Assignment 1

Professor: Tongliang Liu
Group members: Alex Elias (UniKey: aeli0392, SID: 500407020) & Luca Quaglia (UniKey: lqua3126, SID: 500175059)

Abstract

A non negative matrix $X \geq 0$ can be approximately factored into two non negative matrices $D \geq 0$ and $R \geq 0$ by minimising an objective function of the form $\|X - D\,R\|$, where $\|\cdot\|$ is a matrix norm. Two types of norm are are explored: L_2 norm NMF and $L_{2,1}$ norm NMF. Their robustness to noise is investigated through numerical experiments using two datasets of images of human faces and a variety of noise types.

1 Introduction

Non negative data abound in science, engineering and other fields. By nature, these type of data is such that all its components are non negative. A common example are images: an image can be represented as a matrix of integer numbers (i.e. pixels values) that encode specific colours. The range of pixels values is often in the range 0 to 255 or 0 to 65535, so they are all non negative.

Non negative matrix factorisation (NMF) is a very powerful technique to learn some meaningful representation of non negative data. Let's suppose that we have a dataset X of examples, where X is a matrix with f rows and n columns and with all non negative entries. Each column is an example described by f features. And the dataset X is composed by n examples. The NMF algorithm aims at finding two non negative matrices, D and R, such that their product is similar to X. More formally:

$$X \approx D\,R \quad X \in \mathbb{R}^{f\cdot n}, D \in \mathbb{R}^{f\cdot k}, R \in \mathbb{R}^{k\cdot n} \quad X \geq 0, D \geq 0, R \geq 0 \tag{1}$$

The factorisation is not exact, like for example the LU factorisation of a square matrix, but only approximate. To make this notion of "approximate" rigorous we need to introduce a definition of norm $\|\cdot\|$ for a matrix. We can use various definition for the norm but as long as the definition satisfies the following 3 properties:

$$\|A + B\| \leq \|A\| + \|B\| \quad \forall A, B \in \mathbb{R}^{f\cdot n} \tag{2}$$

$$\|\lambda A\| = |\lambda|\,\|A\| \quad \forall A \in \mathbb{R}^{f\cdot n}, \forall \lambda \in \mathbb{R} \tag{3}$$

$$\|A\| = 0 \implies A = \mathbf{0} \tag{4}$$

the norm $\|\cdot\|$ is well defined.

Given a definition of norm, the meaning of $X \approx D\,R$ can be precisely expressed as:

$$X \approx DR \quad \leftrightarrow \quad D, R = \underset{D\geq 0, R\geq 0}{\arg\min} \|X - D\,R\| \tag{5}$$

The NMF thus becomes a minimisation problem.

The intention of the matrices D and R is to serve as a dictionary of features for the dataset X and a representation of the dataset X, respectively. D can be thought as a basis of k vectors and R can be thought as containing the coefficients over this basis for the examples in the dataset. The basis vectors in the dictionary D are non-orthogonal, unlike the basis obtained with PCA (Principal Component Analysis). Often the NMF algorithm leads to sparse representations, unlike the representation from PCA. This sparsity attribute often outweighs the orthogonality attribute. Another point of contrast with PCA is that the non-negativity of the features extracted with the NMF algorithm is a plus if the data are intrinsically non-negative, as PCA often leads to negative coefficients that do not have physical sense for non-negative data.

Once a dictionary D has been found, we can represent a single example $\curvearrowleft$, for example, as:

$$\mathbf{x} \approx D\,\mathbf{r} \tag{6}$$

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_f \end{pmatrix} \approx \begin{bmatrix} d_{11} & \dots & d_{1k} \\ \vdots & \ddots & \vdots \\ d_{f1} & \dots & d_{fk} \end{bmatrix} \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_k \end{pmatrix} \tag{7}$$

The dictionary D needs to be learnt over a set of examples (the matrix X) and hopefully it can manage to capture some salient features of these examples. If that happens, the dictionary D can then be used for a series of tasks, for example, to remove noise from noisy examples. Let's consider a dataset X of clean examples and to learn a dictionary D. If we are given a noisy example $(\tilde{x})$, we can first find its representation over the dictionary D:

$$\tilde{\mathbf{x}} \approx D\,\tilde{\mathbf{r}} \tag{8}$$

The reconstructed de-noised example is simply $D\,\tilde{\mathbf{r}}$.

2 Related Work

NMF was introduced in the late 1990's by Lee and Seung [1, 2] and it has found many applications in a variety of fields: image processing, pattern recognition, chemistry, biology... [3] There are various ways of implementing the non-negative matrix factorisation procedure. All methods can be framed in the general way presented in the previous section, transforming the factorisation problem into a minimisation problem. The main difference of most of the various methods is in the definition of the norm $\|\cdot\|$.

In the classical NMF method (also called L_2 norm NMF) [1, 2], the norm that is used is the Frobenius norm:

$$\|A\|_2 = \sqrt{\sum_{ij} a_{ij}^2} \tag{9}$$

In the Hypersurface Cost Based NMF [4], the norm that is used is:

$$\|A\|_{HS} = \sum_{ij} \left(\sqrt{1 + a_{ij}^2} - 1 \right) \tag{10}$$

In the L_1 norm NMF [4], the norm that is used is:

$$\|A\|_1 = \sum_{ij} |a_{ij}| \tag{11}$$

In the $L_{2,1}$ norm NMF [5], the norm that is used is:

$$\|A\|_{2,1} = \sum_j \sqrt{\sum_i a_{ij}^2} \tag{12}$$

Given a dataset X, all methods try to solve the following minimisation problem:

$$D, R = \underset{D \geq 0, R \geq 0}{\arg\min} \|X - D\,R\| \tag{13}$$

As outlined in [4], the L_2 norm NMF has a simple iterative scheme to update D and R, given some initial random guesses for the two matrices. The iterative method is quite fast. The Hypersurface Cost Based NMF leads to a gradient descent method that is quite time consuming because it invokes Armijo's rule based line search. The L_1 norm NMF has a non-smooth objective function and it needs to be approximated to perform the minimisation procedure. This leads to inefficiencies when the dataset is large. The $L_{2,1}$ norm NMF has a simple iterative scheme to update D and R, similar to the one for the L_2 norm NMF. This iterative method is also quite fast.

