performance on different noises it is meaningful to understand the capability of NMF for further application in the machine learning field.

[1] https://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html
[2] http://vision.ucsd.edu/ iskwak/ExtYaleDatabase/ExtYaleB.html

2 Previous work

Non-negative matrix factorization is a technique for decomposing a non-negative matrix X into two lower-rank non-negative matrices, D and R, to approximate X as accurately as possible. It is shown in (Eq. 1).

$$X \approx DR \tag{1}$$

The standard form, known as Frobenius norm-based NMF or L_2-NMF, minimizes the squared Euclidean distance (Eq. 2) between X and DR [6]. While computationally efficient, L_2-NMF is sensitive to outliers. When an outlier is present, the term $(a_i - b_i)^2$ can become disproportionately large since the squaring operation magnifies the impact of this individual term in the summation. To address this limitation, various adaptations have been proposed.

$$\text{Euclidean Distance} = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2} \tag{2}$$

For instance, L_1-NMF was introduced to minimize the L_1 norm (Eq. 3) of the residual matrix $X - DR$, thereby enhancing robustness to outliers at the expense of increased computational complexity due to the non-smooth nature of the L_1 norm based loss function. Then, Manhattan-NMF is introduced to reduce the computational complexity, which solves L_1-NMF by approximating with a smooth one using Nesterov's method [4].

$$\|x\|_1 = \sum_{i=1}^{n}\left(\sum_{j=1}^{m}|x_{ij}|\right) \tag{3}$$

Another adaptation is $L_{2,1}$-NMF, which employs the $L_{2,1}$ norm (Eq. 4) in its objective function to mitigate the influence of large-magnitude noise [5]. Compared to the L_1 norm's robustness to outliers but computational inefficiency and the L_2 norm's sensitivity to outliers, the $L_{2,1}$ norm offers a balanced approach, providing robustness to column-wise outliers while maintaining computational efficiency.

$$\|x\|_{2,1} = \sum_{i=1}^{n}\left(\sum_{j=1}^{m}|x_{ij}|^2\right)^{1/2} \tag{4}$$

L_1-NMF, Manhattan-NMF and $L_{2,1}$-NMF, [4] and [5] share a common drawback, they all have relatively poor performance for grossly corrupted data. Therefore, Cauchy-NMF is introduced, which employs the Cauchy loss function (Eq. 10) in its optimization procedure, providing an even higher level of robustness to extreme outliers and heavy-tailed noise at the cost of increased optimization complexity due to the non-convex nature of the loss function [8].

In addition, other variants have been proposed to enhance robustness in more specialized contexts. The Correntropy-Induced Metric NMF (CIM-NMF) was introduced to incorporate a non-linear similarity measure, correntropy (Eq. 5), into the objective function. This allows CIM-NMF to perform well for non-Gaussian data distributions or non-linear feature relationships [3]. Due to the connection between the correntropy-induced metric and the robust M-estimators shown in CIM-NMF, Huber-NMF is introduced. It measures the approximation errors by using the Huber function (Eq. 13). Huber-NMF was formulated to balance the L_1 and L_2 norms, offering partial robustness to outliers while maintaining computational efficiency [3].

$$g(e,\sigma) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-e^2/2\sigma^2\right) \tag{5}$$

This section collectively underscores the ongoing efforts to enhance the robustness of NMF algorithms across a diverse array of noise conditions.

3 Methods

3.1 Pre-processing

In the present study, image data are subjected to a normalization process utilizing Min-Max Scaling, as shown in Eq. 6. This normalization technique generally entails the identification of the minimum $\min(x)$ and maximum $\max(x)$ values within the dataset. Followed by a rescaling operation predicated on these extremal values.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{6}$$

For the 8-bit grayscale images employed in this study, the pixel intensity values are confined to a discrete range between 0 and 255. In this specific context, the Min-Max Scaling operation simplifies to a division of each pixel value by 255, effectively scaling the pixel intensities to a $[0, 1]$ range.

This normalization can improve numerical stability, accelerate algorithmic convergence, enhance model robustness, and lead to more efficient computational resource utilization.

3.2 NMF algorithm

This section outlines the NMF algorithms employed to investigate their robustness across various noise conditions.

