COMP5328 - Advanced Machine Learning

Assignment 1

Due: 22/10/2020, 11:59PM

This assignment is to be completed in groups of 2 to 3 students. It is worth 25% of your total mark.

1 Objective

The objective of this assignment is to implement Non-negative Matrix Factorization (NMF) algorithms and analyze the robustness of NMF algorithms when the dataset is contaminated by large magnitude noise or corruption. More specifically, you should implement at least two NMF algorithms and compare their robustness.

2 Instructions

2.1 Dataset description

In this assignment, you need to apply NMF algorithms on two real-world face image datasets: (1) ORL dataset[1]; (2) Extended YaleB dataset[2].

- **ORL dataset**: it contains 400 images of 40 distinct subjects (i.e., 10 images per subject). For some subjects, the images were taken at different times, varying the lighting, facial expressions and facial details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position. All images are cropped and resized to 92×112 pixels.
- **Extended YaleB dataset**: it contains 2414 images of 38 subjects under 9 poses and 64 illumination conditions. All images are manually aligned, cropped, and then resized to 168×192 pixels.

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

Note: we provide a tutorial for this assignment, which contains example code for loading a dataset to numpy array. Please find more details in **assignment1.ipynb**.

2.2 Assignment tasks

1. You need to implement at least two Non-negative Matrix Factorization (NMF) algorithms:
   - You should implement at least two NMF algorithms with at least one not taught in this course (e.g., L_1-Norm Based NMF, Hypersurface Cost Based NMF, L_1-Norm Regularized Robust NMF, and $L_{2,1}$-Norm Based NMF).
   - For each algorithm, you need to describe the definition of objective function as well as the optimization methods used in your implementation.
2. You need to analyze the robustness of each algorithm on two datasets:
   - You are allowed to design your own data preprocessing method (if necessary).
   - You need to use salt and pepper noise similar to those shown in Figure 1. The noise is generated by randomly change some pixel values in a image to be either 255 (white) or 0 (black). You should use a hyper-parameter p to control the noise level. For example if $p = 0.4$, then 40% of the pixel values in a image should be modified to either 255 (white) or 0 (black). You should also use a hyper-parameter r to control the ratio of being 255 (white) to the number of modified pixel values. For example, if $r = 0.3$, then 30% of the modified pixel values should be 255 (white). You can choose your own value for both p and r. You are also encouraged to design your own noise other than salt and pepper noise.
   - You need to demonstrate each type of noise used in your experiment (show the original image as well as the image contaminated by noise).
   - You should carefully choose the NMF algorithms and design experiment settings to clearly show the different robustness of the algorithms you have implemented.
3. You **are only allowed to** use the python build-in library, **numpy** and **scipy** (if necessary) to implement NMF algorithms.

2.3 Programming and External Libraries

This assignment is **required** to be finished by **Python3**. When you implement NMF algorithms, you are **not allowed** to use external libraries which contains

NMF implementations, such as **scikit-learn**, and **Nimfa** (i.e., you have to implement the NMF algorithms by yourself). You are allowed to use **scikit-learn** for evaluation only (please find more details in **assignment1.ipynb**). If you have any ambiguity whether you can use a particular library or a function, please post on canvas under the "Assignment 1" thread.

2.4 Evaluate metrics

To compare the performance and robustness of different NMF algorithms, we provide three evaluation metrics: (1) Relative Reconstruction Errors; (2) Average Accuracy (**optional**); (3) Normalized Mutual Information (**optional**). **For all experiments, you need to use at least one metric, i.e., Relative Reconstruction Errors**. You are encouraged to use the other two metrics, i.e., Average Accuracy and Normalized Mutual Information.

- **Relative Reconstruction Errors (RRE)**: let V denote the contaminated dataset (by adding noise), and $\hat{V}$ denote the clean dataset. Let W and H denote the factorization results on V, the **relative reconstruction errors** then can be defined as follows:

$$RRE = \frac{\|\hat{V} - WH\|_F}{\|\hat{V}\|_F}. \tag{1}$$

- **Average Accuracy:** Let W and H denote the factorization results on V, you need to perform some clustering algorithms (i.e., K-means) with **num_clusters** equal to **num_classes**. Each example is assigned with the cluster label (please find more details in **assignment1.ipynb**). Lastly, you can evaluate the accuracy of predictions Y_{pred} as follows:

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

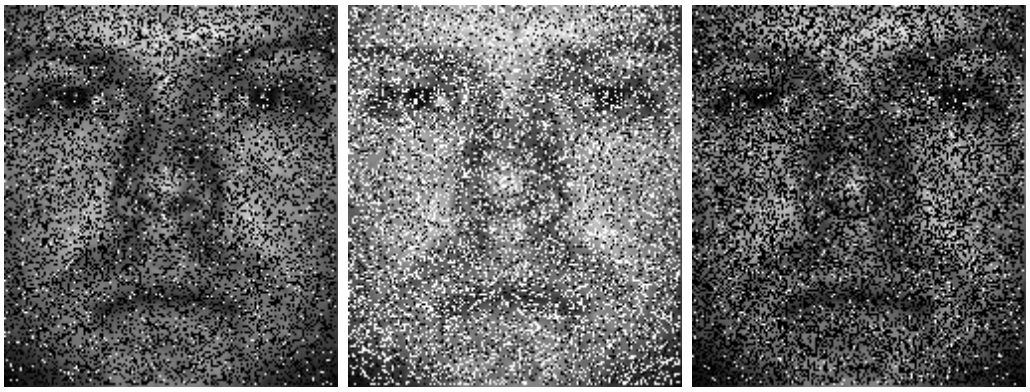
- **Normalized Mutual Information (NMI):**

