Compare robustness of nonnegative matrix factorization algorithms

Chen Chen 480458339
Xiaodan Gan 440581983
Xinyue Wang 440359463

21st September 2018

Abstract

Abstract text goes here, justified and in italics. The abstract would normally be one paragraph long.

Contents

1 Introduction

Non-negative matrix factorization (NMF) is a matrix decomposition technique that approximates a data matrix with non-negative entries with the multiplication of two non-negative matrices

$$V \approx WH.$$

In this approximation, matrix W is the basis and matrix H is the weight matrix corresponding to the new dictionary W. NMF is applicable to an extensive range of domain. For example, Lee and Seung [1999] suggest that NMF is useful for image processing problems including facial recognition.

Matrices W and H are often referred as the basis images and weights. This is because the observed image V is approximated by a linear combination of W with positive coefficients H. Due to the additive nature of the algorithm, the dictionary W are often parts of images that are easily interpretable. This property also distinguishes NMF from other traditional image processing methods such as principal components analysis (PCA). Guillamet and Vitrià [2002] demonstrate that NMF performs better in image classification problems in comparison with PCA. Moreover, NMF is also applicable to text mining such as semantic analysis. Generally, NMF is useful to discover semantic features of an article by counting the frequency of each word, and then approximating the document from a subset of a large array of features [Lee and Seung, 1999].

In practice, face images could be easily corrupted during data collection by noise with large magnitude. Corruption may result from lighting environment, facial expression or facial details. An NMF algorithm that is robust to large noise is desired for real-world application. Therefore, the objective of this project is to analyse the robustness of NMF algorithms on corrupted datasets. We implement

two NMF algorithms proposed by Lee and Seung [2001] on real face image datasets, ORL dataset and Extended YaleB dataset. We add artificial noises to the face images are contaminated. We then apply nonparametric statistical methods to analyse the robustness of these two algorithms from simulation results.

2 Related work

Researchers proposed many NMF algorithms. As the objective function of NMF is non-convex, for which the traditional gradient decent method could be very sensitive to step sizes and converge slowly, Lee and Seung [2001] first proposes to algorithms which minimises Euclidean distance or Kullback-Leibler divergence (KLNMF) between the original matrix and its approximation. Although these algorithms are easy to implement and have reasonable convergent rate [Lee and Seung, 2001], they require more iterations than alternatives such as gradient descent [Berry et al., 2007]. Also, the algorithms may fail on seriously corrupted dataset which violates its assumption of Gaussian noise or Poisson noise, respectively [Guan et al., 2017]. Moreover, Yang et al. [2011] indicate that these methods are sensitive to the initial selection of matrices W and H. The algorithms require many iterations to retrieve from poorly selected initial values.

Apart from different loss functions, several optimization methods were proposed to improve the performance of NMF. After Lee and Seung [2001] proposed multiplicative update rules MUR, Lin [2007] proposed a modified MUR which guaranteed the convergence to a stationary point. This modified MUR, however, did not improve the convergence rate of traditional MUR [Guan et al., 2012]. Moreover, as MUR does not impose sparseness, Berry et al. [2007] proposed a projected nonnegative least square (PNLS) method to enforce sparseness. In each nonnegative least square sub-problem, this algorithm projects the negative elements of least squares solution directly to zero. Nevertheless, PNLS does not guarantee convergence [Guan et al., 2012]. In contrast to these gradient-based optimization methods, Kim and Park [2008] presented an active set method which divides variables into two sets, free set and active set. They update free set in each iteration by solving a unconstrained equation. Even though the active set method has a nice convergence rate, it assumes strictly convexity in each nonnegative least square sub-problem [Kim and Park, 2008]. These assumptions are easily violated in real life applications.

There exists many robust NMF algorithms which include penalties in the objective functions. For example, Lam [2008] proposes L_1-norm based NMF to model noisy data by a Laplace distribution which is less sensitive to outliers. However, as

L_1-norm is not differentiable at zero, the optimization procedure is computationally expensive. Kong et al. [2011] proposed an NMF algorithm using L_{21}-norm loss function which is more stable. The updating rules used in L_{21}-norm NMF, however, still converge slowly because of a continual use of the power method [Guan et al., 2017].

3 Methods

Some carefully designed NMF are robust to various noises. These robust algorithms aim to significantly reduce the amount of noise while preserving the edges without blurring the immage [Barbu, 2013].

3.1 NMF and Gaussian noise

Gaussian noise is a noise with a probability density function being normal with mean zero. Lee and Seung [2001] propose the first NMF with the objective function between imaga V and its NMF factorisation W and H being

$$\|V - WH\| = \sum_{ij} \left[V_{ij} - (WH)_{ij}\right]^2. \tag{1}$$

To minimise this object function of least square, Lee and Seung [2001] prove the convergence of the multiplication update rule

$$H_{jk} \leftarrow H_{jk} \frac{(W^T V)_{jk}}{(W^T WH)_{jk}} \text{ and } W_{ij} \leftarrow W_{ij} \frac{(VH)_{ij}}{(WHH^T)_{ij}}. \tag{2}$$

Here, $()_{ij}/()_{ij}$ denotes elementwise division of the two matrix. Liu and Tao [2016] shows this NMF algorithm minimises Gaussian.