3.2.1 L_2-NMF

The traditional NMF algorithm is a matrix decomposition technique that deals with noise obeys a Gaussian distribution. The L_2 norm generates a smooth and continuous cost surface, which might be beneficial for convergence to a local minimum in an optimization problem [6]. It is meaningful to find global solutions in the complex question. The objective of L_2-NMF is to find a reasonable representation of the original data in lower dimensions. The objective function as shown in Eq. 7.

$$\mathbf{D}, \mathbf{R} = \min_{D,R} \|X - DR\|_F \tag{7}$$

where $||\cdot||_F$ denotes the Frobenius (L_2) norm which is the square root of the sum of the squares of all the matrix elements and for an $m \times n$ metric Q with elements q_{ij} it is defined by Eq. 8.

$$\|Q\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} q_{ij}^2} \tag{8}$$

To find the optimal D and R that best approximate the given matrix X and at the same time the Non-negativity and convergence must be guaranteed, we decide to use the Multiplicative Update Rule which has already been proven effective in practice to update the initial D and R which are random given of the appropriate size. The rule can be represented by Eq. 9.

$$\begin{aligned} D_{ij} &= D_{ij} \frac{(XR^T)_{ij}}{(DRR^T)_{ij}} \\ R_{ij} &= R_{ij} \frac{(D^T X)_{ij}}{(D^T DR)_{ij}} \end{aligned} \tag{9}$$

3.2.2 Cauchy-NMF

Cauchy NMF (Cauchy-NMF) is proposed to offer heightened robustness against extreme outliers and heavy-tailed noise distributions [8]. The Cauchy-NMF incorporates the Cauchy loss into its objective function. The sub-linear growth of the Cauchy loss function is a critical attribute that distinguishes it from L_1 and L_2 norms. While L_1 and L_2 norms penalize errors in a linear or quadratic manner, respectively, the logarithmic nature of the Cauchy loss ensures that even large

deviations contribute only moderately to the overall loss. Mathematically, the objective function F for Cauchy-NMF can be expressed by Eq. 10

$$\min_{D,R} \sum_{i=1}^{N} \sum_{j=1}^{M} \ln \left(\left\| X_{i,j} - \sum_{k=1}^{K} D_{ik} R_{kj} \right\|^2 + \gamma^2 \right) \tag{10}$$

γ is a scale parameter that modulates the sensitivity of the loss function to deviations between X and DR. The above equation can be approximates to $F(D, R) = 2 \ln (\|X - DR\|)$, then Cauchy-NMF can be viewed as the $L_{2,1}$-NMF in the logarithmic space [8]. Therefore, the Cauchy-NMF has similar multiplicative update rules (Eq. 11) as the $L_{2,1}$-NMF and it is presented below.

$$\begin{aligned} \Delta_{ii} &= \frac{1}{\sum_{j=1}^{p} (X-DR)_{ij}^2 + \gamma^2} \\ D_{jk} &= D_{jk} \frac{\left(X \Delta R^T\right)_{jk}}{\left(DR \Delta R^T\right)_{jk}} \\ R_{ki} &= R_{ki} \frac{\left(D^T X \Delta\right)_{ki}}{\left(D^T DR \Delta\right)_{ki}} \end{aligned} \tag{11}$$

3.2.3 Huber-NMF

Similar to CIM-NMF, the Huber NMF (Huber-NMF) incorporates the Huber loss into its objective function (Eq. 12) to enhance robustness to outliers. It was inspired by the connection between the correntropy-induced metric and the robust M-estimators showed in CIM-NMF [3]. The Huber loss function serves as a measure of approximation error between the original matrix X and its low-rank approximation DR. The Huber-NMF aims to be robust to outliers to a certain extent while still being smooth enough to be efficiently optimized.

Mathematically, the objective function F for Huber-NMF can be expressed by Eq. 12.

$$\min_{D,R} \sum_{i=1}^{N} \sum_{j=1}^{M} \ell_{\text{Huber}} \left(X_{ij} - \sum_{k=1}^{K} D_{ik} R_{jk} \right) \tag{12}$$

Here, ℓ_{Huber} is the Huber loss function is defined by Eq. 13.

$$\ell_{\text{huber}}(e) = \begin{cases} e^2 & \text{if } |e| \leq c \\ 2c|e| - c^2 & \text{if } |e| \geq c \end{cases}, \tag{13}$$

Where $c = \text{median}(\left|(X_{ij} - \sum_{k=1}^{K} D_{ik} R_{jk}\right|)$ is the cutoff parameter to balance between L_2-norm and L_1-norm.