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

  where $I(\cdot, \cdot)$ is mutual information and $H(\cdot)$ is entropy.

Note: we expect you to have a rigorous performance evaluation. To provide an estimate of the performance of the algorithms in the report, you can repeat multiple times (e.g., 5 times) for each experiment by randomly sampling 90% data from the whole dataset, and average the metrics on different subset. You are also required to report the standard deviations.

(a) $p = 0$ (Original Image) (b) $p = 0.2, r = 0.1$ (c) $p = 0.2, r = 0.7$



(d) $p = 0.4, r = 0.1$ (e) $p = 0.4, r = 0.7$ (f) $p = 0.6, r = 0.1$

Figure 1: An example face image (a) and its occluded versions (b, c, d, e, and f) generated by adding salt and pepper noise with different values of hyper-parameters p and r. Note that the hyper-parameter p is used to control the noise level (i.e., number of the modified pixels/total number of pixels). The hyper-parameter r is used to control the ratio of being 255 (white) to the number of modified pixel values.

3 Report

The report should be organized similar to research papers, and should contain the following sections:

- In **abstract**, you should briefly introduce the topic of this assignment and describe the organization of your report.
- In **introduction**, you should first introduce the main idea of NMF as well as its applications. You should then give an overview of the methods you want to use.

- In **related work**, you are expected to review the main idea of related NMF algorithms (including their advantages and disadvantages).
- In **methods**, you should describe the details of your method (including the definition of objective functions as well as optimization steps). You should also describe your choices of noise and you are encouraged to explain the robustness of each algorithm from theoretical view.
- In **experiment**, firstly, you should introduce the experimental setup (e.g., datasets, algorithms, and noise used in your experiment for comparison). Second, you should show the experimental results and give some comments.
- In **conclusion**, you should summarize your results and discuss your insights for future work.
- In **reference**, you should list all references cited in your report and formatted all references in a consistent way.

The layout of the report:

- Font: Times New Roman; Title: font size 14; Body: font size 12
- Length: Ideally 10 to 15 pages - maximum 20 pages

Note: You are encouraged to use LaTeX. Optionally, a MS-Word template is provided.

4 Submissions

The submission contains two parts: **source code** and **report**. Detailed instructions are as follows:

1. Go to Canvas and upload the following files/folders compressed together as a zip file.
   (a) report (a pdf file): the report should include each member's details (student id and name).
   (b) code (a folder)
      i. algorithm (a sub-folder): your code could be multiple files.
      ii. data (an empty sub-folder): although two datasets should be inside the data folder, please **do not** include them in the zip file. We will copy two datasets to the data folder when we test the code.

2. Only one student needs to submit the zip file which must be named as student ID numbers of all group members separated by underscores. E.g. "xxxxxxxx_xxxxxxxx_xxxxxxxx_.zip".

3. Your submission should include the report and the code. A plagiarism checker will be used.

4. You need to clearly provide instructions on how to run your code in the appendix of the report.

5. Indicate the contribution of each group member.

6. A penalty of minus 5 (5%) marks per each day after due (email late submissions to TA and confirm late submission dates with TA). Maximum delay is 10 days, after that assignments will not be accepted.

5 Marking scheme

Category	Criterion	Marks	Comments
Report [80]	Abstract [3] •problem, methods, organization. Introduction [5] •What is the problem you intend to solve? •Why is this problem important? Previous work [6] •Previous relevant methods used in literature? Methods [25] •Pre-processing (if any) •NMF Algorithm's formulation. •Noise choice and description. Experiments and Discussions [25] •Experiments, comparisons and evaluation •Extensive analysis and discussion of results •Relevant personal reflection Conclusions and Future work [3] •Meaningful conclusions based on results •Meaningful future work suggested Presentation [5] •Grammatical sentences, no spelling mistakes •Good structure and layout, consistent formatting •Appropriate citation and referencing •Use graphs and tables to summarize data Other [8] •At the discretion of the marker: for impressing the marker, excelling expectation, etc. Examples include clear presentation, well-designed experiment, fast code, etc.		

Category	Criterion	Marks	Comments
Code [20]	•Code runs within a feasible time •Well organized, commented and documented		
Penalties [−]	•Badly written code: [−20] •Not including instructions on how to run your code: [−20]		

Note: Marks for each category is indicated in square brackets. The minimum mark for the assignment will be 0 (zero).