Optimization for Computer Vision - Final Project

Tom Labiausse - tom.labiausse@student-cs.fr

January 2, 2024

1 Introduction

This report aims to present, explain and reproduce some experimentation of the K-SVD algorithm introduced in the following article **K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation** written by *Michal Aharon, Michael Elad* and *Alfred Bruckstein*. The K-SVD method enables sparse representation of vectorized data based on a adaptative dictionnary of atoms that can be learned direclty from a vectorized dataset one wish to study. Sparse representations have many applications such as compression, reconstruction or classification. The specificity of the K-SVD method is that it generalizes the K-means algorithm in a way and is compatible with any pursuit methods for computing the sparse representation of a vector adding flexibility to the method.

Everything in this project was implemented from scratch using only the *numpy* and *scipy* libraries for linear algebra manipulations in Python. A significant part of the work was to implement the K-SVD method before conducting reconstruction experiments on black and white face images. Experiments on the MNIST dataset were also conducted but didn't fit in this report (see the notebook).

2 Method

2.1 Context, notations and main idea behind K-SVD

Above all, let's introduce some notations. Given a dataset of N vectors $Y = (Y_i)_{i\in[1,N]}$ each of dimension n, the task is to find or at least approximate $X^* \in \mathbb{R}^{KxN}$ and $D^* \in \mathbb{R}^{nxK}$ such that :

$$(X^*, D^*) = argmin_{(X,D)}||Y - DX||_F^2$$

subject to:

$$\forall i \in [1, N], ||X_i||_0 \leq T_0$$

D represent the dictionary composed of K atoms (one per column) that one wants to learn while X contains the sparse coding of vectors Y. $Y \in \mathbb{R}^{nxN}$ contains one data vector per column and X_i is the i^{th} column of X containing the sparse representation of Y_i. The sparsity constraint T_0 then imposes a maximum number of atoms from D to reconstruct the vector Y_i.

As explained in the paper, the K-SVD method can be summarized as a generalization of the K-means algorithm. First, we initialize the dictionary D as random or using an overcomplete dictionary such as the concatenation of Fourier basis functions, wavelets, etc... Then, we repeat the following two steps until convergence:

- **Sparse Coding:** Find the sparse representation X given a fixed dictionary D using a pursuit method such as *Orthogonal Matching Pursuit* (OMP). This problem can split into N independant sub-problems written as one needs to solve for each $i \in [1, N]$:

$$argmin_{X_i} ||Y_i - DX_i||_2^2$$

subject to :

$$||X_i||_0 \leq T_0$$

- **Dictionary update:** Update the dictionary D to better fit the data. Each atom of the dictionary is updated sequentially by performing a *Singular Value Decomposition* (SVD). In the same time, the method also updates the non-zero coefficients of X.

K-SVD algorithm loops over these two steps until a stopping criterion is reached. This can be a predefined number of iterations or a specific threshold for the residual reconstruction error.

The parallel between K-SVD and K-means relies in the fact that both methods alternate between a coding step which is a closest centroid affectation for K-means and a dictionary update step which is the computation of each centroid based on the nearest points in K-means. K-SVD relies on a linear algebra tool called the SVD (see **appendix 4.2**) to perform the update step explaining the name of the method.

The **Sparse Coding** step in K-SVD is quite straightforward since there is a lot of literature about pursuit methods for sparse representation given a fixed dictionary. Since the specifity of K-SVD relies in the **Dictionary update**, the following section will explain it in more details.

2.2 Dictionary update with mathematical details

The update of D is done sequentially one atom after the other so that understanding one atom update is sufficient to understand the full dictionary update step. Let's note D_k the k^{th} column of D and x_k the k^{th} row of X meaning that x_k contains the activations of atom k for each vector. Using these notations, one can rewrite the objective function to minimize in the following way:

$$||Y - DX||_F^2 = ||Y - \sum_{j \neq k} D_j x_j - D_k x_k||_F^2 = ||E_k - D_k x_k||_F^2$$

E_k is defined as the reconstruction error when the k^{th} atom isn't used and the last term $D_k x_k$ is a rank 1 matrix. It is clear that E_k doesn't depend on D_k nor on x_k. The main idea behind the K-SVD algorithm is to update the k^{th} atom D_k and the corresponding activations x_k so that their tensor product compensate the remaining error E_k as much as possible.