The multiplicative update rules for Hubber-NMF are present by Eq. 14. Where $\circ$ is the Hadamard (elementwise) product.

$$\begin{aligned} E_{ij} &= X_{ij} - \textstyle\sum_{k=1}^{K} D_{ik} R_{jk} \\ c &= \text{median}(|E_{ij}|) \\ W_{ij} &= \begin{cases} 1 & \text{if } |E_{ij}| \leq c \\ \frac{c}{|E_{ij}|} & \text{otherwise} \end{cases} \\ D_{ik} &= D_{ik} \frac{(W \circ XR)_{ik}}{(W \circ (DR^T) R)_{ik}} \\ R_{jk} &= R_{jk} \frac{\left((W \circ X)^T D\right)_{jk}}{\left((W \circ (DR^T))^T D\right)_{jk}} \end{aligned} \tag{14}$$

3.3 Noise introduction

Noise is generally the unwanted component in an image. Data acquisition, transmission, and storage processes can be subject to contamination, resulting in noise [1]. We have designed three methods for generating corresponding noise to simulate the contamination that occurs at different stages in the creation of a photograph. The contaminated images with different noises is showed in Figure 1.

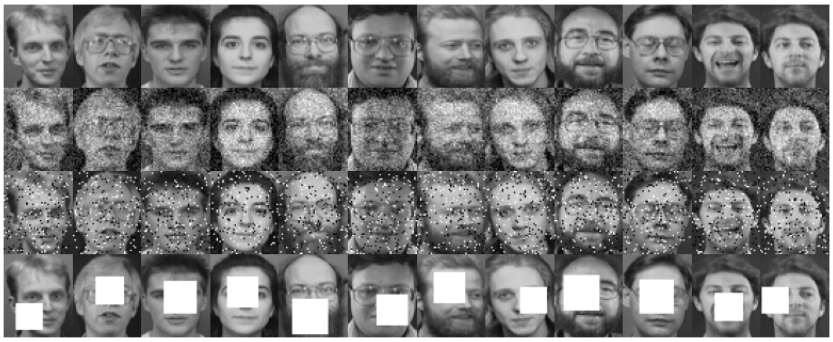

Figure 1: Sample of Images with Different Noises

3.3.1 Images with Gaussian noise

The noise during the image sensor sampling process primarily originates from environmental conditions during acquisition, light levels (low light conditions requiring high gain amplification), sensor temperature (higher temperature implies more amplification noise), and the interface circuitry connected to the sensor. The most frequently used practical model for image noise is Gaussian noise [2].

We designed a method to simulate Gaussian noise. First we create a noise matrix with the same shape as the image, and randomly set the value of each element in the matrix based on Gaussian distribution. The probability density function (pdf) for the Gaussian distribution is given by Eq. 15. Then, add the grayscale value of each pixel of the noise matrix and the original image, and ensure that the value of each pixel is less than the upper limit, and output the contaminated image, as depicted in the second row of Figure 1.

$$f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{15}$$

3.3.2 Images with salt and pepper noise

Another common type of image noise is salt and pepper noise. The effect is similar to sprinkling white and black dots. They are often caused by bit errors during analog-to-digital conversion or transmission processes [1].

We have designed a method to simulate salt and pepper noise. According to the probability specified in the experiment, we select a corresponding number of random pixel positions and set the pixels at those positions to either pure black or pure white, as depicted in the third row of Figure 1.

3.3.3 Images with a block occlusion

Block occlusion is a widely used noise added to datasets to evaluate algorithm performance. We have designed a method that selects an offset coordinate by randomly choosing a row and column number within an image. Using this offset coordinate, we draw a square with a size randomly ranging from 30% to 50% of the image's height. Finally, all the pixels within this square area are set to pure white, as depicted in the fourth row of Figure 1.

4 Experiments and discussions

In previous work, some experiments have shown that specific NMF algorithms exhibit variations in performance under certain types of noise. However, due to the differences in datasets, types of noise, levels of contamination, and types of algorithms used across various studies, it is hard to determine which algorithm is the most robust.

Therefore, we will input several contaminated image data subsets into L_2-NMF, Cauchy-NMF, and Huber-NMF for training, obtaining their respective D and R. Then, we will use three evaluation methods to conduct a side-by-side comparison and analyze their robustness.