As outlined in [4], the various method have different tolerance to the presence of outliers and noise. The L_2 norm NMF is not very robust because the L_2 norm is sensitive to outliers. The L_1 norm NMF is less sensitive to outliers for small dataset but its robustness worsen when the dimensionality of the dataset increases. The $L_{2,1}$ norm NMF is more than the traditional L_2 norm NMF because the $L_{2,1}$ norm lessens the impact of the outliers.

3 Methods

3.1 Traditional NMF

The traditional NMF algorithm seeks to find, given a non-negative matrix X, two non-negative matrices D and R such that $X \approx DR$:

$$D, R = \underset{D \geq 0, R \geq 0}{\arg\min} \|X - DR\|_2^2 \tag{14}$$

where $\|\cdot\|_2$ is the L_2 (also called Frobenius) norm of a matrix (see Equation 9).

The minimisation problem is convex in D (keeping R fixed) and it is convex in R (keeping D fixed) but not simultaneously in D and in R. So it cannot be hoped to find with certainty the global minimum (the optimal D and R) but only a local minimum. Gradient descent methods could be applied but convergence can be slow. So, multiplicative methods have been developed. These methods usually start from random guesses for D and R and iteratively update D and R using update formulae. For the traditional NMF [2], these update equations can be written as:

$$\begin{aligned} R_{ij} &\leftarrow\!\rightarrow R_{ij} \frac{(D^\top X)_{ij}}{(D^\top DR)_{ij}} \\ D_{ij} &\leftarrow\!\rightarrow D_{ij} \frac{(XR^\top)_{ij}}{(DRR^\top)_{ij}} \end{aligned} \tag{15}$$

3.2 $L_{2,1}$ norm NMF

The$L_{2,1}$ norm NMF algorithm seeks to find, given a non-negative matrix X, two non-negative matrices D and R such that $X \approx DR$:

$$D, R = \underset{D \geq 0, R \geq 0}{\arg\min} \|X - D\,R\|_{2,1} \tag{16}$$

where $\|\cdot\|_{2,1}$ is the $L_{2,1}$ norm of a matrix (see Equation 12).

The minimisation problem is convex in D (keeping R fixed) and it is convex in R (keeping D fixed) but not simultaneously in D and in R. So it cannot be hoped to find with certainty the global minimum (the optimal D and R) but only a local minimum. Gradient descent methods could be applied but convergence can be slow. So, multiplicative methods have been developed. These methods usually start from random guesses for D and R and iteratively update D and R using update formulae. For the $L_{2,1}$ norm NMF [5], these update equations can be written as:

$$\begin{aligned} D_{ij} &\leftarrow\!\rightarrow D_{ij} \frac{(X\Delta R^\top)_{ij}}{(DR\Delta R^\top)_{ij}} \\ R_{ij} &\leftarrow\!\rightarrow R_{ij} \frac{(D^\top X\Delta)_{ij}}{(D^\top DR\Delta)_{ij}} \end{aligned} \tag{17}$$

where the matrix Δ is diagonal with diagonal elements:

$$\Delta_{jj} = \frac{1}{\sqrt{\sum_i (X - D\,R)_{ij}^2}} \tag{18}$$

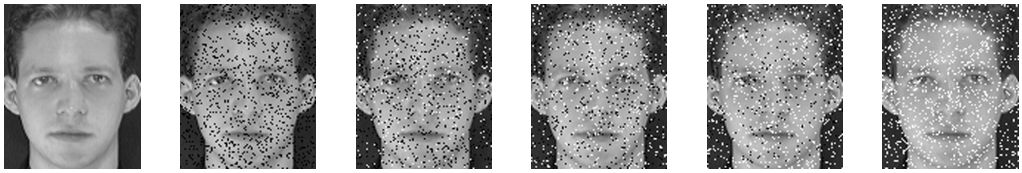
3.3 Noise generation

The robustness of NMF algorithms can be explored by taking a clean dataset X, contaminating it with random noise, learning the dictionary D from the contaminated dataset and studying how well it can reconstruct images.

A type of noise that is commonly used to contaminate datasets of images is the salt and pepper noise: randomly selected pixels are replaced with a black (the *pepper*) or a white (the *salt*) pixel. Two parameters define this noise: p, the fraction of pixels to be replaced, and r the fraction of white pixels among the pixels to be contaminated (the rest of the pixels are set to be black). We will be using this type of noise for all the experiments. In Figure 1 we show the effect of p on the level of noise added to a clean image. Visually, when p is around 0.3, it becomes hard to make out details of the faces. In Figure 2 we show the effect of r on the type of noise: even if the level of noise is the same, the amount of white and black pixels has a visual effect and, as we will see, an effect on how well the NMF algorithms run.

