K-WEB: NONNEGATIVE DICTIONARY LEARNING FOR SPARSE IMAGE REPRESENTATIONS

Marco Bevilacqua[⋆†] *Aline Roumy*[⋆] *Christine Guillemot*[⋆] *Marie-Line Alberi Morel*[†]

[⋆] INRIA Rennes, Campus de Beaulieu, 35042 Rennes Cedex, France
[†] Alcatel-Lucent, Bell Labs France, Route de Villejust, 91620 Nozay, France

ABSTRACT

This paper presents a new nonnegative dictionary learning method, to decompose an input data matrix into a dictionary of nonnegative atoms, and a representation matrix with a strict ℓ^0-sparsity constraint. This constraint makes each input vector representable by a limited combination of atoms. The proposed method consists of two steps which are alternatively iterated: a sparse coding and a dictionary update stage. As for the dictionary update, an original method is proposed, which we call *K-WEB*, as it involves the computation of k WEighted Barycenters. The so designed algorithm is shown to outperform other methods in the literature that address the same learning problem, in different applications, and both with synthetic and "real" data, i.e. coming from natural images.

Index Terms— Dictionary learning, sparse representations, K-SVD, NMF

1. INTRODUCTION

Dictionary learning methods aim at finding a suitable representation of the data, namely a set of atoms that form a dictionary and possibly make particular structures present in the data explicit. When the learning process is performed by trying to adapt the dictionary to a set of signal examples Y, dictionary learning can be seen mathematically as a matrix factorization problem, where the matrix Y, containing column by column the input data vectors, is factorized into two other matrices such that $Y \approx DX$: D is the learnt dictionary; X is the representation matrix, in which each column represents the "projection" of the related input vectors w.r.t. the dictionary found. Nonnegative Matrix Factorization (NMF) [1] refers to a few recent techniques designed to perform such factorization, with a nonnegative constraint on all the factors. Therefore, NMF is in all respects a dictionary learning tool. Moreover, the nonnegative property makes it particularly attractive for image processing purposes, where, unless some transformations are applied on the image pixels, we have to deal with nonnegative values.

Sparse models for images turn out to be useful for compression (e.g. [2]) and a number of applications that require the solution of ill-posed problems (e.g. denoising [3]). Therefore, we want to introduce sparsity into the dictionary learning framework, s.t. the atoms that we learn are capable to sparsely represent the input data. That would mean to have a sparse constraint on the representation matrix X. NMF is shown to generally tend to sparse structures: due to the nonnegative constraint that makes the representation purely additive, the dictionary matrix D, at the end of the learning process, often reveals "parts" of the input objects. However, there is no evidence of sparsity on the matrix X.

In [4] the author provides a NMF method with sparseness constraints on the columns of D or X, where the sparseness of a vector is defined as a function of its ℓ^1 and ℓ^2 norms. In this paper, we want to propose instead a new method to solve the same problem as NMF, i.e. finding a nonnegative representation of nonnegative data, but with a strict sparse constraint on the ℓ^0-pseudo-norm of the matrix X: each column of X is constrained to have at most L nonzero components. The algorithm consists of two steps that alternatively recompute X and D. In particular, the method for updating the dictionary computes singularly each dictionary atom as the weighted barycenter of an opportune set of vectors: we call this method *K-WEB*.

In Section 2 we present the problem of NMF with ℓ^0-sparsity constraints and revise some methods in the literature that solve it; Section 3, instead, describes in detail the proposed method. Before concluding, Section 4 shows two applications in the image domain, and compares our proposed algorithm with equivalent methods in the literature.

2. NONNEGATIVE MATRIX FACTORIZATION WITH ℓ^0-SPARSITY CONSTRAINTS

Nonnegative matrix factorization (NMF) [1] aims at factorizing an input matrix Y in two matrices D and X with only nonnegative entries, such that $Y \approx DX$ according to some error measure. The same authors propose in [5] the following multiplicative rules to alternatively update D and X

$$X_{ij} \leftarrow X_{ij} \frac{\left(D^T Y\right)_{ij}}{\left(D^T D X\right)_{ij}} , \tag{1}$$

$$D_{ij} \leftarrow D_{ij} \frac{\left(YX^T\right)_{ij}}{\left(DXX^T\right)_{ij}} \ , \qquad (2)$$

where $(A)_{ij}$ indicates the element at the i-th row and the j-th column of the matrix A. (2) and (1) are therefore element-wise multiplicative rules: each element of the matrices D and X is updated by multiplying itself by an appropriate factor. The above mentioned rules are shown in [5] to lead to a local minimum of the approximation error $\|Y - DX\|_F^2$.