3.2 KLNMF and Poisson noise

Poisson noise or shot noise is a type of electronic noise that occurs when the finite number of particles that carry energy, such as electrons in an electronic circuit or photons in an optical device, is small enough to give rise to detectable statistical fluctuations in a measurement. Lee and Seung [2001] suggest that KLNMF is a

algorithm that minimising the Kullback-Leibler divergence

$$\begin{aligned} D(V||WH) &= \sum_{ij}\left(V_{ij}\log\frac{V_{ij}}{(WH)_{ij}} - V_{ij} + (WH)_{ij}\right) \\ &= \sum_{ij}\left(-V_{ij}\log(WH)_{ij} + (WH)_{ij} + C(V_{ij})\right). \end{aligned} \tag{3}$$

where $C(V_{ij}) = V_{ij}\log V_{ij} - V_{ij}$. $C(V_{ij})$ is a function of the observed image matrix V only. Lee and Seung [2001] also suggest a multiplication update rule to find as the optimisation procedure of KLNMF

$$H_{jk} \leftarrow H_{jk}\frac{\sum_i W_{ij}V_{ik}/(WH)_{jk}}{\sum_{i'} W_{i'j}} \text{ and } W_{ij} \leftarrow W_{ij}\frac{\sum_k H_{jk}V_{ik}/(WH)_{jk}}{\sum_{k'} H_{ik'}}. \tag{4}$$

As this original image matrix V is observed, minimising this Kullback-Leibler divergence (3) is equivalent to minimising

$$\sum_{ij}\left(-V_{ij}\log(WH)_{ij} + (WH)_{ij} + C(V_{ij})\right).$$

, for arbitrary bounded function $C(V_{ij})$. Taking exponential of the negative of this score function, the problem transforms to maximising the following likelihood function

$$L(WH|V) = \prod_{ij}\left((WH)_{ij}^{V_{ij}}\, e^{-(WH)_{ij}} + C(V_{ij})\right).$$

Choosing constant $C(V_{ij})$ to be $-\log V_{ij}!$ gives

$$L(WH|V) = \prod_{ij}\left(\frac{(WH)_{ij}^{V_{ij}}\, e^{-(WH)_{ij}}}{V_{ij}!}\right).$$

Hence, the probability density function of each element of the original matrix V is Poisson

$$P(V_{ij}) = \frac{(WH)_{ij}^{V_{ij}}\, e^{-(WH)_{ij}}}{V_{ij}!}$$

is a sufficient condition to yield this likelihood. Hence KLNMF is most suitable for images with Poisson noise.

3.3 Preprocess

We did not preprocess the images because both of NMF and KLNMF are scale sensitive, because both objective functions (1) and (3) varies when original matrix V and result matrices W and H scale proportionally, i.e. for λ real, $D(V||WH) \neq D(\lambda V||\lambda WH)$. To overcome this issue, we could have normalise the matrices W and H in each iteration [Fevotte and Idier, 2011], but it will result in a even slower computation.

3.4 Gaussian and Poisson are asymptotic equivalent

We design an Gaussian noise and a Poisson noise with different magnitude. Poisson distribution with parameter λ (integer) is equivalent to the sum of λ Poisson distributions with parameter 1 [Walck, 1996, p. 45]. Hence for λ large, Central Limit Theorem implies that Poisson distribution with parameter λ is well approximated by $N(\lambda,\lambda)$. When applying Poisson noise to an image, we do not have degree of freedom to choose any parameter. The variance is the magnitude of the pixels. To compare the robustness of KLNMF with NMF with different noise, we choose the variance of Gaussian noise to be the different from the magnitude of the pixel, that is, $N(0,\text{Var}) \neq N(0,V) \approx \text{Poi}(V) - V$. Figure 1 visualises the similarity of Poisson distribution and Normal distribution with parameter $V = 40$. To overcome this issue, the Gaussian noise we use should be larger or smaller variance in comparison with the mean of the images. We choose the variance of the noise to be 80.

3.5 Salt & Pepper noise

Apart from Gaussian and Poisson noises, we also test our two algorithms on the commonly seen **Salt & Pepper noise** noise. The noise presents itself by having dark pixels in bright regions and bright pixels in dark regions [Sampat et al., 2005].

3.6 Multiple initial estimates assure the algorithms stable

As discussed in the section of related work, The problem of nonnegative matrix factorization is not a convex problem. Hence the results update rules (2) and (4) coverage to may be local minima instead of global minima, depending on the initial approximation. Our task was to compare the robustness of the algorithm and we do not want the instability of our algorithms to affect our comparison. To address this issue, we implement several (i.e. n) initial estimates for each matrix