(a) clean (b) p=0.05 r=0.5 (c) p=0.1 r =0.5 (d) p=0.2 r=0.5 (e) p=0.4 r=0.5

Figure 1: Salt and Pepper Noise: effect of p



(a) clean (b) p=0.1 r=0 (c) p=0.1 r =0.25 (d) p=0.1 r=0.5 (e) p=0.1 r=0.75 (f) p=0.10 r=1

Figure 2: Salt and Pepper Noise: effect of r

Another type of noise that is also commonly used to contaminate datasets is occlusion noise. We implemented two types. In one, a block of contiguous pixels of rectangular shape is replaced by black pixels. In the other, horizontal and vertical thin lines are placed randomly on the image.

4 Experiments

We conducted experiments using two datasets of images: the ORL dataset and the Extended YaleB dataset. The ORL dataset contains 400 images of 40 distinct subjects with various level of illumination, different facial expressions, with/without glasses. All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position.The Extended YaleB dataset contains contains 2414 images of 38 subjects under 9 poses and 64 illumination conditions. Images within the same dataset have the same dimensions: 92x112 pixels in the ORL dataset and 168x192 pixels in the Extended YaleB dataset. Due to the computational cost of performing the NMF factorisation with a large dataset of large images, all images, in both datasets, were resized to be 30x37 pixels.

To produce the final matrix X used as input to the NMF algorithm, each of the images in the same dataset was firstly flattened by concatenating all rows in the corresponding matrix of pixels, so to produce a column vector of length $f = 1100$ (=30x37). This quantity f is the number of features. Then, all column vectors representing the images in the dataset were concatenated into the final matrix $X \in \mathbb{R}^{f \cdot n}$. This quantity n is the number of examples ($n = 400$ for the ORL dataset and $n = 2414$ for the Extended YaleB dataset).

We conducted experiments using the two NMF algorithms described above: the L_2 norm NMF and the $L_{2,1}$ norm NMF. The first is the original NMF algorithm introduced when the NMF was originally invented. The second is theoretically a more robust NMF algorithm based on a norm $\|\cdot\|$ less prone to be influenced by outliers. We mainly concentrated on investigating the impact of the salt and pepper noise on the NMF process. We also did some experimentation using two types of occlusion: the rectangular and the grid.

To assess the impact of the noise on the performance of the NMF algorithm, we considered the *reconstruction error*. The *clean* dataset X is firstly contaminated by noise to produce a *noisy* dataset $\tilde{X}$. The dictionary $\tilde{D}$ and the representation $\tilde{R}$ are then learnt from the noisy dataset $\tilde{X}$. Finally the reconstruction error is defined as:

$$\frac{\left\| X - \tilde{D}\,\tilde{R} \right\|}{\|X\|} \tag{19}$$

The interpretation of this error is that we usually collect noisy data and we are hoping to infer something useful about the clean data (that are not usually available).

Instead of using the whole dataset X, before performing any of the experiments, we extracted a random subset $\bar{X}$ (without replacement) from the whole dataset X. We repeated each experiment 5 times (drawing different subsets) in order to have a measure of the variability of the results.

In the first experiment we investigate the effect of k, the number of features in the dictionary D, on the reconstruction of images given clean data. More specifically, a dataset X of images is first factored into a dictionary D and a representation R. Then we analyse some images by comparing the original image with $D\,R$, the reconstructed image. The dimensions of D and R are $\mathbb{R}^{f \cdot k}$ and $\mathbb{R}^{k \cdot n}$. The number n of examples in the dataset and the number of features f in each example is fixed and given by the charateristics of the data under examination. The number of features k used by the NMF algorithm is instead an hyper-parameter that can be selected. In Figure 3 and Figure 4, we present a series of noiseless images (first row) and then the reconstructed images $D\,R$ using $k = 20, 50, 75, 100$ and two different flavours of NMF algorithms. By visually inspecting the reconstructed images, we have the impression that with a higher number of dimensions, the NMF algorithm is able to come closer and closer to the original image. Figure 5 shows the reconstruction error for this case and we see that indeed the reconstruction error is decreasing when the number of components k increases. Results are also presented in tabular form in Table 1.

	k=1	k=10	k=20	k=50	k=75	k=100
Mean (L2)	0.2924	0.1953	0.1679	0.1309	0.1143	0.1030
Sd (L2)	4.388e-4	5.490e-4	5.389e-4	4.132e-4	3.425e-4	4.910e-4
Mean (L21)	0.2932	0.1952	0.1684	0.1317	0.1150	0.1030
Sd (L21)	7.719e-4	3.443e-4	3.402e-4	4.218e-4	3.889e-4	2.324e-4

Table 1: Mean and Standard Deviation of Reconstruction Error, trained and evaluated on noiseless data

In the second experiment we investigate the effect of amount p of noise on the data. As in the first experiment, we consider a dataset $\tilde{X}$ and factored in into a dictionary $\tilde{D}$ and a representation $\tilde{R}$. This time the dataset $\tilde{X}$ is a contaminated by noise version of the clean dataset X. Using the dictionary $\tilde{D}$, we reconstruct the representation R of the clean dataset X and we display the reconstruction using different values of p. Figure 6 presents clean images in the first row and then, in pairs, contaminated images and their reconstructions (reconstructed using a dictionary D lernt from equivalent noise). We can see that the reconstruction of the clean images degrades with the increasing level of noise. This is probably not too surprising because we expect the reconstruction to become harder in the presence of a lot of noise. We can examine the reconstruction error in Figure 7 and Figure 8 for varying levels of p and varying levels of k. Overall, the reconstruction error increases with p as we have observed. More interestingly, for a fixed value of p, the reconstruction error worsen with increasing k. We attribute that to the fact the with more available dimensions k, the dictionary not only learns features of the images but its ability to discriminate noise from actual data becomes weaker. Some of the characteristic of the noise are incorporated into the dictionary D.

