Investigating the feature extraction capabilities of nonnegative matrix factorisation algorithms for black-and-white images

How Hui Liew[1,*], *Wei Shean* Ng[1,**], and *Huey Voon* Chen[1,***]

[1]Department of Mathematical and Actuarial Sciences,
Universiti Tunku Abdul Rahman (Sungai Long Campus),
Jalan Sungai Long, Bandar Sungai Long, 43000 Kajang,
Selangor Darul Ehsan, Malaysia

Abstract. Nonnegative matrix factorisation (NMF) is a class of matrix factorisation methods to approximate a nonnegative matrix as a product of two nonnegative matrices. To derive NMF algorithms, the optimisation problems for NMF are developed and the divergence used in the optimisation problems can have many forms. The β-divergence is the most popular and is used in this research. The NMF algorithms derived from the β-divergence have a few hyperparameters including the rank and the initial conditions. This paper surveyed on the software implementations of the NMF algorithms and then applied the open source software implementations of Frobenius norm based NMF algorithm, KL divergence based NMF algorithm and binary matrix factorisation (BMF) with fixed ranks to three classes of black-and-white images. For black-and-white images with a lot of common features (like MNIST), KL divergence NMF with appropriate initial guess is empirically found to be best NMF algorithm for black-and-white image feature extraction compare to other NMF algorithms. All NMF algorithms for data with little to no common features are useful in generating feature images which can be used to inspire art design as well as in the realm of computer vision.

1 Introduction

Nonnegative matrices arise in many applications such as data mining [1], text mining [2], document clustering [3], pattern recognition [4], signal filtering [5], feature extraction [6], blind source separation [7, 8], gene function prediction [9], classification [10], etc. Various nonnegative matrix factorisation (NMF) algorithms have been developed for handling the stated different scenarios.

This research aims (i) to understand the mathematical formulation of NMF, (ii) to investigate the software implementations of various NMF algorithms, and (iii) to investigate the image features that can be extracted from NMF algorithms empirically on some black-and-white images (which can be represented as Boolean matrices).

*e-mail: liewhh@utar.edu.my
**e-mail: ngws@utar.edu.my
***e-mail: chenhv@utar.edu.my

The literature review is conducted in Section 2. The mathematical formulation for NMF algorithms is given in Section 3. The generalisation of NMF algorithms has led to the classification of NMF algorithms which is summarised in Section 4. Software is becoming increasingly important in modern research and a survey of open source software libraries for NMF is given in Section 5. Since NMF algorithms are derived from an optimisation problem, they are dependent on the initialisation which is mentioned in Section 6. Our main results on the study of black-and-white images are summarised in Section 7.

2 Literature Review

Standard numerical methods for matrix factorisation such as LU (lower–upper decomposition), QR decomposition (Q is an orthogonal matrix, R is an upper triangular matrix), SVD (singular value decomposition) [11] usually lead to factored matrices with a combination of positive and negative values. The negative values usually do not have appropriate meaning when X is used to denote a collection of normalised gray-scale images or a collection of word representations of documents.

Nonnegative matrix factorisation (NMF) is a class of *approximation matrix factorisation* methods that preserve the nonnegativity in the factored matrices by approximation rather than equality as in LU, QR and SVD:

$$X \approx WH. \tag{1}$$

Here, $W \geq 0$ is an $n \times k$-matrix, $H \geq 0$ is an $k \times p$-matrix and k is usually chosen such that $k(n + p)$ is much smaller than np [12].

Unlike the canonical matrix factorisation methods such as LU, QR and SVD, NMF is not unique because X can be decomposed into different products

$$X \approx \widetilde{W}\widetilde{H}$$

where $\widetilde{W} = WQ$ and $\widetilde{H} = Q^{-1}H$, Q is a monomial matrix (also known as the generalised permutation matrix) with all nonnegative elements [13].

The development of nonnegative matrix factorisations (NMF) algorithms has more than thirty years of history. The first characterisation of NMF algorithm starts with the study of nonnegative rank factorisation [14] in 1981. It was not until 1999 that NMF became popular due to its ability to extract features from grey image data and text data [1].

Since then, NMF (1) is generalised to a constrained optimisation problem [15]:

$$\min_{W,H} D(X\|WH). \tag{2}$$

The divergence $D(X\|Y)$ measures the difference matrices X and Y and can be regarded as a generalisation of distance function.

Various NMF algorithms are proposed by varying the divergence $D(X\|Y)$. Popular divergences from statistics and information theory including the Bregman divergence [16], the Amari's α-divergences [17], the γ-divergences [18], the Ψ-divergence [19], etc. have been studied and NMF algorithms have been proposed and the convergences have been investigated.

In 2009, NMF was well established with the publication of the book [20] on NMF and Nonnegative Tensor Factorisation (NTF) algorithms. In 2013, a survey paper on various generalisations of NMF are published [21]. The development of NMF algorithms continues in various generalisations in algorithmic aspect such as block coordinate descent [22] until the publication of another textbook in 2020 [23]. The recent development of NMF is focused on the applications of NMF to various scientific and engineering fields [7, 8, 24].