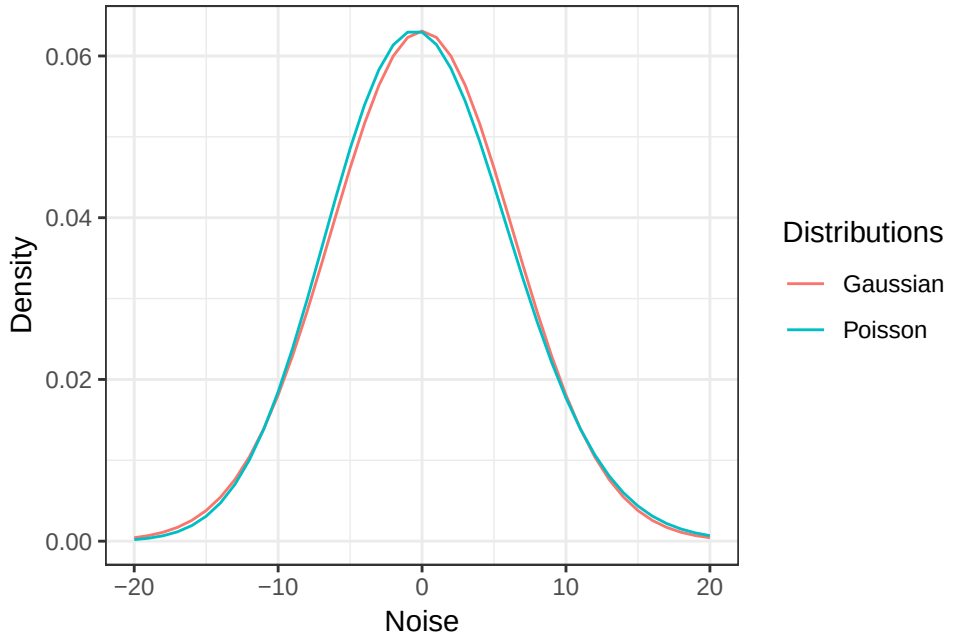

Figure 1: Compare a Gaussian noise $\mathrm{N}(0,40)$ with Poisson noise $\mathrm{Poi}(40)-40$. They two distributions are asymptotically equivalent and have similar density functions.

factorization problem. We use the factorized matrices W and H corresponding to the least residual (1) and (3), for NMF and KLNMF, algorithms respectively, as the final result of factorization. This design of multiple starting point improves the stability of of algorithm, but it requires more computational power. To improve the computational speed, we make the number n equal to the number of cores of the CPU. We assign each of the n initial estimates randomly with an uniform distribution. Then these each of the n initial estimates are assigned to a different core of the CPU. This boost the CPU utilisation to 100% instantly and improved the computational speed by 70% on the ORL data. The following part of our code implements this idea of parallel computing.

Algorithm 1: Centring image data

```
args = zip(repeat(V,ncpu), repeat(r,ncpu),
    repeat(niter[name2],ncpu), repeat(min_error[name2],ncpu))
result = pool.starmap(algo, args)
```

where `algo` is the NMF algorithm and `niter` is the number of iterations. We use Chen's 16-thread Xeon high performance computer (Figure 3.6) to run this algorithm.

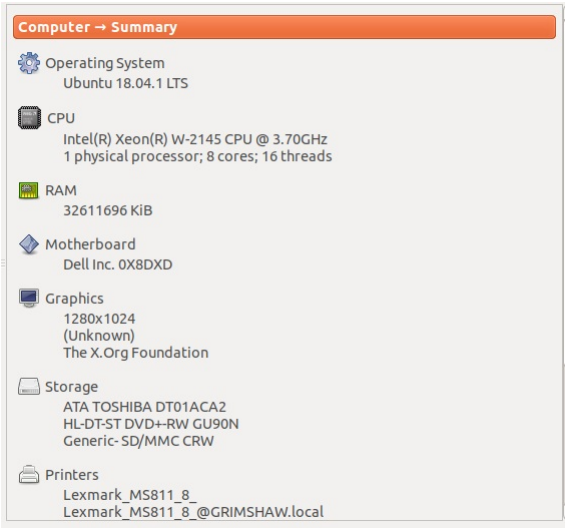

Figure 2: The algorithms run in this high performance computer so that the computing time is reasonable. The multiple initial estimates assure the algorithms are stable.

3.7 KLNMF requires more iterations

KLNMF converges slower than NMF. We made the plot of the logarithm of residual (1) and (3) with respect to the logarithm of the number of iterations. We measured the slope of the log-log plot for both algorithms and found that the slope of the NMF residual plot is 2.5 times as that of the KLNMF plot (Figure 3). This estimates that the rate of convergence of NMF is 2.5 faster than KLNMF. As a result, we set the number of iterations as 500 and 1200 for NMF and KLNMF algorithms, respectively (i.e. roughtly 2.5 more iterations).

3.8 Evaluation metrics and their confidence intervals

The assignment instruction asks us to compare the performance of NMF and KLNMF by using evaluations metrics including Relative Reconstruction Errors (RRE), Average Accuracy (AA), Normalized Mutual Information (NMI). The instruction states the formulae of these metrics. However, to systematically compare the metrics, we construct an 95% confidence interval for any metric (e.g. RRE) by bootstrapping percentile confidence interval. The idea of bootstrapped is straightforward—we resample a subset of 40 samples among the sample space of 80 simulations and calculate the mean. We repeat this process 1000 times. The 2.5% and 97.5% percentiles of the 80 resampled means are then the bootstrapping percentile confidence