In the third experiment we investigate the effect of the ratio r of white/black noise on the data, for a fix level of p. As in the first experiment, we consider a dataset $\tilde{X}$ and factored in into a dictionary $\tilde{D}$ and a representation $\tilde{R}$. This time the dataset $\tilde{X}$ is a contaminated by a noisy version of the clean dataset X. Using the dictionary $\tilde{D}$, we reconstruct the representation R of the clean dataset X and we display the reconstruction using different values of r. Figure 9 presents clean images in the first row and then, in pairs, contaminated images and their reconstructions (again based on a dictionary D trained on equivalent noise). We can see that the reconstruction of the clean images does not seem to degrade noticeably if there is more pepper noise or salt noise. To assess the validity of this observation, we can examine the reconstruction error in Figure 10 and Figure 11 for varying levels of r and varying levels of k. Overall, the reconstruction error increases with k as we have observed. More interestingly, for a fixed value of p and k, the reconstruction error is slightly worse when we have more salt noise than pepper noise. We hypothesise this is because the average pixel in the image is closer to pepper than salt.

In the fourth experiment, we investigate the robustness of the NMF algorithms to the presence of noise. We consider both the clean dataset X and its factorisation D and R and the contaminated dataset $\tilde{X}$ and its dictionary $\tilde{D}$. Using the dictionary $\tilde{D}$, we compute the representation $\tilde{R}$ of the clean dataset. In Figure 12 and Figure 13, we present on the first row examples of clean images and in the second row noisy images. The third row is the representation $\tilde{D}\,\tilde{R}$ and the fourth row is the representation $D\,R$. So $\tilde{R}$ is the representation we get from noisy data and r is the best representation that we could get if we had access to clean data. The fifth row is the difference of the two reconstructions. The level of noise is the same in all sets of images but the number of dimension k is increasing. Comparing the fifth row of the various set of images allows gathering information about how close the reconstruction from noisy data comes to the reconstruction from noiseless data and how it correlates with facial features. We can observe that for k=20, there are some hints of face features in the difference. The number of dimension is too small to have a representation capability flexible enough to well reconstruct the images. The difference becomes more and more uniform going to $k = 50$ and $k = 100$ where we hit a soft spot: there are differences but they are randomly distributed. Increasing k, the reconstruction worsens because the NMF starts to learn the noise besides the image.

Finally, we can comment on the sparsity of the representation generated by the NMF algorithm and on the reconstruction in the presence of occlusions. Figure 14 presents examples of a visual representation of the matrix R. We can see that the matrices R are sparse, with some large elements

and many small elements. Sparsity is a very useful characteristics and it endows the MNF algorithms with very effective dimensionality reduction properties. In Figure 15, we present reconstruction results in presence of salt and pepper noise and occlusions.

5 Conclusion

NMF has some interesting effects when considering robustness to noise. While in theory other norm methods (specifically $L_{2,1}$) seem like they should have better results, there is marginal benefit.
In future, we would like to experiment with faces that the network has not seen before to ensure that the NMF model is not simply learning faces and representing only the faces it has seen, ensuring the model generalises well. We have identified this as a major limitation of our design. We would also like to experiment further with the Yale dataset, as faces are aligned more deliberately, which may give the NMF algorithm an edge.

References

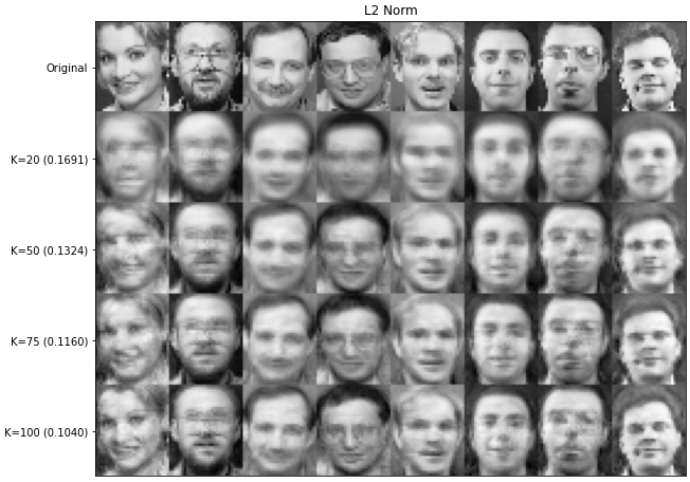
[1] Daniel D. Lee & H. Sebastian Seung (1999) Learning the parts of objects by non-negative matrix factorization *Nature* 401 (6755), pp. 788–791

[2] Daniel D. Lee & H. Sebastian Seung (2001) Algorithms for Non-negative Matrix Factorization *Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference*, MIT Press, pp. 556–562

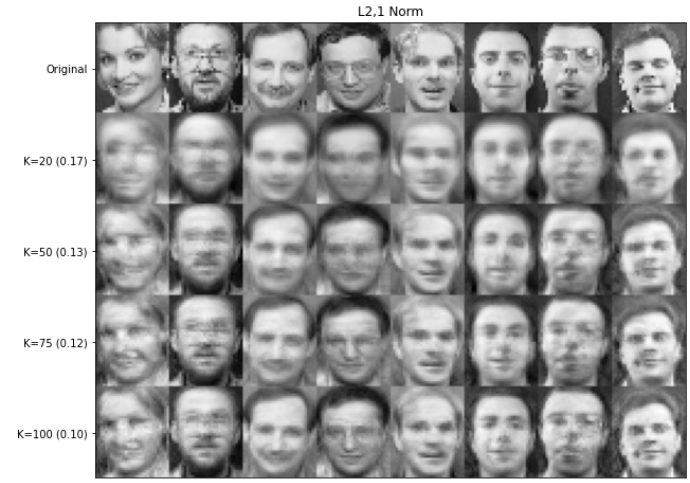
[3] Liu, W., Zheng, N. & You, Q. (2006) Nonnegative matrix factorization and its applications in pattern recognition *Chinese Science Bulletin* 51, 7–18