4.1 Experimental design

We designed an experimental scheme divided into four steps, as shown in Figure 2. Firstly, we load the data and preprocess it by normalizing and converting each image to a dataset. This is done by reading the image file as a 2D matrix, then converting it into a 1D column vector, and performing normalization using Min-Max Scaling (Section 3.1). After that, we stack all the image column vectors together to form a matrix for the dataset.

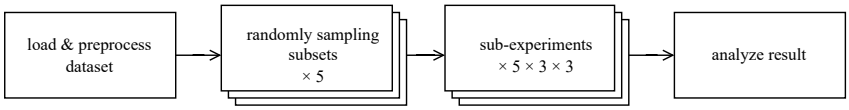

Figure 2: Experiment Procedure

Secondly, we randomly sample 90% training data from the whole dataset for five times and get five sampling subset. The benefit of conducting repeated experiments is that it enhances the reliability of the experimental results and it also allows us to observe the impact on the algorithm training results caused by variations in the training set.

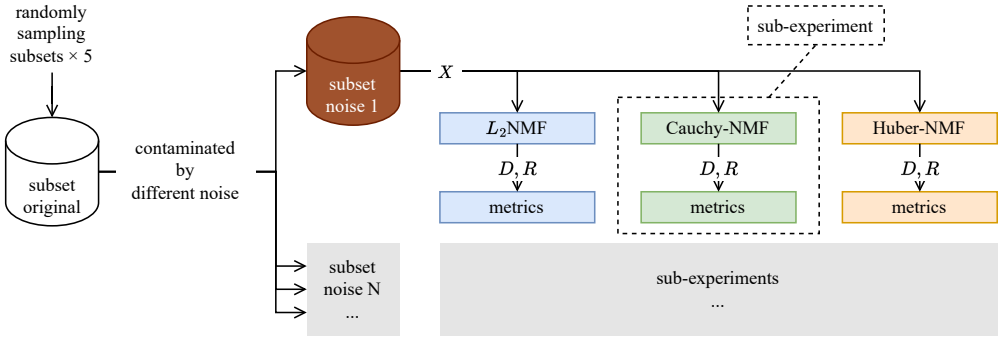

Figure 3: Run sub-experiments

For the third step, we will conduct several sub-experiments using each subset. Each sampling subset is contaminated with three types of noise, generating three noise subsets (subset noise $1 \cdots$ subset noise 3). We will input these noise subsets as X into different NMFs, considering both coverage conditions and maximum iteration steps, and then obtain outputs D and R. Subsequently, we obtain the X_{pred} by multiplying metrices D and R, and evaluate the output of each sub-experiment by calculating the Root Means Square Errors (RMSE) as given in (Eq. 16), Average Accuracy (Acc) as given in (Eq. 17), and Normalized Mutual Information (NMI) as given in (Eq. 18) with the original subset $\hat{X}$ and Y. Where $I(\cdot,\cdot)$ is mutual information and $H(\cdot)$ is entropy.

$$\text{RMSE} = \sqrt{\frac{1}{N}\|\hat{X} - DR\|_F^2} \tag{16}$$

$$\text{Acc}(Y, Y_{\text{pred}}) = \frac{1}{n}\sum_{i=1}^{n}\{Y_{\text{pred}}(i) == Y(i)\} \tag{17}$$

$$\text{NMI}(Y, Y_{\text{pred}}) = \frac{2 \cdot I(Y, Y_{\text{pred}})}{H(Y) + H(Y_{\text{pred}})} \tag{18}$$

Finally, we will compute the results of each sub-experiment (noise and NMF combination) across different subsets, obtaining the mean and standard deviation of RMSE, Acc, and NMI, as shown in Figure 4. This will help us to analyze the performance of different NMF algorithms.

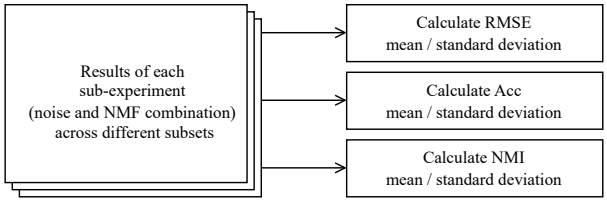

Figure 4: Calculate results

We will execute the experimental scheme on both the ORL and CroppedYaleB datasets for a rigorous performance evaluation.