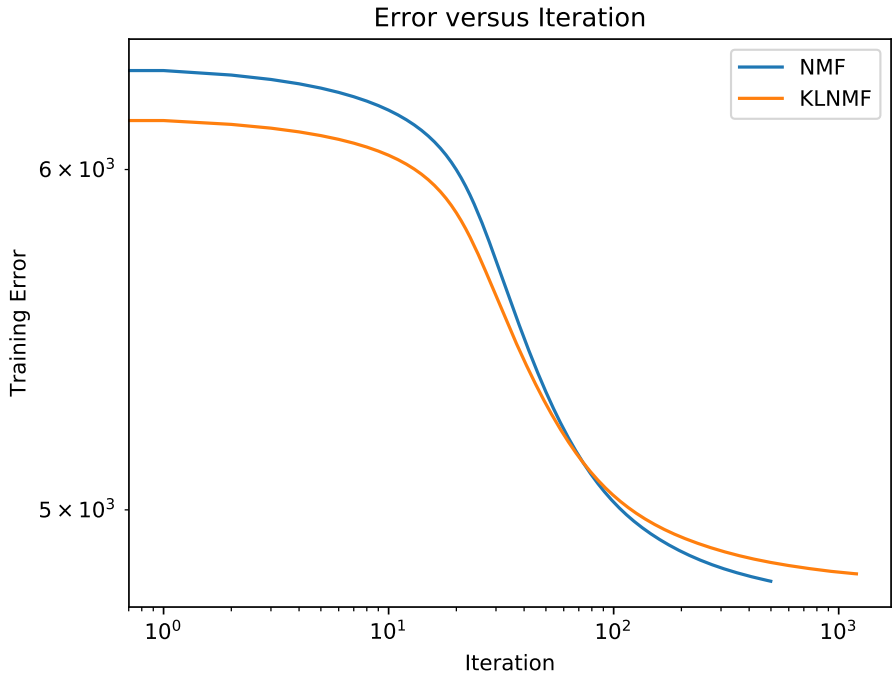

Figure 3: Residual of objective function (1) and (3) versus the number of iterations. NMF converges more than twice faster than KLNMF.

interval

$$(\mathrm{RRE}^*_{2.5}, \mathrm{RRE}^*_{97.5}), \tag{5}$$

where RRE^*_α is the α percentile of the bootstrapped distribution from our sample space with 80 simulations. We run 80 simulations so that the confidence interval we construct is precise.

Boostrapping does not require the sample space follows specific distributions. We apply this nonparametric method to construct confidence interval here because we do not have the knowledge about the specific distributions of these three metrics.

3.9 Statistical method compares the robustness of algorithms

We implement Kolmogorov-Smirnov test to test the hypothesis that the algorithms NMF and KLNMF have different robustness. Again Kolmogorov-Smirnov test is distribution free, so we do not need to know the distributions of the evaluation metrics.

Let RRE_i denote the RRE generated from our NMF algorithm by the ith simulation.

Define the empirical distribution of a sample set generated by algorithm α, perhaps the 80 RRE results, as

$$\hat{F}_\alpha(x) = \frac{1}{n}\sum_{i=1}^{80} 1_{RRE_i \le x}. \tag{6}$$

The test statistic is the supremum among the differences of the empirical distribution generated using definition (6) [Walck, 1996]

$$D = \sup_x |F_{\alpha_2}(x) - F_{\alpha_2}(x)|. \tag{7}$$

We compare the test statistics D with the critical value of 0.215, which corresponds to 80 samples and a 95% of confidence level. We reject the null hypotheses that the two algorithms produce similar RRE (or other evaluation metrics) with 95% confidence level if the test statistics $D > 0.215$. The technique is extended to compare AA and NMI.

4 Experiments

4.1 Dataset

We illustrate our two NMF algorithms on two real-world face image datasets: ORL and Extended YaleB (Belhumeur et al. [1997]). Both ORL and Extended YaleB datasets contain multiple images of distinct subjects with various facial expression, lighting condition, and facial details. Images in ORL are cropped and resized to 92×112 pixels. We further rescale it to 30×37 pixels. Similarly, we reduce the size of images in Extended YaleB to 42×48 pixels. For each dataset, we flatten the image matrix into a vector and append them together to get a matrix V with shape $d \times n$ where integer d is the number of pixels in one image and integer n is the number of images. In each epoch, we use 90% of data.

4.2 Noise

We implement three kinds of noises including Gaussian noise, Poisson noise and Salt & Pepper noise.

4.2.1 Gaussian Noise

We design the Gaussian noise by normal distribution with 0 mean and 80 standard deviation (Algorithm 2). The ORL dataset has a global pixel mean of 40 and the CroppedYale data set has that of 70. Hence the designed Gaussian noise contaminates the images significantly. We choose the standard deviation to be 80 so that our Gaussian noise are less likely to coincident with the designed Poisson noise. To satisfy the nonnegative constant, negative value in contaminated image is set to zero

Algorithm 2: Gaussian Noise Design

```
def normal(subVhat):
    """Design a Gaussian noise."""
    V_noise = np.random.normal(0, 80, subVhat.shape) #*
        np.sqrt(subVhat)
    V = subVhat + V_noise
    V[V < 0] = 0
    return V, V_noise
```

4.2.2 Poisson Noise

The Poisson noise is not additive and has no hyperparameters to be set. Unlike Gaussian noise, contaminated images are drawn directly from Poisson distribution with parameter set to be pixel values. Then, the Poisson noise is calculated from the difference between the contaminated image and the original image, as discussed in Section 3 and demonstrated in algorithm 3.