[4] N. Guan, T. Liu, Y. Zhang, D. Tao & L. S. Davis (2019) Truncated Cauchy Non-Negative Matrix Factorization *IEEE Transactions on Pattern Analysis and Machine Intelligence* vol. 41, no. 1, pp. 246-259

[5] D. Kong, C. Ding & H. Huang (2011) Robust nonnegative matrix factorization using L21-norm *CIKM '11: Proceedings of the 20th ACM international conference on Information and knowledge management*, pp. 673–682



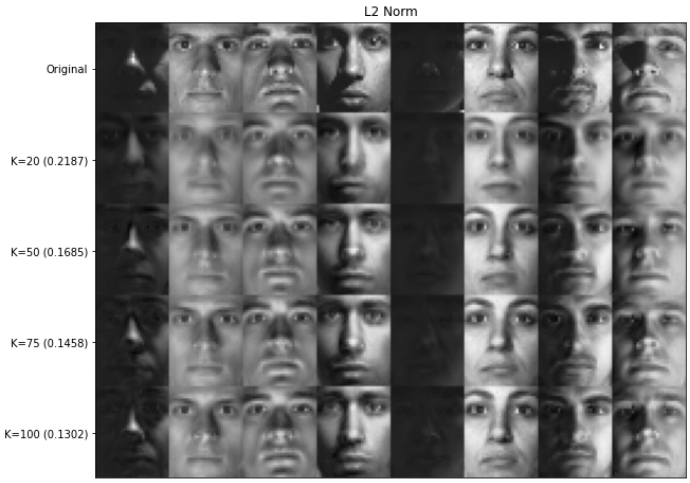
(a) L_2 norm NMF: effect of k (using noiseless data)



(b) L_{21} norm NMF: effect of k (using noiseless data)

Figure 3: Noiseless data: effect of k (ORL dataset)

(a) L_2 norm NMF: effect of k (using noiseless data)



(b) L_{21} norm NMF: effect of k (using noiseless data)

Figure 4: Noiseless data: effect of k (YALE dataset)

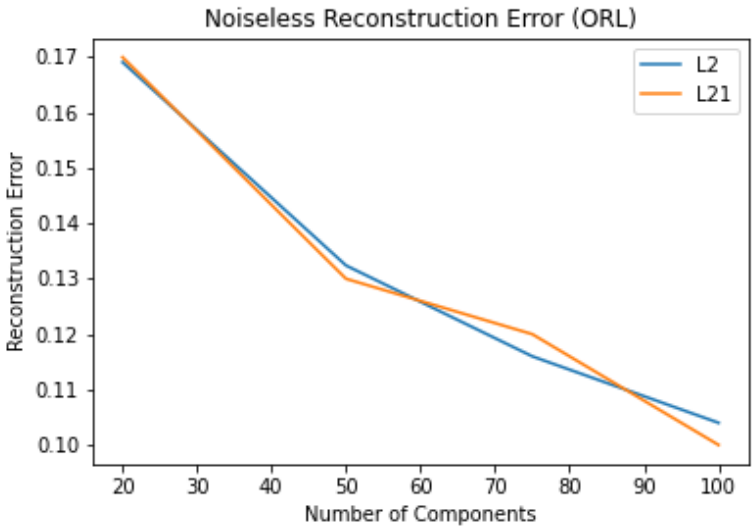
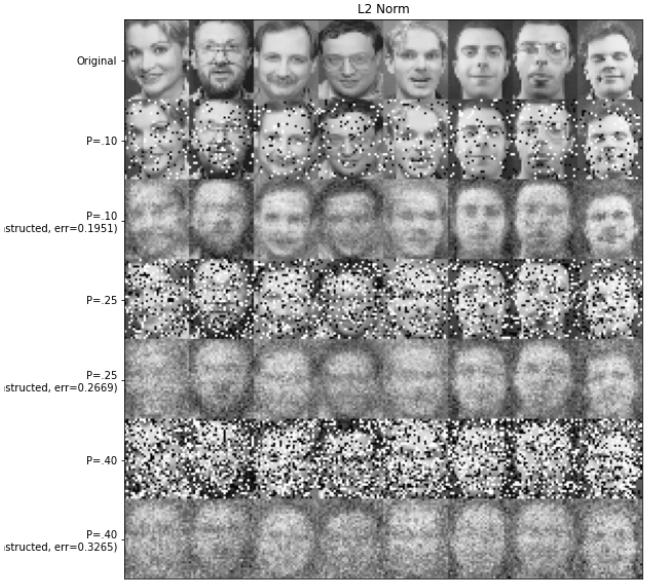
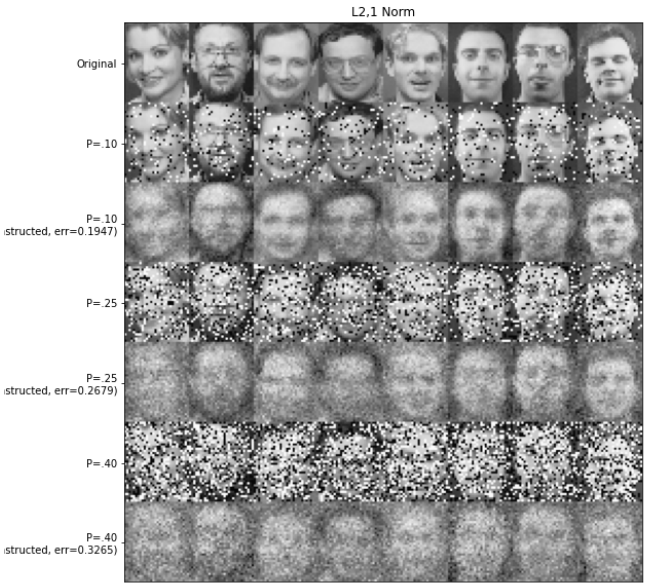