It can be tempting to apply a SVD directly to matrix E_k in order to find the best rank 1 approximation meaning to approximate D_k and x_k to minimize the objective function. However, doing that sequentially for each atom doesn't guarantee that the resulting activations for each vector would respect the sparsity constraint T_0. Therefore the resulting dictionary might not be adapted at all to allow a sparse representation of Y.

To ensure sparsity, it is necessary to lightly modify the optimization problem by looking at the vectors which are currently relying on atom k in their decomposition. Let's define ω_k as the indices of non zero coefficients in x_k and compute the matrix $\Omega_k \in R^{Nx|\omega_k|}$ containing ones at positions $(\omega_k(i), i)$ and zeros elsewhere. This allows to modify the optimization problem by only

looking at the reconstruction error for vectors using the k^{th} atom at this step of the algorithm. Let's introduce the following notations:

- $\forall i \in [1, K],\ x_i^R = x_i \Omega_k \in R^{|\omega_k|}$
- $Y^R = Y\Omega_k \in R^{nx|\omega_k|}$
- $E_k^R = E_k \Omega_k \in R^{nx|\omega_k|}$

The objective function to minimize can now be redefined by using a specific subset of vectors:

$$||Y^R - \sum_{j \neq k} D_j x_j^R - D_k x_k^R||_F^2 = ||E_k^R - D_k x_k^R||_F^2$$

Applying a SVD decomposition on matrix $E_k^R \in R^{nx|\omega_k|}$, one can now update the k^{th} atom D_k as well as the non zero activations x_k^R. Although the modification of activations only apply to vectors already using the k^{th} atom, it is important to remember that we alternate **Dictionary update** steps with **Sparse Coding** steps which globally determine activations of all atoms for each vector. A crucial advantage of using SVD to update atoms also relies in the guarantee that all atoms remain normalized (as columns of orthogonal matrices). Normalized atoms is is indeed a necessary condition to be able to a run a pursuit method like OMP for sparse coding.

3 Analysis and experiments

3.1 Contextual Analysis and relation to Computer Vision

Sparse representation is a highly relevant concept in computer vision. Indeed, it has been shown experimentally that natural images have sparse representations in specific basis such as the Fourier basis or other wavelet basis. These representations can be leveraged to design efficient compression or reconstruction methods. However Fourier basis as an example can be used for a wide variety of images but does not necessarily give optimal sparse representation for a specific set of images. K-SVD therefore introduces a way to directly learn a dictionary from specific dataset of images. By doing that, it is possible to design sparse representation that would yield better results in downstream tasks such as image reconstruction as we will see in the experiments section.

K-SVD relies on a pursuit method to perform its **Sparse Coding** step. However it offers flexibility by allowing any pursuit method. Moreover, sparse codes are not only updated during that stage but also when performing the **Dictionary update** as we saw in **section 2.2**. This specificity considerably accelerates the learning process compared to a method where the dictionary and code updates are independent.

Relying to the properties of the SVD, it is easy to prove that the **Dictionary update** step guarantees the reconstruction error to decrease. By considering that the pursuit method gives a sufficiently good approximation of the solution to the **Sparse Coding**, we obtain a monotonic reduction in the reconstruction error. Therefore, the convergence of K-SVD to a local minimum is guaranteed.

The convergence property of K-SVD being highly dependant on the pursuit method used during the **Sparse Coding**, it is very important to be able to switch between several pursuit methods or design them by taking into considerations the specificity of a given dataset. It turns out that classic pursuit methods such that OMP perform relatively well when T_0 is small enough compared to n leading to a good behavior of K-SVD when this condition is satisfied.