3 Mathematical Formulation

Consider a nonnegative matrix X of dimension $n \times p$. The elements of the matrix X are $x_{ij} \geq 0$ for $i = 1, ..., n$ and $j = 1, ..., p$ and we usually denote a nonnegative matrix using the notation $X \geq 0$.

To extract features from black-and-white images, we will apply the NMF algorithms based on a subclass of the Bregman divergence called the *β-divergence* [7, 25–27] which are closely related to the Tweedie distribution [28]:

$$D_\beta(X||Y) = \frac{1}{\beta(\beta-1)} \sum_{i,j} (X_{ij}^\beta - \beta X_{ij} Y_{ij}^{\beta-1} + (\beta-1) Y_{ij}^\beta) \tag{3}$$

where $Y = WH, \beta \in \mathbb{R} \setminus \{0, 1\}$.

When $\beta = 2$, the loss function (3) becomes the squared Frobenius norm:

$$D_2(X||Y) = \frac{1}{2}||X - Y||_F^2 = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{p} (x_{ij} - y_{ij})^2. \tag{4}$$

Taking limit $\beta \to 1$, the (generalised) Kullback-Leibler (KL) divergence (also known as the I-divergence) is obtained:

$$D_1(X||Y) = \sum_{i,j} (X_{ij} \ln \frac{X_{ij}}{Y_{ij}} - X_{ij} + Y_{ij}). \tag{5}$$

Taking limit $\beta \to 0$, the Itakura-Saito (IS) divergence is obtained:

$$D_0(X||Y) = \sum_{i,j} (\frac{X_{ij}}{Y_{ij}} - \ln \frac{X_{ij}}{Y_{ij}} - 1). \tag{6}$$

The squared Frobenius norm (4) can be regarded as the Euclidean distance for two matrices and the Kullback-Leibler divergence (5) will try to approximate the statistical distribution of the matrix WH to the matrix X, therefore we will limit ourselves to the NMF algorithms based on the two divergences (4) and (5).

4 Classification of NMFs

To adapt to the handling of various kind of information (e.g. texts, images, audio signals, etc.) processing problems, the basic NMF based on the optimisation problem (2) are extended in various ways and they are categorised as follows by the survey paper [21] and the NMF and Nonnegative Tensor Factorisation (NTF) textbooks [20, 23]:

(i) Basic NMF (BNMF): A matrix factorisation with nonnegativity constraints only and is given by (1) or (2).

(ii) Constraint NMF (CNMF): An NMF with general regularisation constraints $J_1(\cdot)$ and $J_2(\cdot)$:

$$D(X||WH) + \alpha_1 J_1(W) + \alpha_2 J_2(H). \tag{7}$$

The constraints can be further classified into sparsity constraints, orthogonality constraints, discriminant constraints and manifold (usually graph-regularised) constraints.

(iii) Structured NMF (SNMF): The NMF with extra matrix structures such as extra weights, convolutive forms, etc.

(iv) Generalised NMF (GNMF): An extension of NMF to tensors, kernels, etc.

Gradient descent is a typical method for finding the minimum in an optimisation problem. However, it is found to converge slowly [29]. Several faster optimisation algorithms have been proposed to obtain a nonnegative matrix factorisation. For the Frobenius norm (4), we have many algorithms such as multiplicative update (MU), alternative least square (ALS), etc. to obtain a nonnegative matrix factorisation. However, there are fewer algorithms to find the nonnegative matrix factorisation for KL divergence (5) and even less for IS divergence (6)). This is due to the fact that both the Frobenius norm (4) and the KL divergence (5) are convex functions with respect to W (when H is fixed) and H (when W is fixed), this allows the formation of the alternating update framework which includes MU and ALS as special cases. The algorithms are more complex with the addition of more constraints or when NMF is extended to tensors or nonlinear kernels as summarised in Table 1.

Table 1. NMF algorithms.

Class	β-divergence method (3)	Frobenius norm method (4)	Reference(s)
BNMF	Multiplicative Update (MU) and variants		[29, 30]
	—	Accelerated MU (A-MU)	[31]
	—	Alternative non-negative least squares algorithms (ANLS) and variants (e.g. Projected gradient descent (PGD))	[22, 32, 33]
	—	Alternative Least Square (ALS) and variants	[31]
	Coordinate Descent (CD), Block CD, Cyclic CD, Greedy CD (GCD)		[34, 35]
	Alternating direction method of multipliers (ADMM)		[36, 37]
	Prime-dual (PD)		[38]
CNMF	Sparse NMF, Discriminant NMF, Orthogonal NMF, NMF on Manifold (e.g. graph regularised NMF).		[39, 40] [41]
SNMF	Weighted NMF, Convolutive NMF, NM Tri-factorisation, multi-layer factorisation		[42] [43]
GNMF	Semi-BNF (W or H need not be nonnegative), Nonnegative Tensor Factorisation (NTF), Kernel NMF		[20] [44]

