A Faster K-SVD

Mandela Bright Quashie[1], **Omeiza Olumoye**[1(✉)]
[1]Computational Mathematics, Science and Engineering
Michigan State University, East Lansing, MI 48823, USA
{mbq,olumoyeo}@msu.edu

Abstract

This work is about improving the performance of the K-SVD, which was proposed by Aharon et al. (2006). The K-SVD is considered to be a state-of-the-art algorithm. In this work, we propose a way to make the algorithm better. Specifically, making it faster without compromising the quality of the images We achieved this by replacing the stage where the best rank-1 approximation using singular value decomposition (SVD) is updated in the algorithm with an l_1-norm principal component analysis (l_1-norm PCA).

Keywords: *K-SVD, PCA, Dictionary Learning, Sparse Coding, Norm*

1 Background and Motivation

This work attempts to improve upon the novel K-SVD algorithm by reducing the computation time. The K-SVD algorithm is in two parts: the sparse coding part and the updating of the dictionary. We delved into exploiting both parts of the K-SVD algorithm with the aim of making it efficient.

2 Methods

2.1 Introduction

The work will be based on using the l_1. We will first look at sparse solution and why the l_1 norm is preferred. Thereafter, we briefly recap on dictionary learning, but we will not dwell so much on the sparse coding theory because nothing really changed. Futhermore, we will explore K-Means algorithm and the generalization of it to become the K-SVD. Conclusively, we will build our intuition of our proposed method by replacing the SVD step in the Dictionary method with the l_1-PCA.

2.2 Sparsity and the l_1 norm.

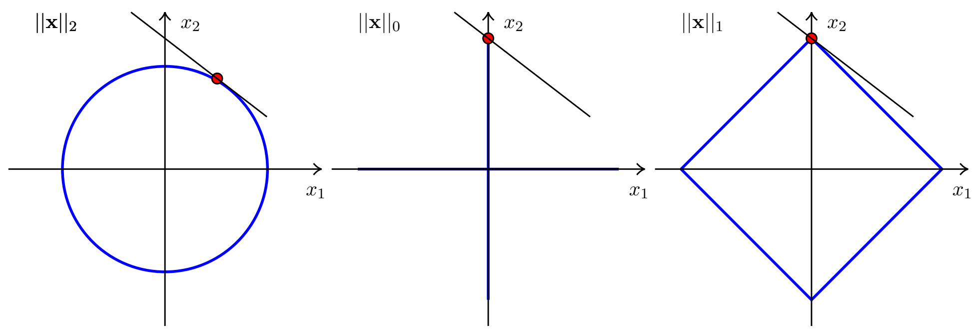
The l_1 norm is widely used in linear algebra for sparsity in general. It is used widely in sparse optimization procedures in statistics, inverse problems, and signal processing. Given the problem

$$y = Ax \tag{1}$$

where we know y and A assuming the system is underdetermined, then we have infinite $\mathbf{x}$'s that satisfy the problem. If we consider that $\mathbf{x} \in \mathbb{R}^{\mathbf{2}}$. The l_2 norm, $\|\mathbf{x}\|_{\mathbf{2}} = \sqrt{x_1^2 + x_2^2}$, thus the solution to the problem is any point that lies on the line shown on Figure 1a. The minimum 2-norm solution is the point the circle meets the line. Every other point will intersect a larger radius and thus have a greater 2-norm. However in our application the minimum 2-norm is not preferred, rather we want the $\mathbf{x}$ that is the sparsest possible and still satisfies the problem. This is generally achieved using the l_0 norm as can be seen in Figure 1b. The l_0 promotes sparsity, however this norm is not really a norm because it fails the scaling condition for a norm. To find the l_0 norm solution of a convex

5th Deep Learning Indaba Conference (DLI 2023).

optimization problem is an NP hard problem. Instead we use the l_1 norm. It is a norm and also tends to give a sparse solution as can be seen in Figure 1c. This generalizes even in higher dimensions.



(a) l_2 (b) l_0 (c) l_1

Figure 1: L_2 regularization, L_0 regularization, and L_1 regularization

2.3 Dictionary Learning

Dictionary learning also referred to as Sparse Dictionary Learning or Sparse Coding. The aim of Dictionary Learning is to build a sparse representation of an input data, where the data is represented by a linear combination of elements, these elements are called atoms. A collection of atoms are dictionary.