NMF can be interpreted as a dictionary learning method. In fact, given a $d \times n$ data matrix Y, we learn a dictionary matrix D of dimension $d \times k$, with $k \le n$ usually, and a representation matrix X. In other words, we can see each input data vector $\mathbf{y_j}$, a column of the matrix Y, as a combination of the atoms of the dictionary, the columns of the matrix D, weighted by the coefficients of the respective column $\mathbf{x_j}$ in the matrix X: $\mathbf{y_j} = \sum_{i=1}^{k} \mathbf{d_i} x_j(i)$.

Now, we want to include in this formulation the concept of *sparse representation*, by imposing that each input vector $\mathbf{y_j}$ is represented by a combination of at most L atoms of the dictionary. This implies that a sparsity constraint is put on the representation matrix X, according to the more natural measure of sparseness which is the ℓ^0-pseudo-norm: each column $\mathbf{x_j}$, which is the encoding of the related input vector $\mathbf{y_j}$, is constrained to have at most L nonzero elements.

Our NMF problem becomes then the following sparse nonnegative matrix factorization problem:

$$\min_{D \ge 0, X \ge 0} \|Y - DX\|_F^2 \quad \text{subject to} \quad \forall i \ \|\mathbf{x_i}\|_0 \le L \ . \qquad (3)$$

2.1. Related works

In the literature not many solutions have been presented to solve the problem (3), i.e. to find, for nonnegative input data, a nonnegative dictionary D and an exactly sparse encoding X w.r.t. the dictionary found. We take as references nonnegative K-SVD (NN K-SVD) [6], the nonnegative adaptation of K-SVD [7], the well-known method for dictionary learning, and the sparse NMF algorithm of Peharz et al. [8].

NN K-SVD, like K-SVD, consists of a 2-step procedure, which is summarized in Algorithm 2.1. Once the dictionary D is initialized (usually with random values or by taking k random vectors from the input data set), two steps are iterated until the desired number of iterations is reached: a *sparse nonnegative coding stage*, where the matrix X is computed, and a *dictionary update stage*, where the dictionary D is refined. As for the first step, a sparse coder referred to as *NNBP* is used. In NNBP, first the support of X is computed: the NMF formula for $X(1)$ is iterated several times, to identify, for each input vector, the salient atoms that we will keep for the sparse representation. For a given column of X, in fact, the k highest coefficients are considered to be referred to the important atoms; the remaining $(n - k)$, instead, are set to zero. The selected coefficients are then recomputed by solving a nonnegative least squares problem. The dictionary update is column-wise: for each dictionary atom $\mathbf{d_j}$, an error matrix is constructed, as the residual on Y when we exclude $\mathbf{d_j}$ from the dictionary D. Then, the best rank-1 approximation for it is found through SVD. The factorization performed is used to update both $\mathbf{d_j}$ and the related row vector of X $\mathbf{x_j^r}$ (which reports the "contribution" of $\mathbf{d_j}$ for all the input vectors); therefore, in the second step of NN K-SVD D and X are contextually updated, as specified in Algorithm 2.1.

```
Algorithm 2.1: NONNEGATIVE K-SVD [6]($Y$)

$D \leftarrow$ initialization
repeat
  $X \leftarrow$ sparsely code Y with D using NNBP
  $D, X \leftarrow$ column and row-wise updates with SVD
until $numIterations = M$
```

The sparse NMF algorithm of Peharz et al. [8] presents a "double-nested" scheme, as reported in Algorithm 2.2. After the dictionary initialization, the representation matrix X is computed with a sparse coder called by the authors *NMP* (Nonnegative Matching Pursuit), a nonnegative version of OMP [9], where, when choosing a new atom, we simply discard it if a negative projection results from that. Then, within an inner loop, D and X are iteratively refined by using the NMF multiplicate rules (2) and (1). Since the multiplications are element-wise, the sparsity of X is perfectly preserved (zeros remain). All this procedure is repeated several times: a new sparse encoding X is found using NMP, and the factorization is refined again in the NMF inner loop.

```
Algorithm 2.2: SPARSE NMF [8]($Y$)

$D \leftarrow$ initialization
repeat
  $X \leftarrow$ sparsely code Y with D using NMP
  repeat
    $D \leftarrow$ update with (2)
    $X \leftarrow$ update with (1)
  until $numIterations = M_2$
until $numIterations = M_1$
```

3. PROPOSED ALGORITHM

As in NN K-SVD and sparse NMF, we want to solve the problem (3), i.e. learn a nonnegative dictionary suitable for exactly sparse representations, by finding a factorization of a signal example matrix Y. For this sake, we want to design two methods to update, possibly separately, the two factors:

- a *nonnegative sparse coding stage* to (re-)compute X;
- a *dictionary update stage* to update D.