The binary matrix factorisation (BMF), relevant to the Boolean matrix that we are considering in this research, is a CNMF (7) with the following constraints [40]:

$$J_1(W) = \|H_{ij}^2 - H_{ij}\|_F^2, \quad J_2(H) = \|W_{ij}^2 - W_{ij}\|_F^2.$$

5 Software Implementations

The availability of the software implementations of the algorithms is becoming more and more important in modern day research because it is difficult to implement complex algorithms correctly. LibNMF [45] is a C implementation of NMF which is released in 2011.

However, the use of LibNMF is relatively restricted. Since then, the relatively less complex NMF algorithms mentioned in Section 4 are implemented in various numerical computing platforms such as MATLAB, Julia and Python and this made our research possible.

There are many MATLAB implementations (e.g. NMF toolbox at https://www.audiolabs-erlangen.de/resources/MIR/NMFtoolbox/) but NMFLibrary (MIT license) [46] has implemented nearly all algorithms mentioned in Table 1.

The software implementations in Python include Scikit-learn [47], libnmf (https://pypi.org/project/libNMF/) and Nimfa [48] as well as other less known libraries.

There is currently only one implementation in Julia, i.e. NMF.jl (https://github.com/JuliaStats/NMF.jl).

A comparison table of the open source software functions of the basic NMF algorithms (Table 1) are summarised in Table 2.

Table 2. Implementation of Basic NMF algorithms in open source libraries.

Algorithm(s)	NMFLibrary	Scikit-learn	NMF.jl
MU	`fro_mu_nmf`, `div_mu_nmf`	`sklearn.decomposition.NMF(solver='mu')`	`alg::MultUpdate{T}`
ANLS	`anls_nmf`, `pgd_nmf`	`nimfa.Lsnmf`	`alg::ALSPGrad`
ADMM	`admm_nmf`, `div_admm_nmf`	—	—
ALS	`als_nmf`	`libnmf.alsnmf.ALSNMF`	`alg::ProjectedALS`
CD	—	`sklearn.decomposition.NMF(solver='cd')`	`alg::CoordinateDescent{T}`
GCD	—	—	`alg::GreedyCD{T}`
PD	`kl_fpa_nmf`	`libnmf.fpdnmf.FPDNMF`	—

In this paper, we pick Scikit-learn's NMF function because MATLAB's licensing fee is too expensive, Julia is still not widely adopted by the industry because the core of the language is constantly changing while Python is popular in modern AI development and Scikit-learn [47] is under active development.

Scikit-learn's NMF implements the classical multiplicative update (MU) algorithm and the more efficient projective gradient descent based coordinate descent (CD) algorithm. The cost function used by Scikit-learn's NMF belongs to the CNMF mentioned in Table 1 with $D(X\|WH)$ being the Frobenius norm (4) and the regularisation terms are

$$\begin{aligned} J_1(W) &= rn\alpha_W \|vec(W)\|_1 + \frac{(1-r)p}{2}\alpha_W \|W\|_F^2 \\ J_2(H) &= rp\alpha_H \|vec(H)\|_1 + \frac{(1-r)n}{2}\alpha_H \|H\|_F^2. \end{aligned} \tag{8}$$

Here n is the number of samples, p is the number of features, $vec(\cdot)$ flattens a matrix A into a vector $vec(A)$, $\|\cdot\|_1$ is the L1 vector norm, $\|\cdot\|_F$ is the matrix Frobenius norm, the regularisation mixing parameter $r \in [0, 1]$ and the regularisation parameters α_W and α_H are nonnegative.

6 Initialisation

NMF algorithms are based on the solution of the optimisation problem (2) and they are dependent on the initial guess. The simplest initial guess is to choose W to be some permutation

matrix and to choose H to be some rows from the data X. When we do not have any knowledge about the data X, then W and H can be initialised randomly (note that neural network also initialised the internal parameters randomly).

Since the cost function (2) for NMF is nonlinear and NMF is not unique (Section 2), the initial guess for the matrices W and H can converge to different matrices. However, the empirical experiments in the next section show that the NMF CD algorithm and MU algorithm are stable, i.e. they converge to similar matrices (with numerical errors).

The Scikit-learn library's NMF [47] accepts user-defined initial guess for W and H with the option `init='custom'`. For example, we can set this option if the first k rows from the data X are chosen as the initial guess for H and the identity matrix $I_{n\times k}$ is chosen for W. This is called *self-dictionary initialisation* [49].