- The ORL dataset contains 400 images of 40 distinct subjects each image is cropped and resized to 92×112 pixels. Considering the computation cost, we reduce the resolution of the input image by a factor of 2 to 46×56 pixels.
- For the extended YaleB dataset, it contains 2414 images of 38 subjects under 9 poses and 64 illumination conditions and all images are resized to 168×192 pixels. Considering the computation cost, we reduce the resolution of the input image by a factor of 4 to 42×48 pixels.

4.2 Experimental results

This section presents the empirical results of applying the NMF algorithms detailed in Section 3.2 to the ORL and CroppedYaleB datasets under various noise conditions described in Section 3.3. Figure 5 showcases sample reconstructions.

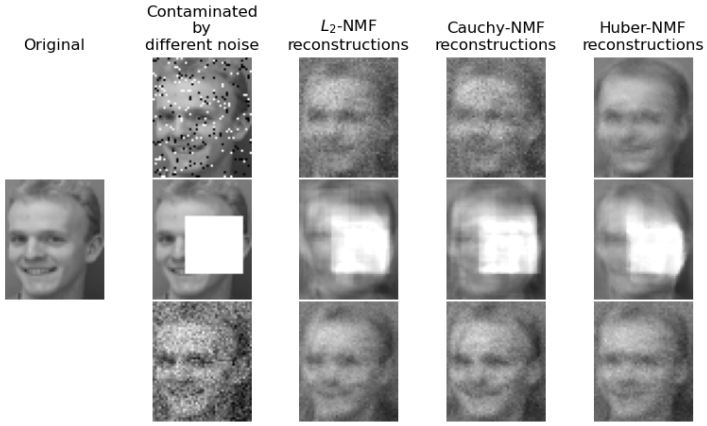

Figure 5: Samples of reconstructed images for different noises

4.2.1 Results on ORL

As shown in Figure 6, the comparative analysis under the Root Mean Square Error (RMSE) metric reveals that the performance of Cauchy-NMF and L_2-NMF exhibit comparable efficacy across three distinct noise conditions in all five sub-experiments conducted on the ORL dataset.

However, it is noteworthy that Cauchy-NMF is marginally superior to that of L_2-NMF in Block Occlusion Noise. Conversely, Huber-NMF demonstrates a marked improvement over both Cauchy-NMF and L_2-NMF, when the images are subjected to salt and pepper noise or Block Occlusion Noise. This superiority is evident across multiple evaluation metrics, including RMSE, Accuracy

(Acc), and Normalized Mutual Information (NMI). However, its performance is relatively poor in the presence of Gaussian Noise.