3.2 Experiments

3.2.1 Dataset

In order to reproduce the black and white image reconstruction experiment presented in the paper, one has to gather a dataset of face images and extract patches that would be used to learn an overcomplete dictionary (more atoms than the number of pixels in each patch). As the paper didn't mention the dataset they used, a reasonable choice is to use the well-known Olivetti dataset composed of 400 64×64 face images. As described in the paper, 11000 patches of size 8×8 are extracted from this dataset leading to a matrix Y of size 64×11000 (see **Figure** 2).

3.2.2 Method and results

This reconstruction experiment aims to compare the results obtained when reconstructing missing values in an image using a predefined DCT or Haar (wavelets) overcomplete dictionary (**Figure** 3) to a learned K-SVD dictionary. It also compares the results obtained when learning a K-SVD dictionary from scratch or direclty from an overcomplete DCT or Haar dictionary. OMP was used as the pursuit method for the **Sparse Coding** step.

We chose an image that was not in the training data decribed in section **3.2** and randomly masked a portion of its pixels before cutting it in patches. Each patch taken was then reconstructed using OMP with different dictionaries. Masked pixels being represented by zeros, they don't contribute to the scalar products when performing OMP.

Figure 1: Example of image reconstruction

In **Figure** 1, we observe much better looking results when using the learned K-SVD dictionary for both corruption levels. When looking at RMSE and MAE between the reconstruction and the original image, the K-SVD method achieves lower errors. The performance gap between the dictionaries is clearly assessed in **Figure** 5, 7 and 6 where the predefined overcomplete DCT and Haar dictionaries are compared to the learned K-SVD dictionaries initialized from scratch or using DCT or Haar atoms.

It is interesting to see that the best performing dictionary for image reconstruction especially for high missing pixels ratio is the K-SVD dictionary initialized from scratch. Indeed, by looking at **Figure** 8, one can see that some atoms seem identical to the original DCT or Haar elements, they were not updated by the K-SVD process.

At last, it's is reassuring to see (**Figure** 4) that the reduction of the reconstruction error is monotonic during training ensuring that OMP gives reasonably good sparse codes for this specific kind of data.

4 Appendix

4.1 Figures

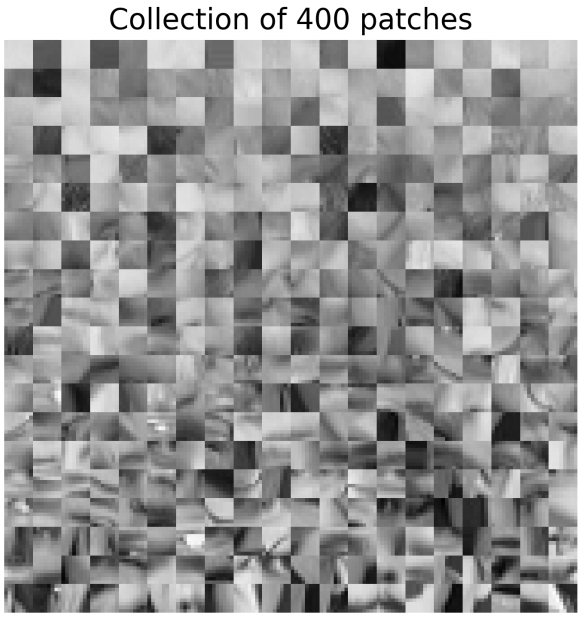
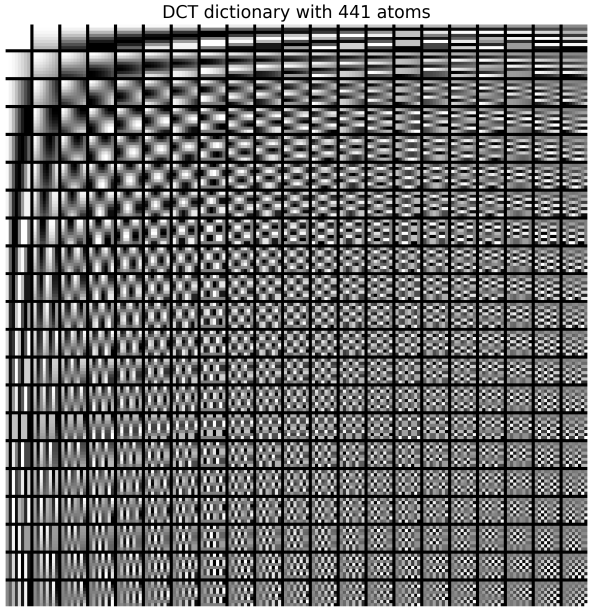
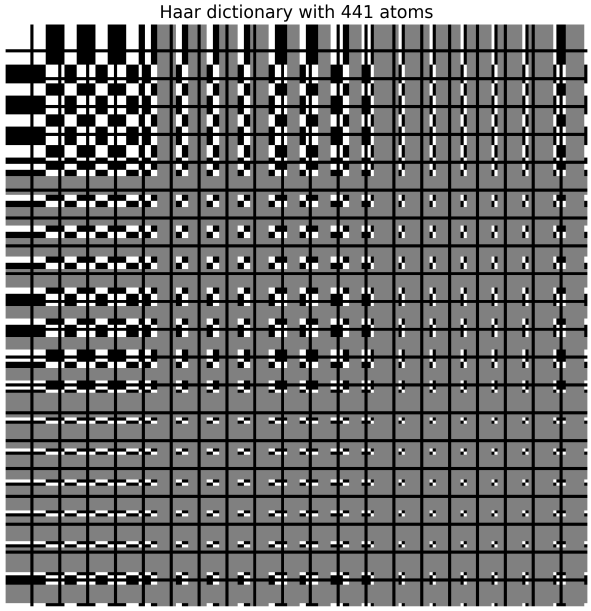