Algorithm 3: Poisson Noise Design

```
def possion(subVhat):
    """Design a Possion noise."""
    V = np.random.poisson(subVhat)
    V_noise = V-subVhat
    return V, V_noise
```

4.2.3 Salt & Pepper Noise

Salt and Pepper noises are added by l[illegible]wing random integers from discrete uniform distribution of the interval [0, 255) [illegible] 4

Algorithm 4: Salt and Pepper Noise Design

```
def salt_and_pepper(subVhat):
```

```
"""Design a salt and pepper noise where make some pixel value
    zeros."""
  V_noise = np.random.randint(low=0, high=255,
      size=subVhat.shape, dtype=int)
  V = subVhat.copy()
  V[V_noise <= 20] = 0
  V[V_noise >= 230] = 255
  return V, V_noise
```

4.3 Experiment Setup

We apply 2 algorithms (NMF and KLNMF) with 4 categories of noises (no noise, Gaussian noise, Poisson noise and Salt & Pepper noise), which results in 8 combinations in each epoch. In each epoch, we randomly select 90% of samples to train NMF algorithms and evaluate three metrics on reconstructed images. The training will terminate when the error reaches the minimum error or the maximum iteration is reached. The minimum error and maximum iteration are hyperparameter which we learn from experiments. We save the learning errors versus number of iteration so that we could draw the plot and observe the convergence of learning process. We increase the number of epochs and calculate the average metricss and confidence interval etc.

4.4 Experiments Results

Figure 4.4 and 4.4 visualise the original image, designed noises, corrupted images and reconstructed images from left to right. From top to bottom, the four rows correspond to no noise, Gaussian noise discussed in Section 4.2.1, Poisson noise discussed in Section 4.2.2 and Salt & Paper noise discussed in Section 4.2.3. The first of Figure 4.4 and 4.4 shows both algorithms reconstructed the original image well without artificial noise and with Poisson noise. However, when the noise is large (the second and last rows in Figure 4.4 and 4.4), the quality of reconstructed images are only marginally more clear than the contaminated image . Our result is consistent with Guan et al. [2017], who assert that NMF may fail to handle extremely corrupted images when assumed distribution of noise is violated. Moreover, the difference between images generated by NMF and KLNMF are not visually significant. Hence, we implement statistical hypothesis test to compare of RREs of the two algorithms.

Substitute RRE results of the two algorithms into Kolmogorov-Smirnovs test (7) gives test statistics $D = 1, 1, 0.6625$ with no noise, Gaussian noise, and Poisson