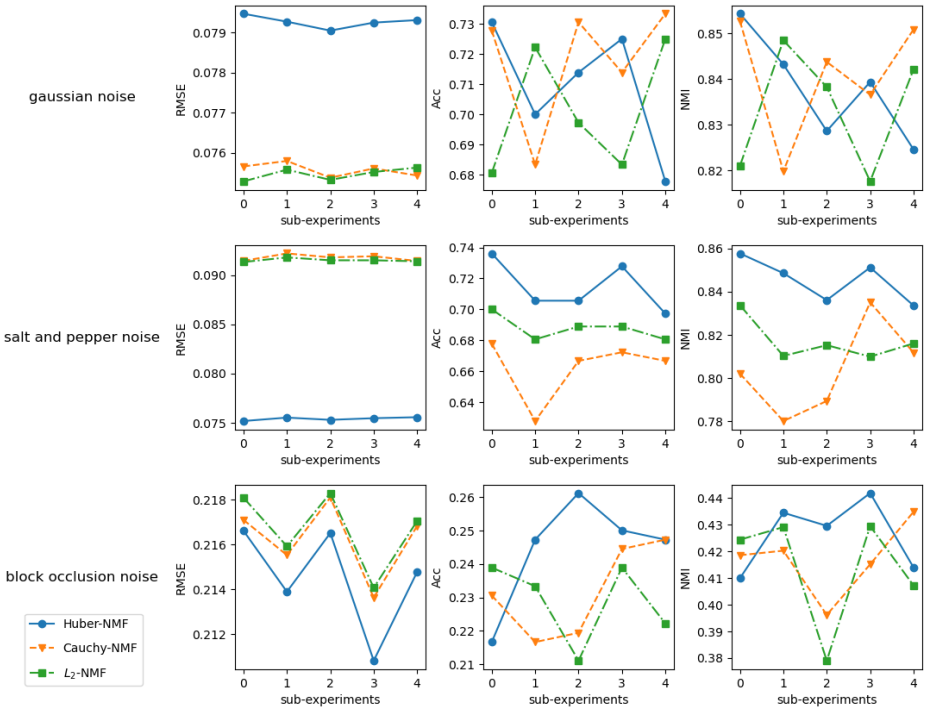

Figure 6: OLR experiments results

It is also worth noting that despite similar RMSE values for Cauchy-NMF and L_2-NMF under Salt and pepper noise, Cauchy-NMF exhibits a lower accuracy, as illustrated in Figure 6. Table 1 provides a comprehensive summary of the mean and standard deviation of the evaluation metrics for the five sub-experiments, with the best scores for each metric highlighted in blue.

Table 1: OLR experiments results

Noise	Metrics		L_2-NMF	Cauchy-NMF	Huber-NMF
gaussian noise	RMSE	mean	**0.075463**	0.075570	0.079271
		std	0.000138	0.000153	**0.000135**
	Acc	mean	0.701667	**0.717778**	0.709444
		std	0.018807	**0.018476**	0.018970
	NMI	mean	0.833511	**0.840736**	0.838022
		std	0.012058	0.011885	**0.010581**
salt and pepper noise	RMSE	mean	0.091500	0.091753	**0.075391**
		std	0.000158	0.000291	**0.000147**
	Acc	mean	0.687778	0.662222	**0.714444**
		std	**0.007158**	0.017708	0.014845
	NMI	mean	0.816949	0.803586	**0.845368**
		std	**0.008716**	0.019058	0.009189
block occlusion noise	RMSE	mean	0.216677	0.216230	**0.214522**
		std	0.001551	**0.001534**	0.002115
	Acc	mean	0.228889	0.231667	**0.244444**
		std	**0.010773**	0.012497	0.014803
	NMI	mean	0.413824	0.416999	**0.425986**
		std	0.019261	0.012436	**0.012189**

4.2.2 Results on CroppedYaleB

As shown in Figure 7, the experimental results obtained under the Root Mean Square Error (RMSE) metric for the CroppedYaleB dataset exhibit similar trends to those observed in the ORL dataset.

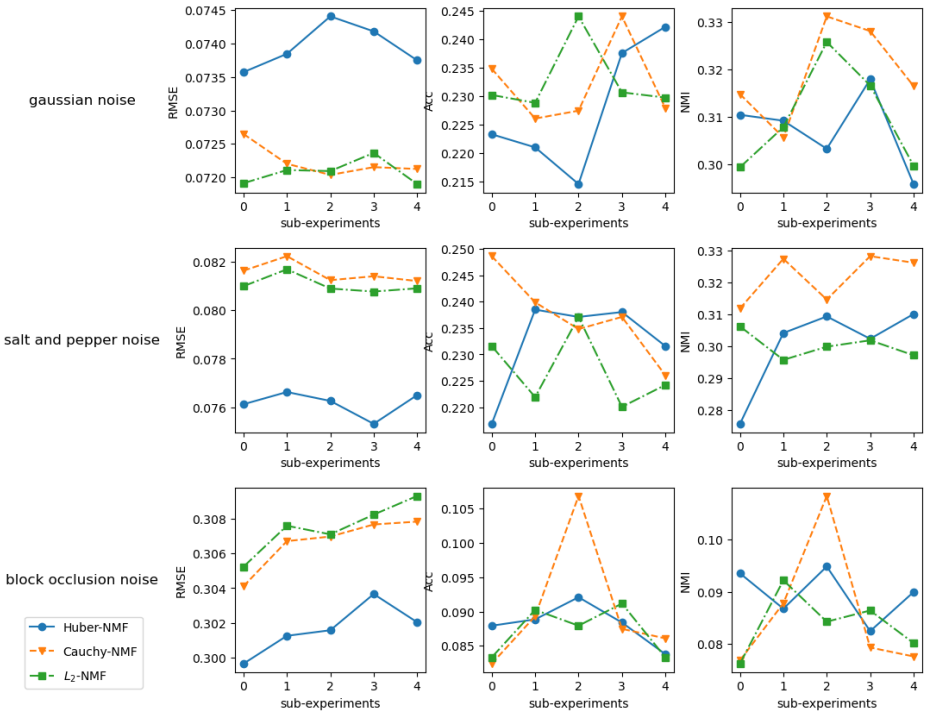

Figure 7: CroppedYaleB experiments results

Specifically, Huber-NMF consistently outperforms the other algorithms in terms of RMSE, when under salt and pepper noise and Block Occlusion Noise conditions.