Figure 2: Example of patches extracted from the Olivetti dataset sorted wrt their variance

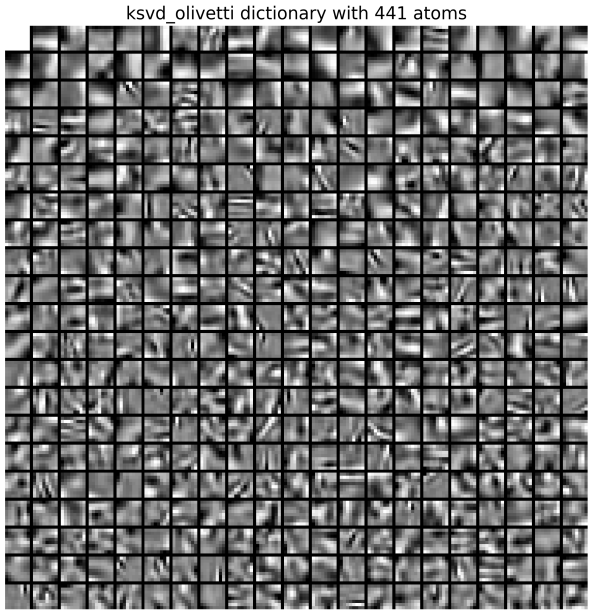


(a) Overcomplete DCT dictionary with 8x8 atoms

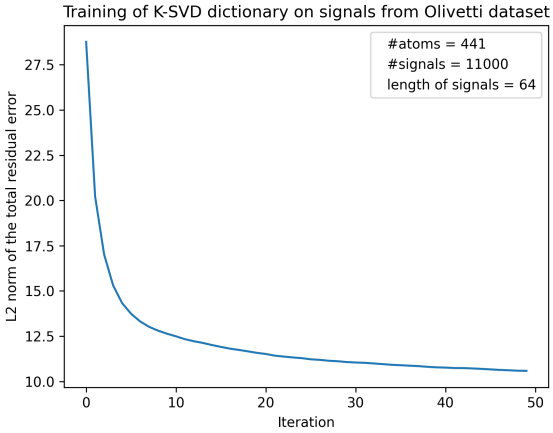


(b) Overcomplete Haar dictionary with 8x8 atoms

Figure 3: Predefined overcomplete dictionaries containing 441 atoms



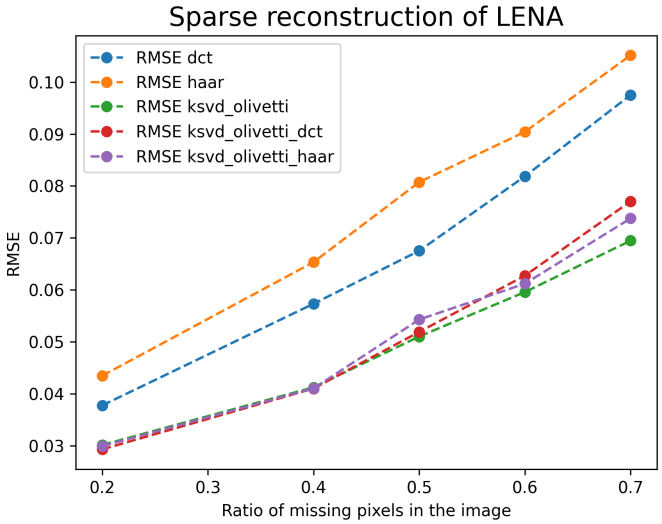
(a) Learned K-SVD dictionary witn 8x8 atoms



(b) Evolution of training error during training

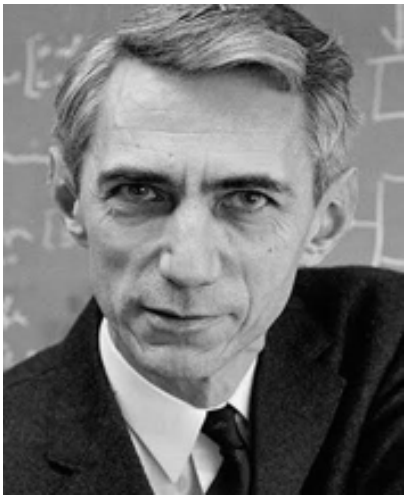
Figure 4: Learning of a 441 atoms dictionary with K-SVD on images from the Olivetti dataset

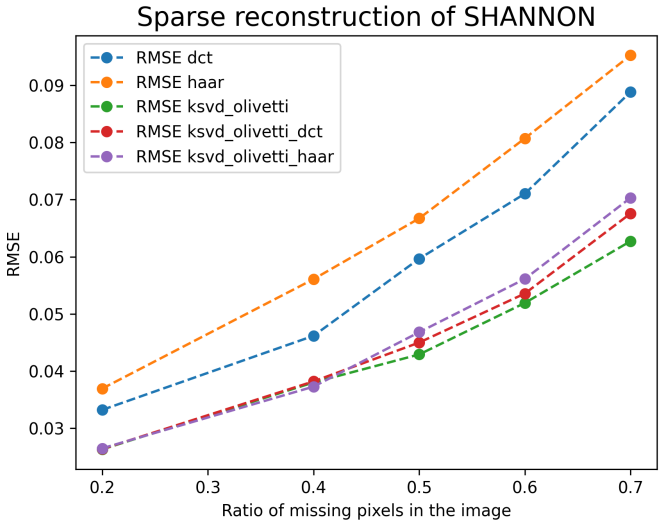
(a) Original image



(b) RMSE for different dictionaries and missing values ratios

Figure 5: Comparison of dictionaries for image reconstructions (2/2)