When $k > \min\{n, p\}$, the option `init='random'` is used by Python's NMF algorithm. It uses the non-negative random matrices which are scaled with the square root of the mean of the data X over the rank k, i.e. $\gamma = \sqrt{\sum_i \sum_j X_{ij}/(npk)}$:

$$W_0 = \gamma R_{n,k}, \quad H_0 = \gamma R_{k,p}$$

where $R_{n,k}$ and $R_{k,p}$ are random normal matrices of the shape $n \times k$ and $k \times p$ respectively.

When $k \leq \min\{n, p\}$, the option `option='nndsvd'` is used by Python's NMF algorithm. It stands for the Nonnegative Double Singular Value Decomposition (NNDSVD) method, which is based on two SVD processes, one approximating the data matrix, the other approximating positive sections of the resulting partial SVD factors utilising an algebraic property of unit rank matrices. The basic NNDSVD algorithm is better fit for sparse factorisation. Its variants NNDSVDa (in which all zeros are set equal to the mean of all elements of the data, `option='nndsvda'`), and NNDSVDar (in which the zeros are set to random perturbations less than the mean of the data divided by 100, which is generally faster, though less accurate than NNDSVDa, `option='nndsvdar'`) are recommended in the dense case[50].

In the empirical study of Boolean matrices associated with black-and-white images, it is found that the user given initial matrices allows NMF to calculate faster followed by the random initial matrices. The NNDSVD and its variants are the slowest because the SVD algorithm can be expensive in computation.

7 Empirical Analysis of Boolean Matrices

In our empirical analysis, we conduct three NMF algorithms: Frobenius-norm based NMF (4) with a custom initialisation and a seeded random initialisation, BMF (binary matrix factorisation, a CNMF with Boolean constraint) with a custom initialisation and an NNDSVD initialisation [40], and KL divergence based NMF (4) with a custom initialisation and a seeded random initialisation against (i) three 2×2 black-and-white images, (ii) MNIST black-and-white images and (iii) selected non-facial black-and-white emoji images.

Black-and-white images are interesting because they can be highly compressed and they are interesting components of digital arts. Black and white images of the size $m \times \ell$ can be represented by Boolean matrices. Since each entry of the Boolean matrix can only be 0 or 1, a $m \times \ell$ Boolean matrix can only have $2^{m\ell}$ possible configurations. The Boolean matrix decomposition is different from numeric matrices because the addition of two ones leads to a one, i.e. $1 + 1 = 1$ (in Boolean algebra, 1 is true and 0 is false, $1 + 1$ is similar to true or true which evaluates to true.

To construct the data matrix X for a collection of n black-and-white images, each black-and-white image will be flattened to a row vector with $m\ell$ elements. Each of the flatten rows will be stacked into X.

First, we consider three 2×2 images represented by Boolean matrices:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}.$$

They can be flattened respectively as row vectors:

$$\begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 & 1 & 0 \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 & 1 & 1 \end{bmatrix}$$

and then stacked to form the following Boolean matrix X:

$$X := \begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{bmatrix}.$$

A possible Boolean matrix decomposition is given below:

$$X = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}}_{W} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}}_{H}.$$

The Boolean matrix H is feature matrix while the Boolean matrix W is the weight matrix.

The factorisation indicates the each image can be obtained as a linear combination of the rows in the feature matrix H.

The Frobenius-norm based NMF (4) with a seeded random initialisation gives

$$X \approx \underbrace{\begin{bmatrix} 1.1046 & 0.0084 \\ 0.1037 & 0.6814 \\ 0.8544 & 0.4878 \end{bmatrix}}_{W_1} \underbrace{\begin{bmatrix} 0.7641 & 0. & 0.0815 & 0.999 \\ 1.1364 & 0. & 1.6071 & 0. \end{bmatrix}}_{H_1} = \begin{bmatrix} 0.8536 & 0. & 0.1036 & 1.1036 \\ 0.8536 & 0. & 1.1036 & 0.1036 \\ 1.2071 & 0. & 0.8536 & 0.8536 \end{bmatrix}.$$

The KL divergence based NMF (5) with a seeded random initialisation gives

$$X \approx \underbrace{\begin{bmatrix} 1.4354 & 0. \\ 0. & 0.5811 \\ 1.0766 & 0.4358 \end{bmatrix}}_{W_2} \underbrace{\begin{bmatrix} 0.5971 & 0. & 0. & 0.7962 \\ 1.475 & 0. & 1.9667 & 0. \end{bmatrix}}_{H_2} = \begin{bmatrix} 0.8571 & 0. & 0. & 1.1429 \\ 0.8571 & 0. & 1.1429 & 0. \\ 1.2857 & 0. & 0.8571 & 0.8571 \end{bmatrix}$$

The BMF with an NNDSVD initialisation gives

$$X \approx \underbrace{\begin{bmatrix} 0. & 1.1432 \\ 1.1769 & 0.6332 \\ 1.1732 & 1.1431 \end{bmatrix}}_{W_3} \underbrace{\begin{bmatrix} 0. & 0. & 0.8463 & 0. \\ 0.9579 & 0. & 0.0064 & 0.7715 \end{bmatrix}}_{H_2} = \begin{bmatrix} 1.095 & 0. & 0.0073 & 0.8819 \\ 0.6065 & 0. & 1. & 0.4885 \\ 1.095 & 0. & 1.0001 & 0.8819 \end{bmatrix}$$

The features H_1, H_2 and H_3 obtained from Frobenius-norm based NMF, KL divergence based NMF and BMF lose information compare to H because the difference between Boolean addition $1 + 1 = 1$ and real number addition $1 + 1 = 2$. However, the feature H_1 from Frobenius-norm and the feature H_2 from KL divergence based NMF are closer to H compare to the feature H_3 from BMF.