Figure 5: Noiseless data: effect of k on reconstruction error



(a) L_2 norm NMF: effect of p (using noisy data)



(b) L_{21} norm NMF: effect of p (using noisy data)

Figure 6: Noisy data: effect of p

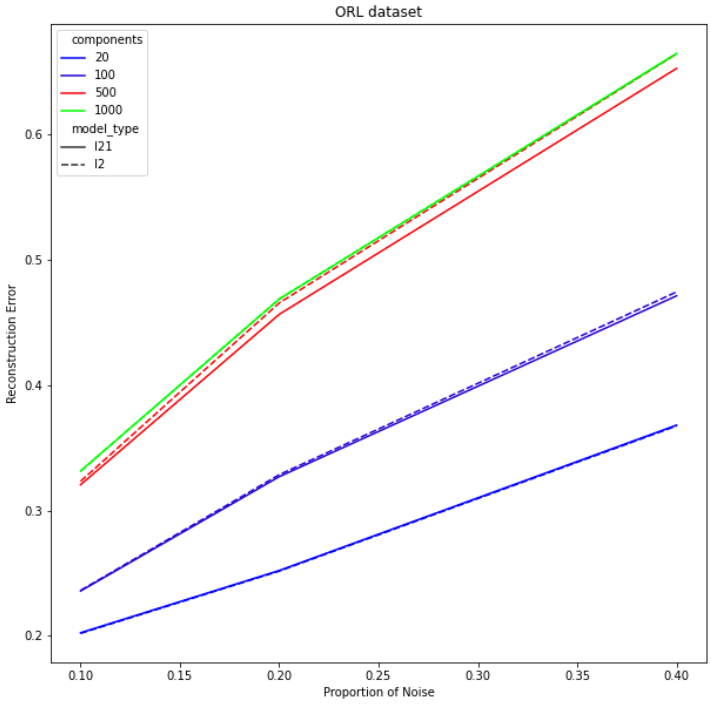


Figure 7: Noisy data: effect of p on reconstruction error

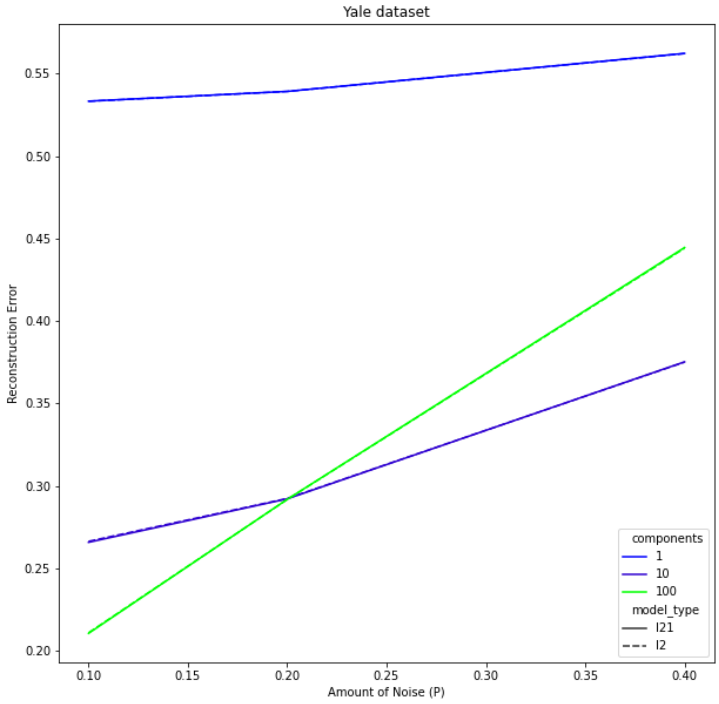
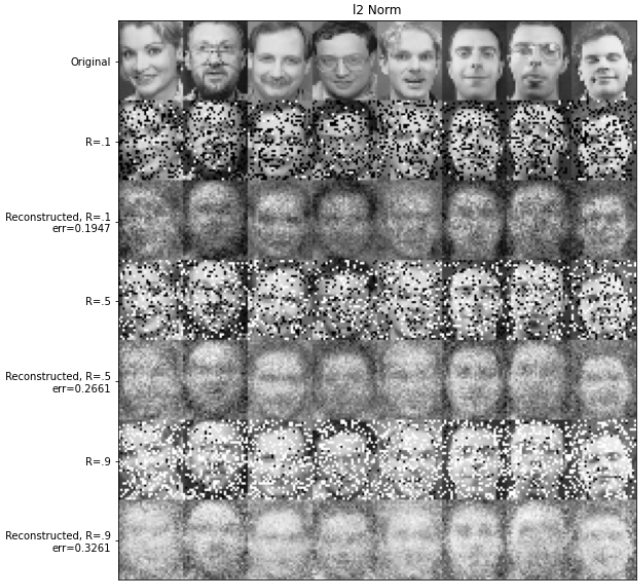
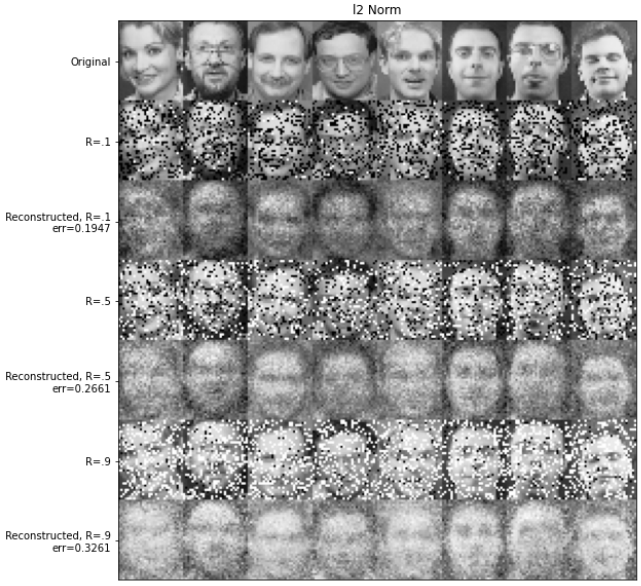