3.1. Nonnegative sparse coding

In the nonnegative sparse coding stage, we aim at finding, for each input vector $\mathbf{y_j}$, the best nonnegative ℓ^0-sparse coding $\mathbf{x_j}$ w.r.t to a given dictionary:

$$\mathbf{x_j} = \underset{\mathbf{x} \geq 0}{\arg\min} \, \|\mathbf{y_j} - D\mathbf{x}\|_2^2 \quad \text{s.t.} \quad \|\mathbf{x}\|_0 \leq L \; . \tag{4}$$

To solve (4), we decide to use the NMP algorithm described in [8]. We believe, indeed, that a greedy "OMP-like" choice of the atoms leads to a better sparse approximation than the NNBP method used in NN K-SVD. In the latter, the atoms, that have the highest coefficients according to a first NMF approximation of X, are chosen, but these atoms are not necessarily those ones which minimize the approximation error. We also believe that the sparse coding stage, and so the computation of the support of X, should be performed as much frequently as possible. Therefore, we choose to adopt the simple 2-step scheme of NN K-SVD (Algorithm 2.1), where the identification of the support of X is done at each step of the unique loop, rather than the double-nested one of [8].

3.2. Dictionary update with *K-WEB*

As for the second step of our algorithm (update of the dictionary), we propose an original way to update D column by column, by considering X fixed and not updating it as well, as done in K-SVD. X and D are therefore updated separately in the two steps of the algorithm.

By considering the error matrix $E = Y - DX$, we can separate the contribution to the matrix approximation due to a particular dictionary atom $\mathbf{d_j}$ in this way:

$$E = \left(Y - \sum_{i \neq j} \mathbf{d_i x_i^r} \right) - \mathbf{d_j x_j^r} = E_{\text{-}j} - \mathbf{d_j x_j^r} \; , \tag{5}$$

where $E_{\text{-}j}$ is the error matrix without considering the contribution of the atom $\mathbf{d_j}$.

As our goal is to minimize the norm of E, we would like $E_{\text{-}j} \approx \mathbf{d_j x_j^r}$. In NN K-SVD the matrix $E_{\text{-}j}$, once pruned from those columns for which the related coefficients in $\mathbf{x_j^r}$ are zero, is factorized according to SVD. However, SVD, as it performs a full factorization, forces us to update both $\mathbf{d_j}$ and $\mathbf{x_j^r}$, which are placed respectively in D and X. We want, instead, to find $\mathbf{d_j}$, by considering $\mathbf{x_j^r}$ fixed. The problem to solve is the following:

$$\mathbf{d_j}^{\star} = \underset{\mathbf{d_j} \geq 0}{\arg\min} \, \|E_{\text{-}j} - \mathbf{d_j x_j^r}\|_2^2 \; . \tag{6}$$

The problem (6) is in a quadratic form in any of the coordinates of $\mathbf{d_j}$, and has the following closed-form solution:

$$\mathbf{d_j}^{\star} = \max \left(\frac{\sum_{i=1}^{n} E_{\text{-}j,i} \, x_j^r(i)}{\mathbf{x_j^r x_j^{rT}}}, \mathbf{0} \right) \; , \tag{7}$$

where $E_{\text{-}j,i}$ is the i-th column of the matrix $E_{\text{-}j}$, and the $\max$ operator is intended element-wise.

The minimization problem described in (6) has a geometric interpretation. Let us first consider that $\mathbf{x_j^r}$ is a vector of all ones. $\mathbf{d_j}^{\star}$ in (6) is then the vector that minimizes the sum of all the Euclidean distances, between itself and the column vectors of $E_{\text{-}j}$, i.e. the barycenter of the related set of points. With a given vector of coefficients $\mathbf{x_j^r}$, the solution is the *weighted barycenter* expressed by (7). Therefore, we call this new method for updating the dictionary, which requires the computation of k WEighted Barycenters, *K-WEB*.