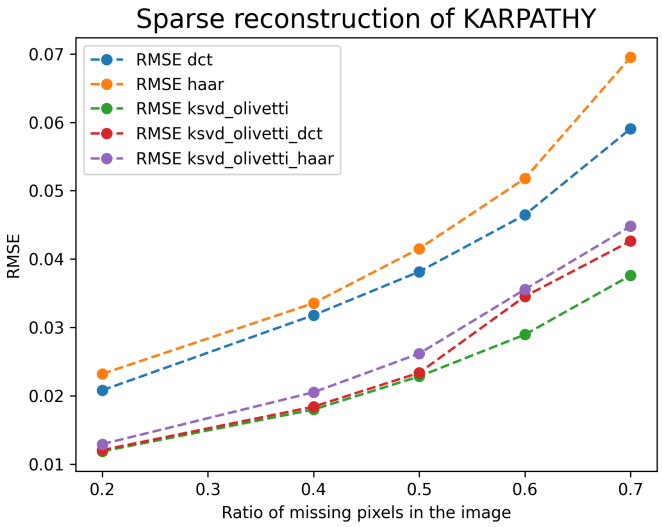




(a) Original image

(b) RMSE for different dictionaries and missing values ratios

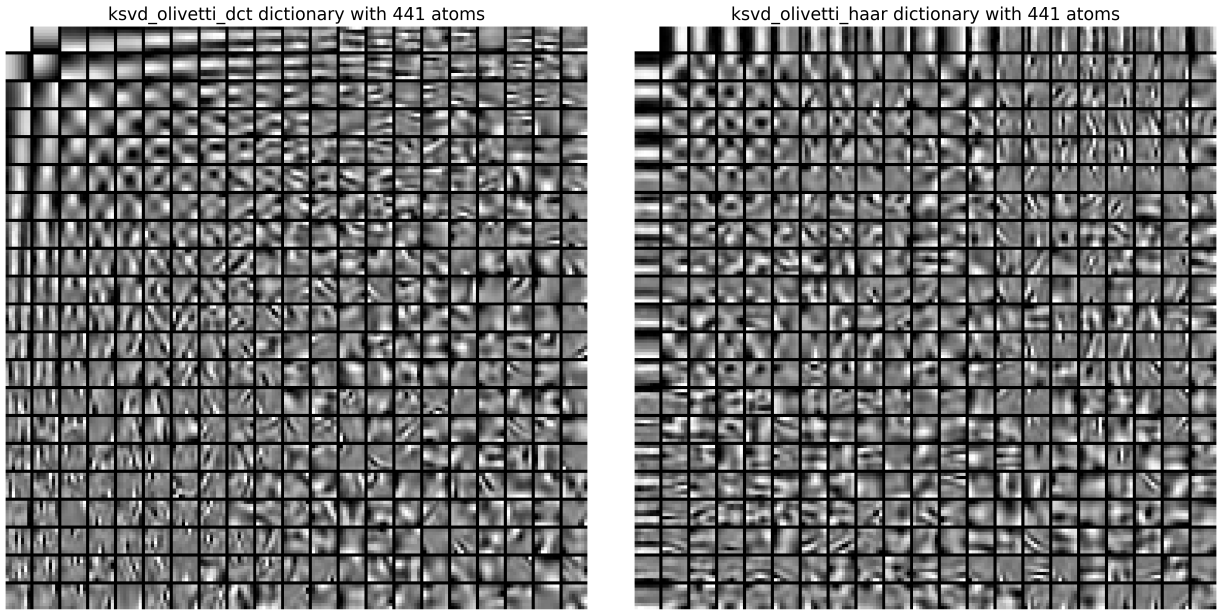
Figure 6: Comparison of dictionaries for image reconstructions (2/2)



(a) Original image

(b) RMSE for different dictionaries and missing values ratios

Figure 7: Comparison of dictionaries for image reconstructions (1/2)



(a) Learned K-SVD dictionary with 8x8 atoms from DCT atoms

(b) Learned K-SVD dictionary with 8x8 atoms from Haar atoms

Figure 8: Learned K-SVD dictionaries with DCT/Haar initialization

4.2 Singular Value Decomposition (SVD)

One usually defines the SVD of any matrix E $\in \mathbb{R}^{nxN}$ as the following:

$$E = U\Delta V^T = \sum_{i=1}^{d} \delta_i U_i V_i^T$$

where U $\in O(n)$, V $\in O(N)$ and $\Delta \in \mathbb{R}^{nxN}$ is a diagonal matrix containing positive values δ_i sorted in descending order on its diagonal called the singular values of E. d is the rank of E and U_i and V_i are the i^{th} column of U and V respectively. The SVD can be used to find the best rank 1 approximation of any matrix E by only keeping the first term in the sum form of the SVD which corresponds to the higher singular value of E and its associated left and right singular vectors.