Figure 8: Noisy data: effect of p on reconstruction error



(a) L_2 norm NMF: effect of r (using noisy data)



(b) L_{21} norm NMF: effect of r (using noisy data)

Figure 9: Noisy data: effect of r

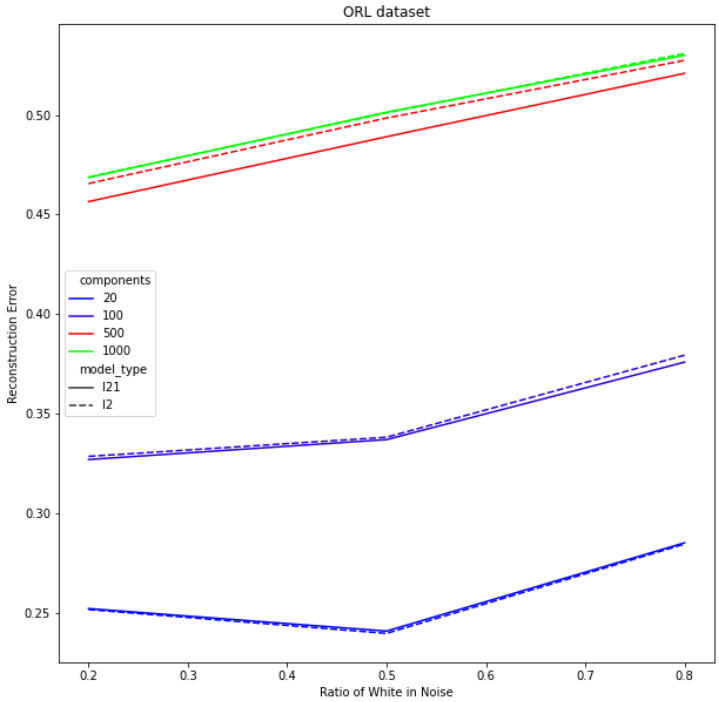


Figure 10: Noisy data: effect of r on reconstruction error

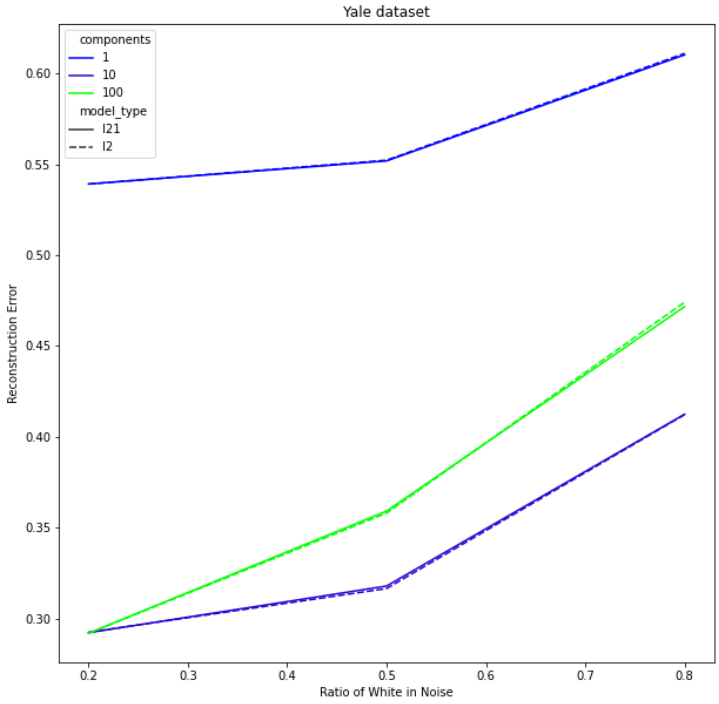
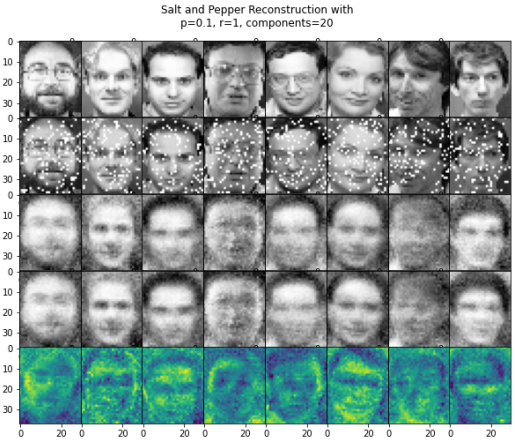
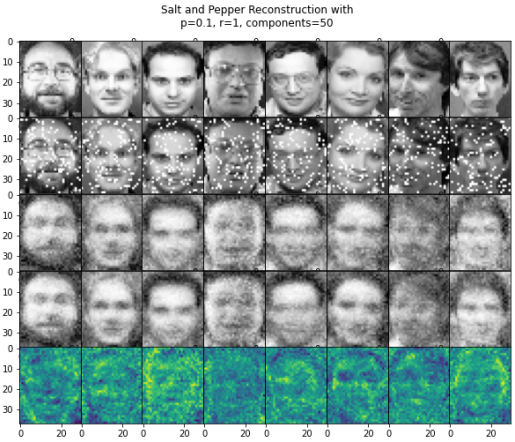