Second, we investigate the MNIST data [51] which has been analysed extensively in various supervised learning and dimensional reduction research [52].

The MNIST data (from https://github.com/sbussmann/kaggle-mnist) has 42000 rows and each row contains a 28×28 grey image of a digit with grey colours ranging from 0 to 255. To convert the MNIST data to a Boolean matrix, a threshold of 100 is used. The values below 100 will be mapped to 0 and the rest will be mapped to 1. Figure 1 shows that after applying the threshold of 100, the Boolean matrices in the second row are obtained and they are reasonable approximation to the original grey images. Note that if we choose a larger threshold, the digit image will be thinner while a smaller threshold will lead to a thicker digit image.

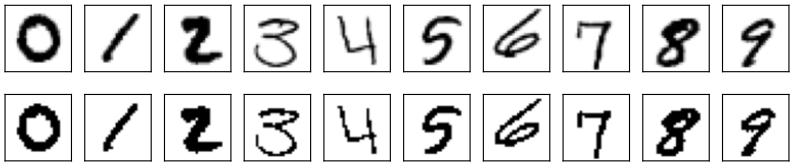

Figure 1. The first row is the 10 digits before applying the threshold 100 while the second row is the 10 digits after applying the threshold 100 converting them to Boolean matrices.

Since the Frobenius-norm NMF is found to be closely related to kernel K-mean clustering [53], and we are expecting the features of the MNIST data to be the 10 digits (corresponding to 10 clusters of digits 0 to 9), the rank k for the NMF algorithms are chosen to be 10.

For a fixed initialisation, we choose W to be a 10×10 identity matrix and H to be the images H_0 of the digits 0 to 9 from Figure 1. Applying the three NMF algorithms, the features obtained are shown in three rows in Figure 2.

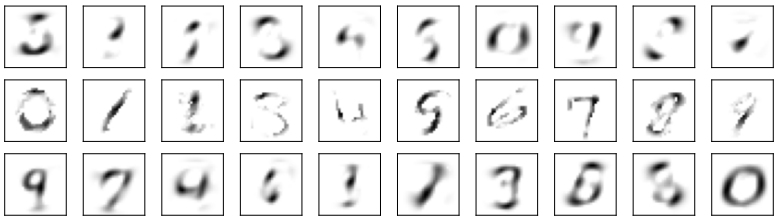

Figure 2. The image features H obtained from Frobenius-norm (4) based NMF and initial matrices $W = I$ and $H = H_0$ (first row), KL (5) based NMF (second row) and initial matrix $W = I$ and $H = H_0$ and BMF (third row) with NNDSVD initialisation.

In the first row, only the digits 5, 2 are sort of recognisable while parts of other digits are identified. In the second row, the KL divergence based NMF captures all the 0 to 9 digits rather well (despite the poor quality of the digits 2, 3, 4 and 5). In the third row, only the digits 9, 7, 1, 3, 6, 8 and 0 identified by BMF are recognisable.

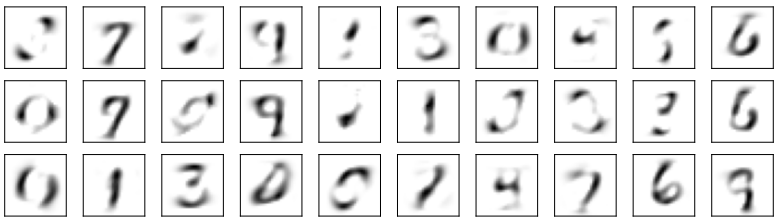

Figure 3. The ten image features H obtained for Frobenius-norm (4) based NMF, KL (5) based NMF and BMF with W and H initialised to random matrices.

When both W and H are initialised to random matrices, the NMF algorithms will converge to different results as illustrated in Figure 3.

In the first row, only the digits 7, 9, 3, 0 and 6 identified by the Frobenius norm based NMF algorithm are sort of recognisable, while other digits are only parts of some of the digits. In the second row, the KL divergence based NMF is only capture 0, 7, 9, 1 and 6 while other features are only portions of digits. In the third row, only the digits 0, 1, 3, 7, 6 and 9 identified by BMF are recognisable.