Table 2: CroppedYaleB experiments results

Noise	Metrics		L_2-NMF	Cauchy-NMF	Huber-NMF
gaussian noise	RMSE	mean	**0.072079**	0.072235	0.073950
		std	**0.000169**	0.000214	0.000301
	Acc	mean	**0.232689**	0.232044	0.227716
		std	**0.005696**	0.006710	0.010432
	NMI	mean	0.309858	**0.319214**	0.307325
		std	0.010135	0.009281	**0.007381**
salt and pepper noise	RMSE	mean	0.081052	0.081544	**0.076166**
		std	**0.000327**	0.000376	0.000461
	Acc	mean	0.226980	**0.237293**	0.232413
		std	**0.006399**	0.007312	0.008166
	NMI	mean	0.300206	**0.321716**	0.300351
		std	**0.003701**	0.006971	0.012686
block occlusion noise	RMSE	mean	0.307481	0.306649	**0.301651**
		std	0.001344	0.001333	**0.001287**
	Acc	mean	0.087201	**0.090424**	0.088214
		std	0.003328	0.008502	**0.002646**
	NMI	mean	0.083880	0.086003	**0.089513**
		std	0.005427	0.011809	**0.004507**

A comparison between Cauchy-NMF and L_2-NMF reveals marginal superiority of the former in the context of Block Occlusion Noise and of the latter in the case of salt and pepper noise. Nevertheless, across all three distinct noise conditions and five sub-experiments, Cauchy-NMF and L_2-NMF obtains comparable RMSE values.

As corroborated by the data presented in Table 2, Huber-NMF excels most of evaluation metrics for Block Occlusion Noise and registers the lowest RMSE for salt and pepper noise. Conversely, Cauchy-NMF exhibits optimal performance in terms of Accuracy (Acc) and Normalized Mutual Information (NMI), whereas L_2-NMF demonstrates the least variability in metric scores (std), under salt and pepper noise conditions. Finally, L_2-NMF emerges as the superior algorithm under Gaussian Noise, while Cauchy-NMF yielding comparable results.

4.3 Discussion

Based on the experimental results obtained on both the ORL and CroppedYaleB datasets (Section 4.2), several key observations can be made regarding the performance of L_2-NMF, Cauchy-NMF and Huber-NMF under different noise conditions.

Firstly, L_2-NMF and Cauchy-NMF generally exhibit comparable performance across all noise conditions, as evidenced by their similar RMSE values. However, Cauchy-NMF shows a slight outperformance over L_2-NMF in scenarios involving Block Occlusion Noise. This could potentially be attributed to the Cauchy loss function's ability to better handle sparse and occlusive noise, as it was originally designed to handle extreme outliers.

Secondly, Huber-NMF stands out as the most robust algorithm when dealing with salt and pepper noise and Block Occlusion Noise, outperforming both L_2-NMF and Cauchy-NMF across multiple evaluation metrics, including RMSE, Acc, and NMI. This superior performance may be due to the Huber loss function's capacity to balance robustness and efficiency, making it more adaptable to a variety of noise distributions.

However, Huber-NMF's performance deteriorates in the presence of Gaussian Noise, where L_2-NMF takes the lead. This could be because the Huber loss function is less effective when the noise follows a Gaussian distribution, a condition under which the L_2 norm is known to be optimal.

5 Conclusions and future work

5.1 Conclusions

We have introduced three different NMFs which are L_2-NMF Cauchy-NMF and Huber-NMF and present their natures and algorithms. We design three different kinds of noise to contaminate data which are Gaussian Noise salt and pepper noise and Block Occlusion Noise in order to show and compare the performance and robustness of different NMF by implementing them on different contaminated dataset under various noise conditions.

In this study, we present the experimental design and results. Our evaluations reveal that L_2-NMF and Cauchy-NMF offer comparable performance with slight advantages under specific noise conditions, Huber-NMF emerges as the most versatile, albeit with limitations under Gaussian noise.