Brief of Dictionary learning:

- $\boxed{InputData}$ $Y = [y_1, y_2, \ldots, y_K] \in Y_i \in R^m$
- $\boxed{Learn\ a\ Dictionary}$ $D \in R^{m \times n} : [d_1, d_2, \ldots, d_n]$
- $\boxed{Learn\ a\ Representation}$ $X = [x_1, x_2, \ldots, x_K],\ x \in R^n$
- $\boxed{Minimize\ the\ Reconstrution\ Error}$ $\min_{x_i \text{sparse}} \|DX - Y\|_F^2$

2.4 Sparse Representation of Signal

Let $y \in \mathbb{R}^n$ be a signal observed, $D \in \mathbb{R}^{n \times K}$ be a dictionary and $x \in \mathbb{R}^K$ be the representation coefficient.
In ideal situation we should have:

$$y = DX$$

However, if we have $D = [d_1, d_2, \ldots, d_K]$ such that each $d_k \in \mathbb{R}^n$, are the basis vectors then we can write y as:

$$y = \sum_{k=1}^{K} x_k d_k$$

where the vector x is assumed to be sparse.
The goal is to solve:

$$\min_{X,D} \|x_i\|_0 \quad \text{subject to} \quad DX = Y$$

where $Y = [y_1, \ldots, y_N]$ is a collection of N observations, and $X = [x_1, \ldots, x_N]$ is a collection of N representation coefficient vectors. The main issue here is that the problem is not convex. In reality there is always noise and so we have $Y \approx DX$.

2.5 K-SVD

We have the K-Means as:

$$\min_{D,X} \|Y - DX\|_F^2 \quad \text{subject to} \quad \forall i, \quad \|x_i\| = 1. \tag{2}$$

When we relax the constraint by allowing $\|x_i\| = T$ we get,

$$\min_{D,X} \|Y - DX\|_F^2 \quad \text{subject to} \quad \forall i, \quad \|x_i\| = T. \tag{3}$$

Now if we fix D, then solve for X in (3) We note that:

$$\|Y - DX\|_F^2 = \sum_{i=1}^{N} \|y_i - Dx_i\|_2^2.$$

Hence all we need to solve:

$$\min_{x_i} \|y_i - Dx_i\|_2^2 \qquad \text{subject to} \quad \forall i, \quad \|x_i\|_0 \leq T. \tag{4}$$

This can be achieved by using any of greedy algorithms such as Matching Pursuit (MP), Orthogonal Matching Pursuits (OMP), Basis Pursuit (BP), and Focal Underdetermined System Solver (FOCUSS) Rubinstein et al. (2008). These techniques can be greedy, convex or non-convex relaxations, etc. We fix X, then update for D so we have the problem:

$$\min_{D} \|DX - Y\|_F^2 \tag{5}$$

We can use Method of Optimal Direction (MOD) by:

$$D = YX^\top (XX^\top)^{-1}$$

The MOD algorithm is efficient for low-Dimensional input data. However computing the pseudoinverse has high complexity and most often difficult to compute. There is also no way of preserving sparsity inherent from X.

2.6 The Fast K-SVD Algorithm

The K-SVD algorithm relies on SVD at the dictionary updating stage. Our proposal is to replace this step with l_1-PCA that was developed by Markopoulos et al. (2014). The normal PCA which is based on the l_2 norm does not do any good. We find the principal component for each of the atom. The algorithm is presented by Kwak (2008) and generalized one can be found in Markopoulos et al. (2014) of updated version in 2018.

2.7 The l_2- PCA Algorithm

1. Initialize: Pick any $w(0)$ and set $w(0) \mapsto \dfrac{w(0)}{\|w(0)\|_2}$ and $t = 0$.
2. For all $i \in \{1, 2, \cdots, n\}$ if $w^\top(t)x_i < 0$, $p_i(t) = -1$, otherwise $p_i(t) = 1$.
3. Flipping and maximization:

   We let $t \mapsto t + 1$ and $w(t) \sum_{i=1}^{n} p_i(t-1)x_i$. Set $w(t) \mapsto \dfrac{w(t)}{\|w(t)\|_2}$.
4. Check for convergence:
   (a) If $w(t) = w(t-1)$ go to step 2.
   (b) Else if there exists i so that $w^\top(t)x_i = 0$, let $w(t)$ be nonzero small vector.
   Set $w(t) \mapsto \dfrac{(w(t) + \Delta w)}{\|w(t) + \Delta w\|_2}$, go to step 2.
   (c) Else, set $w^\star = w(t)$.
   (d) Stop if tolerance is met.