All the features identified in Figure 2 and Figure 3 can be used to reconstruct the digits 0 to 9. However, we feel that the KL divergence based NMF algorithm with appropriate initial guess is the best for capturing the 10 digits.

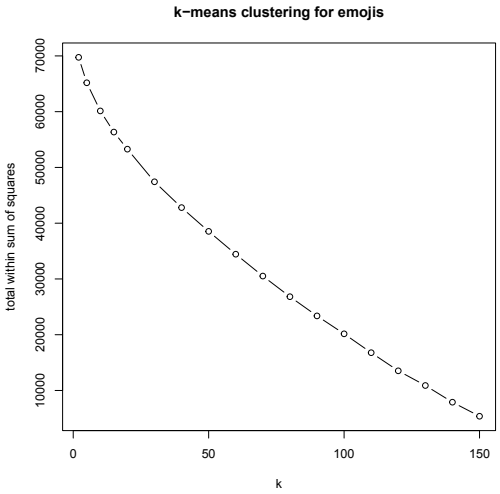

Figure 4. Total within sum of squared errors of the k-means clustering of the emojis data.

Third, we analyse 175 non-facial black-and-white emojis of the size 50×50. To determine the rank, we analyse the total within sum of squared errors of the k-means clustering of the data and obtain the Figure 4. There is no elbow in the curve indicating no clear clusters. We pick the rank of 50 which is close to 50% reduction in the error in Figure 4.

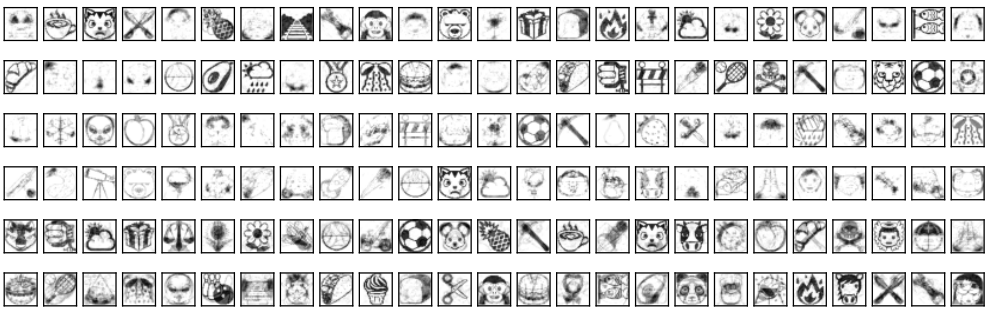

Figure 5. 50 features extracted from the Frobenius norm based NMF algorithm, KL divergence based NMF algorithm and BMF with random initialisation are shown in first and second rows, third and fourth rows and fifth and sixth rows respectively.

Applying the three NMF algorithms, we extract the some black-and-white images such as the cat (first row third column, fourth row twelfth column fifth row seventh column) which appear clearly while KL divergence based NMF algorithm extracts various parts or outlines

from the original black-and-white images. The Frobenius norm based NMF algorithm also extracts "eyes" like first row first column and second row fourth column of the original images. In contrast, BMF seems to mix a few images like the sixth row first to fifth columns.

In contrast to the two earlier scenarios where the rank is determined, the rank in the last scenario is not determined. Setting different ranks would give us different sets of image features. They are like the mixtures of the supplied data and some of them can be regarded as inspired image features.

8 Conclusion

This research sets out to understand the mathematical formulation of NMF, to investigate the software implementations of various NMF algorithms and to investigate the image features extracted by NMF algorithms.

The mathematical formulations of the three NMF algorithms used in this research are stated in the mathematical formulation and classifications of NMFs sections.

In the literature review, we discovered that NMF is a well established with over 30 years of development. However, the software implementations are lagging behind. The only NMF algorithms implemented in Python are just the Frobenius norm based NMF algorithm, KL divergence based NMF algorithm and BMF which are used in our empirical study. MATLAB has a better implementations of NMF algorithms but are costly.

In the investigation of the image features extracted by NMF algorithms, we have considered three scenarios.

The first scenario is a 4×3 Boolean matrix which we know the rank through a Boolean matrix factorisation WH theoretically. Working through the three NMF algorithms, we find that BMF didn't quite capture the theoretical H while the usually NMF captures the theoretical H despite the spread out of values around the value 0 and 1.

The second scenario is a 42000×28^2 Boolean matrix derived from the MNIST data (a collection of 28×28 grey images of digits 0 to 9). The rank is set to be 10, which corresponds to the 10 digits. If we set the rank a value smaller than 10, we will get a mix of the digits while if we set the rank to a value larger than 10, we would break down the digits to portions of the digits. This is not what we want because we want to know if NMF algorithms can capture the complete ten digits. With an appropriate initial guess for matrices W and H, KL divergence based NMF algorithm captures the complete digits 0 to 9 while Frobenius norm based NMF algorithm and BMF algorithm only captures some of the complete digits while only some portions of the ten digits. However, when random matrix initialisation is used, all three NMF algorithms are not able to capture all the ten digits.