The result is meaningful to refer to when implementing NMF on the real life application under different condition. Our study elucidates the imperative of algorithm selection tailored to specific noise types, underscoring the necessity for a condition-oriented approach in employing NMF methodologies in image processing.

5.2 Future work

Considering the inherent limitations of NMF dealing with noise and outlier, in the future we plan to explore more robust NMF variants. We find there are more derivatives of Cauchy-NMF like Truncated Cauchy NMF, Cauchy-balanced NMF, it is meaningful for us to these and discuss the performance of those derivatives between the experiment we have already done, so it will help us to uncover which version of NMF offers superior handling of noise and outliers.

We also decide to explore more complex NMF like Deep NMF which is the combination of non-negative matrix factorization principles with deep learning architecture similar to deep neural networks. So we plan to make the cross-disciplinary research of deep neural network and NMF in the future study to increase the capability of NMF in the real world application.

References

[1] Charles Boncelet. 4.5 - image noise models. In *Handbook of Image and Video Processing*, pages 397–409. Elsevier Inc, second edition edition, 2005.

[2] Philippe Cattin. Image restoration: Introduction to signal and image processing, 2012. MIAC, University of Basel.

[3] Liang Du, Xuan Li, and Yi-Dong Shen. Robust nonnegative matrix factorization via half-quadratic minimization. In *2012 IEEE 12th International Conference on Data Mining*, pages 201–210, 2012.

[4] Naiyang Guan, Dacheng Tao, Zhigang Luo, and John Shawe-Taylor. Mahnmf: Manhattan non-negative matrix factorization. *Journal of Machine Learning Research*, 07 2012.

[5] Deguang Kong, Chris Ding, and Heng Huang. Robust nonnegative matrix factorization using l21-norm. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, CIKM '11, page 673–682, New York, NY, USA, 2011. Association for Computing Machinery.

[6] Daniel Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13. MIT Press, 2000.

[7] H. Sebastian Seung and Daniel D Lee. Learning the parts of objects by non-negative matrix factorization. *Nature (London)*, 401(6755):788–791, 1999.

[8] He Xiong, Deguang Kong, and Feiping Nie. Cauchy balanced nonnegative matrix factorization. *Artif. Intell. Rev.*, 56(10):11867–11903, mar 2023.

A Running the code

Our code file is compatible with both CPU and GPU environments, with the GPU offering faster computational speeds, and is also supported for environment within Colab.

For easy presentation of the output results, we employed the Jupyter Notebook format.

The `main.ipynb` contains all experimental content from this report, segmented into 7 sections.

1. Initialization
2. Functions for Load Dataset and Pre-processing
3. Functions for Adding Noise
4. Functions for Evaluation Metrics
5. NMF algorithms implementation
6. Experiments
7. Generate Charts for the Experiments Results

For most functions, setting `DEBUG_MODE=True` will produce a simple example as output.

Please load this file into your Jupyter Notebook environment and execute the `run all` command to run the complete experiment.

Using the default parameters, it will take 12 minutes for the ORL dataset and 95 minutes for the CroppedYaleB dataset (based on a laptop with an AMD Ryzen 5800h CPU). It will take 5 minutes for the ORL dataset and 40 minutes for the CroppedYaleB dataset (based on the NVIDIA T4 GPU).

B Contributions

Kaiyue Wang:

- Study of the relevant academic paper
- Implementation of the L_2-NMF algorithm
- Experimental design
- Implement experimental functions
- Setup the experimental environment
- Participate in running experiments
- Participate in online/offline meeting
- Report: L_2-NMF in Methods section; Experimental design in Experimental and Discussions section; Conclusion and future work section; review the report

Peng Huang:

- Study of the relevant academic paper
- Implementation of the Huber-NMF algorithm and noise functions
- Design the code file architecture and the experimental scheme and procedure
- Break down the work package into tasks
- Participate in running experiments
- Conduct online/offline group meetings
- Report: Abstract and Introduction section; Noise introduction in Methods section; Draw charts and tables in the report; review the report

Shoudi Huang:

- Study of the relevant academic paper
- Implementation of the Cauchy-NMF algorithms
- Review coding and set up GPU environment
- Participate in running experiments
- Participate in online/offline meeting
- Report: Previous work section; Cauchy-NMF, Huber-NMF, and pre-processing subsections in Methods section; Experimental Results and Discussion subsections in Experiments and Discussions section; review the report