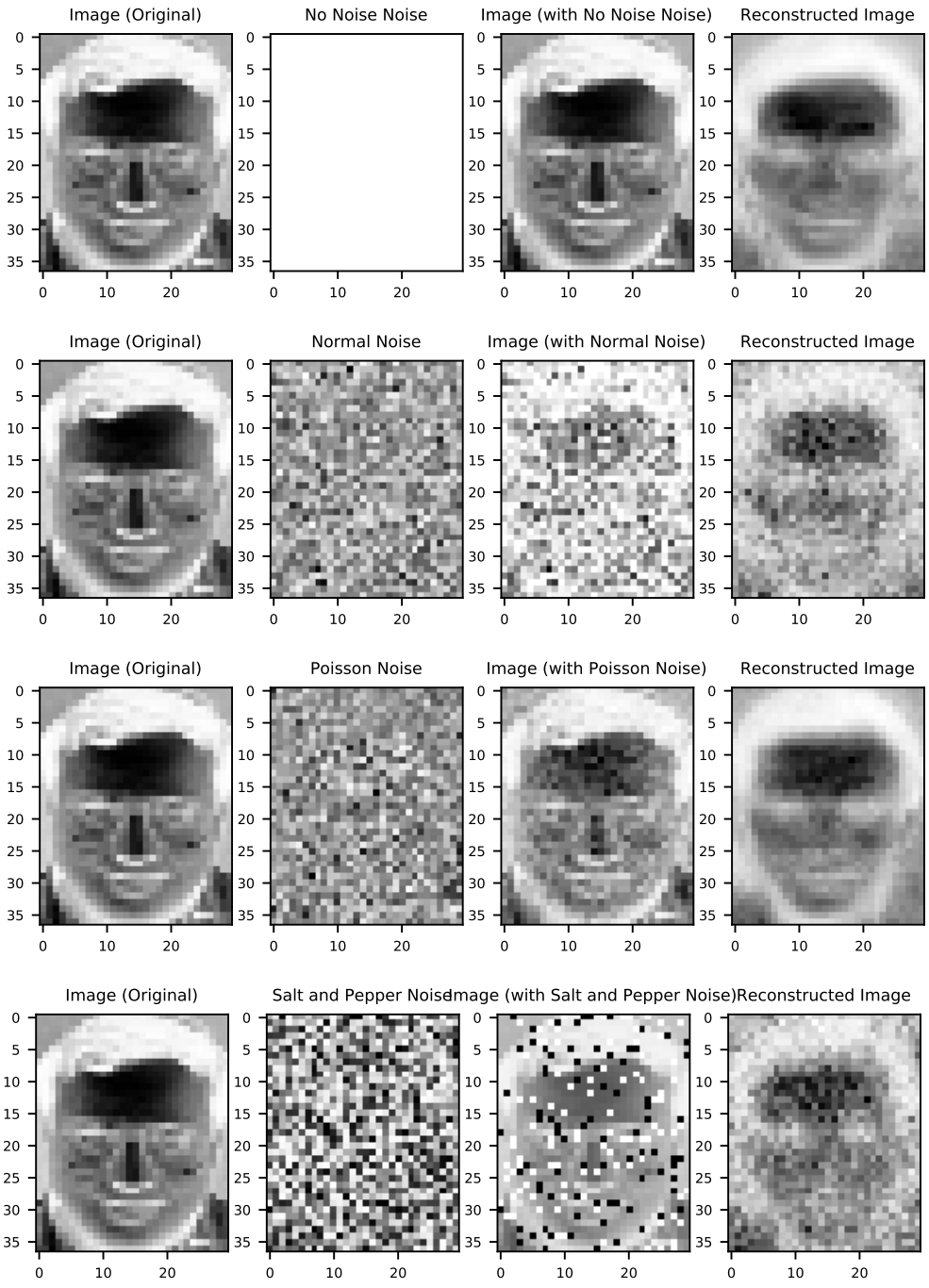

Figure 4: The reconstructed image by **nmf**. The original images (Column 1) are combined with noises (Column 1) including Gaussian Noise with Variance 80 (Row 2), Poisson Noise (Row 3), and Salt & Pepper Noise (Row 4). The corrupted images are shown in Column 3. The reconstructed image are shown in (Column 4). The reconstruction with no noise is shown in Row 1.

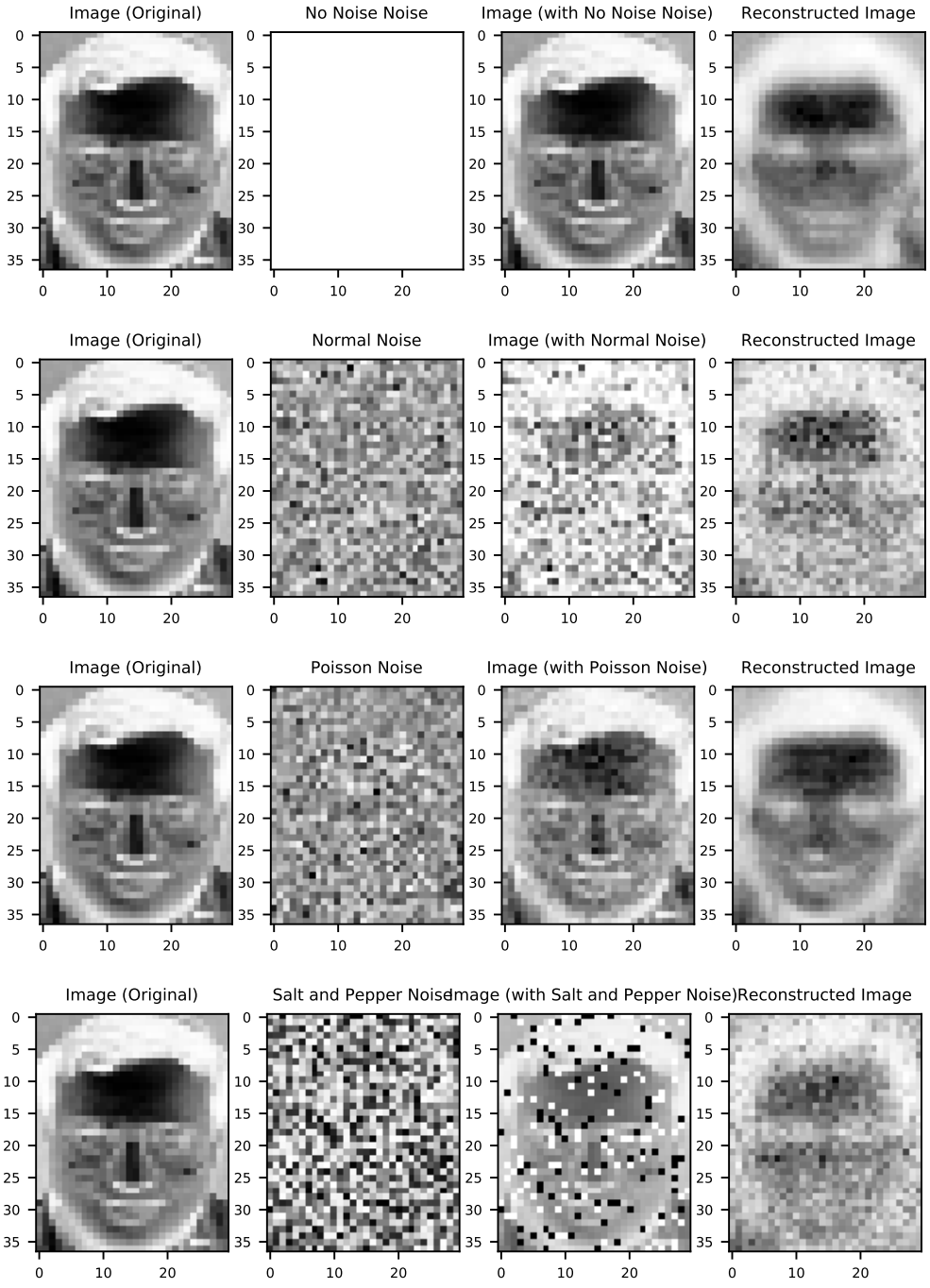

Figure 5: The reconstructed image by **klnmf**. The original images (Column 1) are combined with noises (Column 1) including Gaussian Noise with Variance 80 (Row 2), Poisson Noise (Row 3), and Salt & Pepper Noise (Row 4). The corrupted images are shown in Column 3. The reconstructed image are shown in (Column 4). The reconstruction with no noise is shown in Row 1.