In the third scenario, a 175×50^2 Boolean matrix derived from non-facial emojis is considered. Since the emojis are all different black-and-white images with seemingly no common features, we expect the Boolean matrix to be full rank. Since NMF is related to k-means clustering [53], we apply the k-means clustering to detect the k clusters (which corresponds to the rank in NMF) and find that $k = 50$ reduces the errors in k-means clustering by approximately 50%. So we pick the rank to be 50. The NMF algorithms all give 50 features which are interesting, some images retained their complete shape like the cat image while others are portions or mixtures of emojis.

The second scenario inspires us to conclude that when we have a lot of common data (e.g. pictures of various cartoon characters), we can use KL divergence based NMF algorithm with appropriate initial guess to identify the common characteristics of the data (the cartoon characters in particular).

The third scenario inspires us that various NMF algorithms can be used as tools for creating new images from the old images that we have drawn. These NMF algorithms can be used

by art designers to inspire them with new designs. However, a further research is required to provide a sorting of the features found by various NMF algorithms so that we can better compare the images identified by the various NMF algorithms.

References

[1] D.D. Lee, H.S. Seung, Nature **401**, 788 (1999), https://doi.org/10.1038/44565

[2] P.C. Barman, N. Iqbal, S.Y. Lee, *Non-negative Matrix Factorization Based Text Mining: Feature Extraction and Classification*, in *Neural Information Processing. ICONIP 2006. Lecture Notes in Computer Science*, edited by I. King, J. Wang, L. Chan, D. Wang (Springer, Berlin, Heidelberg, 2006), Vol. 4233, https://doi.org/10.1007/11893257_78

[3] W. Xu, X. Liu, Y. Gong, *Document Clustering Based on Non-Negative Matrix Factorization*, in *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval* (Association for Computing Machinery, New York, NY, USA, 2003), SIGIR '03, p. 267–273, ISBN 1581136463, https://doi.org/10.1145/860435.860485

[4] W. Liu, N. Zheng, Q. You, CHINESE SCI BULL **51**, 7 (2006)

[5] S.W. Fu, P.C. Li, Y.H. Lai, C.C. Yang, L.C. Hsieh, Y. Tsao, IEEE Transactions on Biomedical Engineering (2016)

[6] S. Lee, H.S. Pang, IEEE Access **8**, 122384 (2020)

[7] W. Wang, S. Wang, D. Qin, Y. Fang, Y. Zheng, Biomedical Signal Processing and Control **79**, 104180 (2023)

[8] Y. Xie, K. Xie, Q. Yang, S. Xie, Biomedical Signal Processing and Control **69**, 102899 (2021)

[9] G. Yu, K. Wang, G. Fu, M. Guo, J. Wang, IEEE/ACM transactions on computational biology and bioinformatics **17**, 238 (2020)

[10] S. Phon-Amnuaisuk, Procedia Computer Science **24**, 261 (2013), 17th Asia Pacific Symposium on Intelligent and Evolutionary Systems, IES2013

[11] L. Hogben, ed., *Handbook of Linear Algebra* (CRC Press, 2014)

[12] N.K. Kumar, J. Shneider, ArXiv **abs/1606.06511** (2016)

[13] D. Joyner, *Adventures in Group Theory. Rubik's cube, Merlin's machine, and other mathematical toys* (Johns Hopkins University Press, Baltimore, MD, 2008)

[14] M. Jeter, W. Pye, Linear Algebra and its Applications **38**, 171 (1981)

[15] C. Févotte, J. Idier, Neural Computation **23**, 2421 (2011)

[16] L. Li, G. Lebanon, H. Park, *Fast Bregman Divergence NMF Using Taylor Expansion and Coordinate Descent*, in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Association for Computing Machinery, New York, NY, USA, 2012), KDD '12, p. 307–315, ISBN 9781450314626, https://doi.org/10.1145/2339530.2339582

[17] A. Cichocki, H. Lee, Y.D. Kim, S. Choi, Pattern Recognition Letters **29**, 1433 (2008)

[18] K. Machida, T. Takenouchi, *Non-negative Matrix Factorization based on γ-divergence*, in *2015 International Joint Conference on Neural Networks (IJCNN)* (2015), pp. 1–6

[19] S. Eguchi, Y. Kano, Tech. rep., Tokyo Institute of Statistical Mathematics, Tokyo, Japan (2001)

[20] A. Cichocki, R. Zdunek, A.H. Phan, S.I. Amari, *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-Way Data Analysis and Blind Source Separation* (John Wiley & Sons, Ltd, 2009)

[21] Y.X. Wang, Y.J. Zhang, IEEE Transactions on Knowledge and Data Engineering **25**, 1336 (2013)

[22] J. Kim, Y. He, H. Park, Journal of Global Optimization **58**, 285 (2014)