   This work because in the algorithm, the projection vector w converges to $w^\star$, which is a local maximum point of

   $$\sum_{i=1}^{n} |w^\top x_i|.$$

   The proof was taken from Kwak (2008).

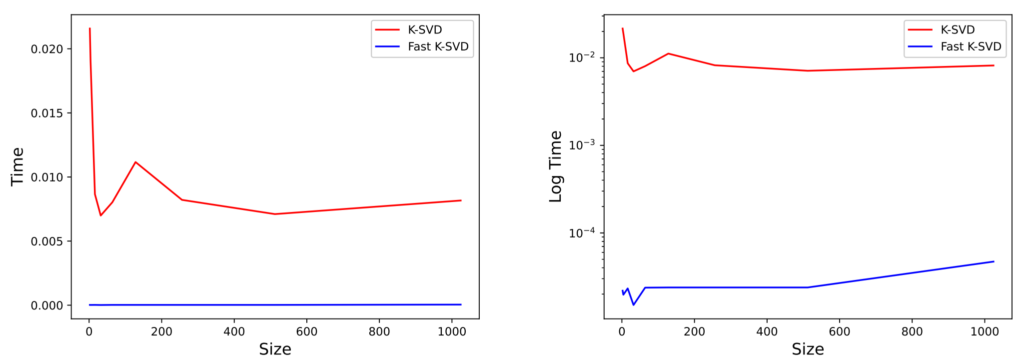
3 Results

In this section we present the results conducted in Python 3.10.7 with the scikit-learn 1.0.2 library on MacBook Pro 2021, Apple M1 Max Chip with memory 32GB.
In the experiment we generated different datasets randomly and using the same parameters each time we simulated results. We can see from the Table 1 below that the times for Faster K-SVD was much smaller than that of the original K-SVD.

Table 1: Computation times for original k-svd and faster k-svd

n	K-SVD	Fast K-SVD
2	7.48712e-02	1.29747391e-05
4	1.82725e-02	1.69372559e-05
16	5.61304e-03	1.13081932e-05
32	4.71804e-03	1.20639801e-05
64	5.65225e-03	1.84845924e-05
128	4.91546e-03	1.39594078e-05
256	5.20929e-03	2.35462189e-05
512	5.47821e-03	1.84965134e-05
1024	5.83408e-03	4.05979156e-05

The results in Table 1 was plotted and be seen in the plot in Figure 2a. The Faster K-SVD variations could not be seen clearly so we took the log of y-axis to ascertain the difference and we can see this from Figure 2b.



(a) Plot of size against time

(b) Plot of size against log time

Figure 2: Plot of size against time for k-svd and fast k-svd

4 Discussion

We found that exploiting the updating of the dictionary portion had an effect on the time of the algorithm runs. The proposed method Faster K-SVD is faster than that of the original K-SVD. Even when we vary the data and add more noise it still beats the K-SVD. In application to signals and images, the Peak Signal-To-Noise Ratio (PSNR) was better from our preliminary studies Ravishankar (2022). This exploitation seems to have a huge potential, due to time constraints we could apply it to diverse problems.

References

Ravishankar, S., "Advanced Signal Processing. Lecture Notes, pages 1200, 2022." *Fall 2022*, vol. 1, pp. 51–68, 2022

Aharon, M., Elad, M., and Bruckstein, A., 2006 "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation" *IEEE Transactions on signal processing*, 54(11):4311–4322.

Rubinstein, R., Zibulevsky, M., and Elad, M., "Efficient Implementation of the K-SVD Algorithm Using Batch Orthogonal Matching Pursuit." *Technical report* 2008.

Kwak, N., "Principal Component Analysis Based on L1-Norm Maximization". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(9):1672–1680,1013–1024, 2008.

Markopoulos, P. P., Karystinos, G. N., and Pados, D. A., "Optimal Algorithms for L1-Subspace Signal Processing", *IEEE Transactions on Signal Processing*, 62 (19): 5046-5058 2014.