Table 1: Average of evaluations metrics over 80 simulations using the ORL dataset. The 95% confidence intervals are calculated using bootstrap.

ORL dataset	RRE	ACC	NMI
NMF no noise	0.1583 (0.1581, 0.1584)	0.7364 (0.731, 0.742)	0.8536 (0.8506, 0.8567)
NMF Gaussian noise	0.2925 (0.2922, 0.2927)	0.447 (0.4423, 0.4521)	0.6212 (0.6176, 0.6247)
NMF Poisson noise	0.1611 (0.161, 0.1613)	0.7313 (0.7262, 0.7367)	0.8493 (0.8456, 0.8527)
NMF Salt and Pepper noise	0.2636 (0.2634, 0.2638)	0.5094 (0.504, 0.5151)	0.6721 (0.6679, 0.6764)
KLNMF no noise	0.1729 (0.1728, 0.173)	0.7406 (0.7352, 0.7458)	0.8599 (0.8568, 0.8632)
KLNMF Gaussian noise	0.2977 (0.2976, 0.2979)	0.4538 (0.4483, 0.4595)	0.6209 (0.6165, 0.6255)
KLNMF Poisson noise	0.1602 (0.1601, 0.1603)	0.7417 (0.7365, 0.7472)	0.8573 (0.8542, 0.8602)
KLNMF Salt and Pepper noise	0.264 (0.2638, 0.2643)	0.5089 (0.5038, 0.5139)	0.6734 (0.6694, 0.6779)

noise, for the ORL dataset. These three test statistics are all much greater than the critical value 0.215. Hence there are strong evidence that the performance of NMF and KLNMF are different in these three problems. For salt and Pepper noise, test statistic $D = 0.2125 < 0.215$, hence we fail to conclude that the two methods have different robustness against Salt and Pepper noise. Further one tail Kolmogorov-Smirnov test concludes that KLNMF performs better reconstructing the original image with no noise and is more robust Poisson noise. In contrast, NMF is more robust against Gaussian noise, even with only 500 iterations (1200 for KLNMF). Hence, NMF is clearly more robust than KLNMF against Gaussian noise.

Theoretical results discussed in Section 3 suggest NMF is more robust against Gaussian noise whereas KLNMF is more robust against Poisson noise. Our experimental results concluded from Kolmogorov-Smirnovs hypothesis tests agree with these theoretical results. Further, as both of the algorithms are not designed for Salt and Pepper noise, they have similar performance against it.

These results can be observe by directly reading whether the confidence intervals overlaps in table 1. Also, the statistical results agree with the visualisation in Figure 4.4. These results can be intuitively understood—also the differences in the robustness of the two algorithms are small, the even smaller variances in the RRE results make them statistically different under Poisson and Gaussian noise.

In terms of the cropped Yale dataset, results from hypothesis tests show that NMF performs uniformly better than KLNMF, suggesting more iterations are required on KLNMF to compare these two algorithms fairly. We fail to do so because the dataset is much larger than ORL and our multiplicative update rules, especially for KLNMF converge too slow.

Table 2: Average of evaluations metrics over 40 simulations using CroppedYale data set. The 95% confidence intervals are calculated using bootstrap.

ORL dataset	RRE	ACC	NMI
NMF no noise	0.1583 (0.1581, 0.1584)	0.7364 (0.731, 0.742)	0.8536 (0.8506, 0.8567)
NMF Gaussian noise	0.2925 (0.2922, 0.2927)	0.447 (0.4423, 0.4521)	0.6212 (0.6176, 0.6247)
NMF Poisson noise	0.1611 (0.161, 0.1613)	0.7313 (0.7262, 0.7367)	0.8493 (0.8456, 0.8527)
NMF Salt and Pepper noise	0.2636 (0.2634, 0.2638)	0.5094 (0.504, 0.5151)	0.6721 (0.6679, 0.6764)
KLNMF no noise	0.1729 (0.1728, 0.173)	0.7406 (0.7352, 0.7458)	0.8599 (0.8568, 0.8632)
KLNMF Gaussian noise	0.2977 (0.2976, 0.2979)	0.4538 (0.4483, 0.4595)	0.6209 (0.6165, 0.6255)
KLNMF Poisson noise	0.1602 (0.1601, 0.1603)	0.7417 (0.7365, 0.7472)	0.8573 (0.8542, 0.8602)
KLNMF Salt and Pepper noise	0.264 (0.2638, 0.2643)	0.5089 (0.5038, 0.5139)	0.6734 (0.6694, 0.6779)

4.4.1 CroppedYale Evaluations metrics FAKE FOR NOW

The raw metrics of ORL dataset are available here:

- Relative reconstruction error
- Average Accuracy
- Normalized Mutual Information

4.5 Personal Reflection

5 Conclusion

Your conclusion goes at the end, followed by References, which must follow the Vancouver Style (see: www.icmje.org/index.html). References begin below with a header that is centered. Only the first word of an article title is capitalized in the References.