The procedure we design is finally summarized in Algorithm 3.1. It is a 2-step procedure: NMP is used as a sparse coder to compute X; the new K-WEB method is subsequently used to update the dictionary matrix D.

```
Algorithm 3.1: K-WEB ALGORITHM($Y$)

$D \leftarrow$ initialization
repeat
  $X \leftarrow$ sparsely code Y with D using NMP
  $D \leftarrow$ column-wise update with K-WEB
until $numIterations = M$
```

4. EXPERIMENTS

To validate our new nonnegative dictionary learning method with ℓ^0 constraints, we apply it to two different scenarios in image processing.

4.1. Image patch approximation

The first application is an image patch approximation problem: we want to train a dictionary suitable for sparsely approximating natural image patches. To do that we take $n = 50000$ 8×8 patches (therefore the dimension of the input data vectors $d = 64$), randomly taken from natural images. The so formed matrix Y is the input of the dictionary learning algorithm, which factorizes it into a dictionary D and a representation matrix X. The number of atoms chosen is $k = 250$; the target sparsity level is $L = 15$, i.e. each column of X contains at most 15 nonzero entries. After the dictionary learning procedure, D is used with other test images: the appropriate sparse coder (NNBP for NN K-SVD, NMP for our proposed algorithm and the sparse NMF method of Peharz et al.) is used to find a sparse approximation of each input patch of the test image, by taking the atoms of D as bases. Table 1 reports the performance (*p-index*) of the three methods with different test images, in terms of average MSE (*avgMSE*) of the patch reconstructions converted in a PSNR-like logarithmic scale:

$$\begin{aligned} \textit{avgMSE} &= \frac{1}{N_p \cdot d} \sum_{i=1}^{N_p} \|\mathbf{y_i} - D\mathbf{x_i}\|_2^2 \\ \textit{p-index} &= 10 \log_{10} \left(\textit{avgMSE}^{-1} \right) \end{aligned} \; , \tag{8}$$

where N_p is the total number of input patches, $\mathbf{y_i}$ is a particular patch, and $\mathbf{x_i}$ is its sparse representation w.r.t. D.

	Images			
Method	**Bird**	**Butterfly**	**Eyetest**	**Head**
NN K-SVD	25.54	24.69	21.18	27.10
Sparse NMF	40.83	32.13	23.32	37.54
K-WEB	**44.25**	**36.44**	**26.20**	**40.99**

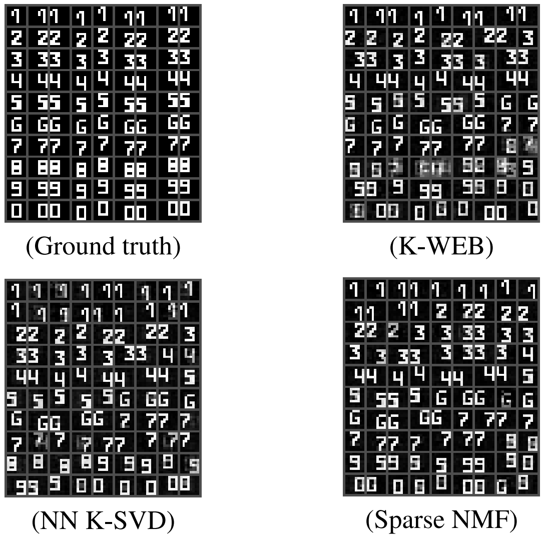
Table 1: Results of the patch approximation problem for different test images.

As we can seen from the table, our method sensitively outperforms the two methods in the literature.

4.2. Dictionary recovery

The second application is a dictionary recovery problem: we have an original dictionary D, with which we generate a data matrix Y, by sparsely combining atoms of it randomly taken (a random sparse matrix X is generated); some Gaussian noise is also added. Thus, $Y \approx DX$. Y, used as an input of the dictionary learning process, is factorized in a dictionary $\hat{D}$ and a coding matrix $\hat{X}$, which we compare with the original ones. The goal of this test is to see if the method is suitable for recognizing truly sparse data and possibly retrieve the original dictionary that was actually used to generate them.

The original dictionary is depicted in the first subplot of Fig. 1, and consists of $k = 90$ 8×8 patches, then vectorized, representing the 10 digits and shifted versions of them. The number of patches generated for Y is $n = 2000$; $L = 5$.


(Ground truth)

(K-WEB)

(NN K-SVD)

(Sparse NMF)

Fig. 1: Original dictionary with the 90 figures and the dictionaries recovered with the different methods.