[23] N. Gillis, *Nonnegative Matrix Factorization* (Society for Industrial and Applied Mathematics, Philadelphia, PA, 2020), https://epubs.siam.org/doi/pdf/10.1137/1.9781611976410, https://epubs.siam.org/doi/abs/10.1137/1.9781611976410

[24] S. Lee, Applied Sciences **10** (2020)

[25] M.N.M. Ndaw, P. Ngom, *Relationship between the bregman divergence and beta-divergence and their applications* (2018), 1805.07086

[26] C. Févotte, J. Idier, Neural Computation **23** (2011)

[27] A. Cichocki, R. Zdunek, S. Choi, R. Plemmons, S.i. Amari, *Non-Negative Tensor Factorization using Alpha and Beta Divergences*, in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07* (2007), Vol. 3, pp. III–1393–III–1396

[28] M.C.K. Tweedie, *An index which distinguishes between some important exponential families*, in *Statistics: Applications and New Directions. Proceedings of the Indian Statistical Institute Golden Jubilee International Conference* (Calcutta: Indian Statistical Institute, 1984), pp. 579–604

[29] D.D. Lee, H.S. Seung, *Algorithms for Non-negative Matrix Factorization*, in *Advances in NIPS* (2001)

[30] C.J. Lin, IEEE Trans. Neural Network **18**, pp.1589 (2007)

[31] N. Gillis, F. Glineur, Neural Computation **24**, 1085 (2012)

[32] C.J. Lin, Neural Computation **19**, 2756 (2007)

[33] J. Kim, H. Park, SIAM J. Sci. Comput. **33**, 3261 (2011)

[34] A. Cichocki, A.H. PHAN, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **92**, 708 (2009)

[35] C.J. Hsieh, I.S. Dhillon, *Fast Coordinate Descent Methods with variable selection for Non-Negative Matrix Factorization*, in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2011), pp. 1064–1072

[36] D.L. Sun, C. Févotte, *Alternating direction method of multipliers for non-negative matrix factorization with the beta-divergence*, in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2014), pp. 6201–6205

[37] D. Hajinezhad, T.H. Chang, X. Wang, Q. Shi, M. Hong, *Nonnegative matrix factorization using ADMM: Algorithm and convergence analysis*, in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2016), pp. 4742–4746

[38] F. Yanez, F. Bach, *Primal-dual algorithms for non-negative matrix factorization with the Kullback-Leibler divergence*, in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE, 2017), pp. 2257–2261

[39] R. Peharz, F. Pernkopf, Neurocomputing **80**, 38 (2012), special Issue on Machine Learning for Signal Processing 2010

[40] Z. Zhang, T. Li, C. Ding, X. Zhang, *Binary Matrix Factorization with Applications*, in *Seventh IEEE International Conference on Data Mining (ICDM 2007)* (2007), pp. 391–400

[41] S. Wang, T.H. Chang, Y. Cui, J.S. Pang, *Clustering by orthogonal nmf model and non-convex penalty optimization* (2021), 1906.00570

[42] Y.D. Kim, S. Choi, *Weighted nonnegative matrix factorization*, in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing* (2009), pp. 1541–1544

[43] A. Degleris, B. Antin, S. Ganguli, A.H. Williams, *Fast convolutive nonnegative matrix factorization through coordinate and block coordinate updates* (2019), 1907.00139

[44] D. Zhang, Z.H. Zhou, S. Chen, *Non-negative Matrix Factorization on Kernels*, in *PRICAI 2006: Trends in Artificial Intelligence*, edited by Q. Yang, G. Webb (Springer Berlin Heidelberg, Berlin, Heidelberg, 2006), pp. 404–412, ISBN 978-3-540-36668-3

[45] A.G.K. Janecek, S.S. Grotthoff, W.N. Gansterer, Computing and Informatics **30**, 205 (2011)

[46] H. Kasai, *NMFLibrary: Matlab library for non-negative matrix factorization (nmf)*, https://github.com/hiroyuki-kasai/NMFLibrary (2017)

[47] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg et al., Journal of Machine Learning Research **12**, 2825 (2011)

[48] M. Zitnik, B. Zupan, Journal of Machine Learning Research **13**, 849 (2012)

[49] J.E. Cohen, N. Gillis, *A new approach to dictionary-based nonnegative matrix factorization*, in *2017 25th European Signal Processing Conference (EUSIPCO)* (2017), pp. 493–497

[50] C. Boutsidis, E. Gallopoulos, Pattern Recognition **41**, 1350 (2008)

[51] L. Deng, IEEE Signal Processing Magazine **29**, 141 (2012)

[52] A. Baldominos, Y. Saez, P. Isasi, Applied Sciences **9** (2019)

[53] C. Ding, X. He, H.D. Simon, *On the Equivalence of Nonnegative Matrix Factorization and Spectral Clustering*, in *Proceedings of the 2005 SIAM International Conference on Data Mining (SDM)* (2005), pp. 606–610