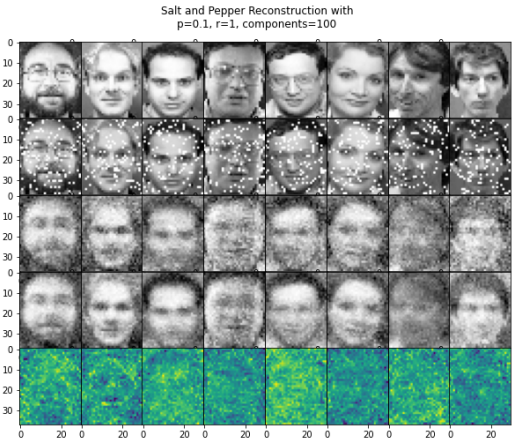
Figure 11: Noisy data: effect of r on reconstruction error (YALE dataset)



(a) p=0.1 r=1 k=20

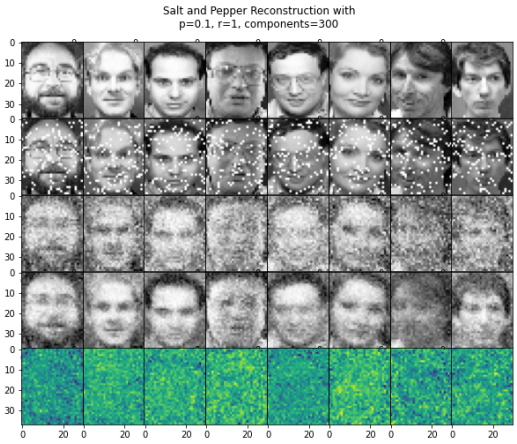


(b) p=0.1 r=1 k=50

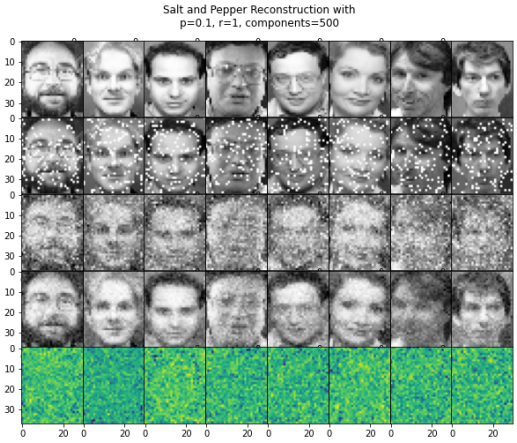


(c) p=0.1 r=1 k=100

Figure 12: Salt and Pepper Noise: effect of k

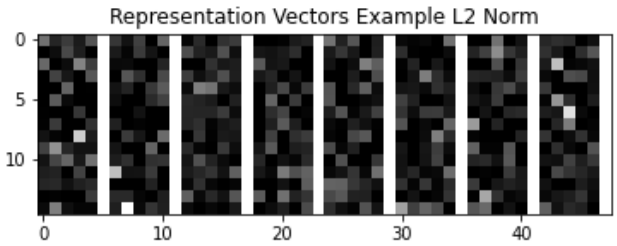


(a) p=0.1 r=1 k=300

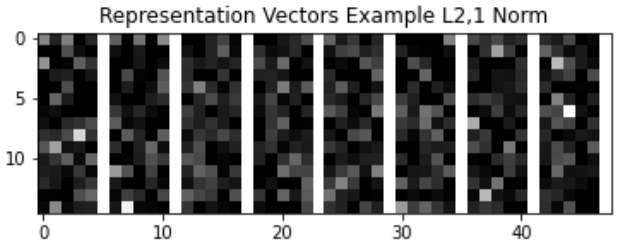


(b) p=0.1 r=1 k=500

Figure 13: Salt and Pepper Noise: effect of k



(a) L_2 norm NMF: sparsity of the representation



(b) L_{21} norm NMF: sparsity of the representation

Figure 14: Sparsity of the representation

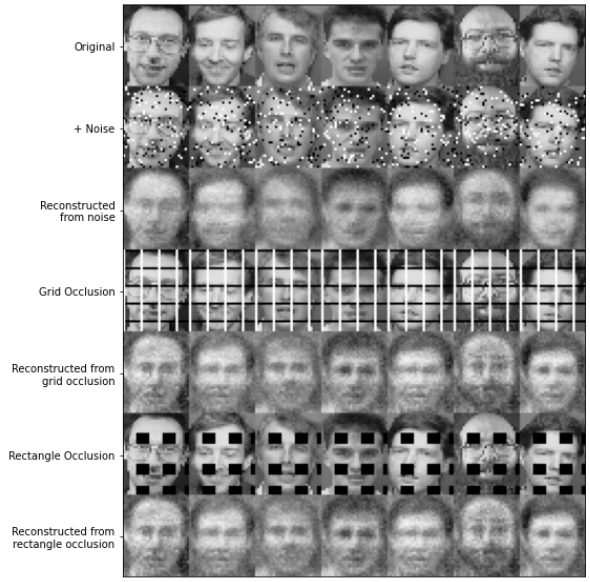


Figure 15: Noisy data and occlusions