Table 2 reports the results of the recovered dictionaries for the tree methods analyzed: NN-KSVD, the sparse NMF method of Peharz et al, and our proposed algorithm. The first parameter presented is the training error *t-err* $= \frac{1}{n \cdot d}\|Y - \hat{D}\hat{X}\|_2^2$, which measure the goodness of the training process itself. The other two parameters better indicate the performance of the test made. The ground truth (GT) error measure the distance between $\hat{D}$ and D: for each atom of the original dictionary $\mathbf{d_j}$, we find the closest atom in $\hat{\mathbf{d}}_\mathbf{j}$ and measure the reciprocal distance ($\text{dist}_j = 1 - |\mathbf{d_j} \cdot \hat{\mathbf{d}}_\mathbf{j}|$); the *GT error* is the sum of all these minimum distances. The percentage of well recovered atoms in another indicator for the success of the test: an atom of D, $\mathbf{d_j}$, is considered well recovered if we find in $\hat{D}$ a vector distant up to a certain threshold ($\text{dist}_j < 0.01$).

Method	**Train. error**	**GT error**	**% recovered**
NN K-SVD	0.0139	1.557	61.11 %
Sparse NMF	0.0129	1.391	63.33 %
K-WEB	**0.0135**	**1.407**	**71.11** %

Table 2: Results of the dictionary recovery problem.

Table 2 shows that the method of Peharz et al. gives a slightly better GT error, but our proposed method is able to better recover a higher number of original digits. Namely, 64 digits out of 90 ($71.\bar{1}\%$) in the case of our algorithm, whereas the sparse NMF method recovers 57 digits out of 90 ($63.\bar{3}\%$). This is confirmed in Fig. 1, that shows as images also the recovered dictionaries. With respect to the other methods, our K-WEB algorithm, despite having some problems with certain digits, leads to more accurate recovered patches; the distribution of the digits is also better “balanced”.

5. CONCLUSION

In this paper we described a novel method to learn a nonnegative dictionary from a nonnegative data set, by explicitly imposing a ℓ^0-sparse constraint. Our method borrows from K-SVD the 2-step scheme. Different implementation choices are made though: NMP, a modified version of OMP that only produces nonnegative coefficients, is used as sparse coder; in order to update the dictionary, instead, a new method called K-WEB is designed. In K-WEB each column of the dictionary is computed as a WEighted Barycenter of a set of points (the columns of an error matrix), and consequently updated. The method has been tested in two different image processing applications, to see if it was suitable for both sparsely representing image patches and for recognizing sparse structures in given input data. In both cases our algorithm performed well and gave better results than other methods in the literature.

As for the future work, we plan to apply our dictionary learning method to other applications in the image processing field. Moreover, by removing the nonnegative constraint, K-WEB can also be used as a dictionary update method in a general sparse dictionary learning procedure, comparable with pure K-SVD.

6. REFERENCES

[1] Daniel D. Lee and H. Sebastian Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 10 1999.

[2] J.F. Murray and K. Kreutz-Delgado, "Sparse image coding using learned overcomplete dictionaries," in *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 10 2004, pp. 579–588.

[3] M. Elad and M. Aharon, "Image Denoising Via Sparse and Redundant Representations Over Learned Dictionaries," *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, 12 2006.

[4] Patrik O. Hoyer, "Non-negative Matrix Factorization with Sparseness Constraints," *J. Mach. Learn. Res.*, vol. 5, pp. 1457–1469, 12 2004.

[5] Daniel D. Lee and H. Sebastian Seung, "Algorithms for Non-negative Matrix Factorization," in *Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference*. 4 2001, pp. 556–562, MIT Press.

[6] Michal Aharon, Michael Elad, and Alfred M. Bruckstein, "K-SVD and its non-negative variant for dictionary design," in *Proceedings of the SPIE conference wavelets*, 2005, pp. 327–339.

[7] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation," *Signal Processing, IEEE Transactions on*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.

[8] R. Peharz, M. Stark, and F. Pernkopf, "Sparse nonnegative matrix factorization using ℓ^0-constraints," in *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 8 2010, pp. 83–88.

[9] Y. Pati, R. Rezaiifar, and P. Krishnaprasad, "Orthogonal Matching Pursuit : recursive function approximation with application to wavelet decomposition," in *Asilomar Conf. on Signals, Systems and Computer*, 1993.