A Appendix

Algorithm 5: The Levenberg–Marquardt algorithm iteratively finds optimal macro-scale Robin boundary conditions.

```
%A=imread('C:\Users\chenc\OneDrive - UNSW\machine
    learning\assignment
    1\data\CroppedYaleB\yaleB01\yaleB01_P00A+000E+00.pgm');
red=3;
```

```
k=40;

imagefiles = dir('data/ORL/*/*.pgm');
imagefiles2=struct2cell(imagefiles);
imagefiles=imagefiles((~endsWith(imagefiles2(1,:),'Ambient.pgm'))');
imagefiles2=struct2cell(imagefiles);
A=imread(strcat(imagefiles(1).folder,'\',imagefiles(1).name));
if size(A,1)==112
    A=A(1:111,1:90);
end
A_list=zeros(size(A,1)/red,size(A,2)/red,red);
nfiles = length(imagefiles);    % Number of files found
matrix_image=zeros(prod(size(A))/red^2,nfiles);
temp=struct2cell(imagefiles);
names=temp(1,:)
for ii=1:nfiles
   currentfilename = imagefiles(ii).name;
   currentfilename=strcat(imagefiles(ii).folder,'\',currentfilename);
   currentimage = imread(currentfilename);
   if abs(size(A,1)-112)<=1
    currentimage=currentimage(1:111,1:90);
   end

   for i=1:red
    A_list(:,:,i)=currentimage(i:red:end,i:red:end);
   end
   A2=uint8(mean(A_list,3));
   matrix_image(:,ii) = A2(:);
end

[w h]=NeNMF(matrix_image,k);
idx = kmeans(h',k)
Y_pred=zeros(size(matrix_image,2),1)
namess=str2mat(string(imagefiles2(2,:))')
namess=str2num(namess(:,end-1:end))
for ii=unique(idx)'
   ind= (idx==ii);
   Y_pred(ind)=mode(namess(ind,:));
end
nmi(Y_pred,namess)
```

References

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

David Guillamet and Jordi Vitrià. Non-negative matrix factorization for face recognition. In *Topics in artificial intelligence*, pages 336–344. Springer, 2002.

Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 556–562. MIT Press, 2001. URL http://papers.nips.cc/paper/1861-algorithms-for-non-negative-matrix-factorization.pdf.

Michael W Berry, Murray Browne, Amy N Langville, V Paul Pauca, and Robert J Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. *Computational statistics & data analysis*, 52(1):155–173, 2007.

Naiyang Guan, Tongliang Liu, Yangmuzi Zhang, Dacheng Tao, and Larry Steven Davis. Truncated cauchy non-negative matrix factorization for robust subspace learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

Zhirong Yang, He Zhang, Zhijian Yuan, and Erkki Oja. Kullback-leibler divergence for nonnegative matrix factorization. In *International Conference on Artificial Neural Networks*, pages 250–257. Springer, 2011.

Chih-Jen Lin. On the convergence of multiplicative update algorithms for non-negative matrix factorization. *IEEE Transactions on Neural Networks*, 18(6): 1589–1596, 2007.

Naiyang Guan, Dacheng Tao, Zhigang Luo, and Bo Yuan. Nenmf: An optimal gradient method for nonnegative matrix factorization. *IEEE Transactions on Signal Processing*, 60(6):2882–2898, 2012.

Hyunsoo Kim and Haesun Park. Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. *SIAM journal on matrix analysis and applications*, 30(2):713–730, 2008.

Edmund Y Lam. Non-negative matrix factorization for images with laplacian noise. In *Circuits and Systems, 2008. APCCAS 2008. IEEE Asia Pacific Conference on*, pages 798–801. IEEE, 2008.

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*, pages 673–682. ACM, 2011.

Tudor Barbu. Variational image denoising approach with diffusion porous media flow. In *Abstract and Applied Analysis*, volume 2013. Hindawi, 2013.

Tongliang Liu and Dacheng Tao. On the performance of manhattan nonnegative matrix factorization. *IEEE Transactions on Neural Networks and Learning Systems*, 27(9):1851–1863, September 2016. doi:10.1109/TNNLS.2015.2458986.

Cedric Fevotte and Jerome Idier. Algorithms for nonnegative matrix factorization with the -divergence. *Neural Computation*, 23(9):2421–2456, 2011. doi:10.1162/NECO_a_00168. URL https://doi.org/10.1162/NECO_a_00168.

Christian Walck. *Hand-book on statistical distributions for experimentalists*. 1996. URL http://www.fysik.su.se/~walck/suf9601.pdf.

Mehul P Sampat, Mia K Markey, Alan C Bovik, et al. Computer-aided detection and diagnosis in mammography. *Handbook of image and video processing*, 2(1): 1195–1217, 2005.

Peter N Belhumeur, João P Hespanha, and David J Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. Technical report, Yale University New Haven United States, 1997.