Sparse Coding for Dictionary Learning in Context of Image De-noising

Dhaivat Deepak Shah
ds3267@columbia.edu

Gaurav Ahuja
ga2371@columbia.edu

Sarah Panda
sp3206@columbia.edu

Abstract

Dictionary learning involves solving the following optimization problem: in ——x —— ————m D, D 22+ 1 where is the input signal, is the dictionary and is the sparse representation of the signal. x D The problem of image restoration has been addressed with a multitude of approaches. All the approaches to solve the optimisation problem fall under the 3 broad categories of Relaxation (Basis Pursuit), Greedy approach(Matching Pursuit) or Hybrid methods. Our project primarily focuses on the Relaxation methodology. Here, both and are unknown. Mairal, Julien, et al, 2009 present an online learning D algorithm[1] which involves two optimization problem. First, assumes the to be available and D minimizes over . This is known as the sparse coding problem. Second, updates the after D obtaining . Mairal, Julien, et al, 2009 use LARS[2] to solve the sparse coding problem. We propose to compare the performance of the online dictionary learning algorithm by solving the sparse coding problem using methods[1][5] like featuresign [3], FISTA[4], Interior point, Sequential Shrinkage or Iterative Shrinkage methods and Stochastic Gradient Descent in the context of image restoration.

1 Introduction

Image restoration is a widely studied problem in image processing and the common approaches to solve the problem include filtering, minimizing variations in an image etc. The problem we address in this paper is to remove noise from a given image using Dictionary Learning approach, assuming the noise to be Gaussian and additive. The approach bases itself on representing an image as a sparse representation over a good dictionary. So our problem of restoring images boils down to learning a good dictionary and finding a good representation of any given patch of an image using the dictionary. Designing dictionaries to better t the above model can be done by either selecting one from a pre-specied set of linear transforms, or by adapting the dictionary to a set of training signals. Past research has shown that dictionaries adapted to training signals lead to a lot better signal restoration than the pre-specified dictionaries. Hence, we strive to learn a dictionary from the input signals for the purpose of restoration. We have used the K-SVD algorithm as our basis for dictionary learning and explore different sparse coding methods that can be used for the same. The next section gives a Literature review of the different components of Dictionary Learning. The third section talks about our experimental setting and further sections discuss the results, our analysis and conclusion.

2 Intro to Dictionary learning

Intro to Dictionary learning - KSVD- general KSVD explaination - Online Dictionary Learning

3 K-SVD for dictionary learning

The goal of K-SVD is to adapt a dictionary to the training signals such that it ensures the best sparse representation for each of the signals, under constraints of reconstruction error and sparsity. It is a two-step iterative process, which begins with a dictionary initialization. The dictionary could be initialized with any pre-calculated well known dictionary, like DCT, or created from the noisy image by randomly selecting atoms from the patches of the given image. The first step of the process determines the sparse code for the given training signals with fixed dictionary. The second step assumed fixed sparse code and updates the dictionary to better fit the data. The second step further updates the sparse representation of the training signals to accelerate convergence.

K-SVD tries to optimize the following overall MSE under sparsity constraints in each iteration:

$$\min_{d,x} \|Y - DX\|_F{}^2 subjectto, \forall I, \|x\|_0 <= L$$

The algorithm is outlined below:

1. **Initialization:** Set dictionary matrix $D \in R^{nXK}$ with l_2 normalized columns
2. **while** not converged(j = 1,2,...) **do**
3. **Sparse Coding:**
4. Compute x_i for every training signal y_i, by solving for $\min_{x_i} \|y_i - Dx_i\|_F{}^2$ subject to $\|x_k\|_0 \leq L$
5. **Dictionary Update:**
6. Update each atom k in 1,2,..K in D_{j-1} as below:
   (a) Let w_k be the set of input signals that use this atom D(k).
   (b) Find the overall error in representation, in matrix E_k, $E_k = Y - \sum(j \neq k) d_j x_j{}^2$
   (c) Restrict E_k by choosing columns corresponding to w_k as E_k^R
   (d) Apply SVD decomposition on the restricted error matrix : $E_k^R = U\delta V^T$ Set $d_k = U[:1]$ and $x_r^k = V[:1] * \delta(1,1)$
7. j= j+1
8. **end while**
9. **Output:** $X^* \leftarrow X_j, D^* \leftarrow D_j$

In this paper, we focus on the first step of K-SVD and that is the sparse coding step. The following sections discuss the problem of sparse coding in detail along with the algorithms we have used.

4 Sparse Coding

As mentioned before, any signal y can be represented as a linear combination of atoms in a dictionary D. Sparse coding problem aims at finding the sparsest such representation. On a high level, those atoms summarize the most important features in the input.

$$\min \|x\|_0 subjecttoy = Dx$$

The above equation tries to find the sparsest coefficient vector that exactly reconstructs the input signal. However, this form of exact determination is NP-hard. So, to make it easier to solve, the problem needs to be relaxed by allowing for a small amount of reconstruction error, ϵ. The problem can be thus formulated as,

$$\min \|x\|_0 subjectto \|y - Dx\| \leq \epsilon$$

Still the problem is computationally expensive and following are two of the well known approaches to approximate the solution to the above.

4.1 Relaxation (Basis pursuit)

In the relaxation methodology, the l0 norm in (2) is replaced with the l1 norm, resulting in the following equivalent objective,

$$\min \|y - Dx\|^2 + \lambda \|x\|_1$$

L1 regularization penalizes the non-sparse terms based on their actual values, which may seem unfair, but past research has shown that it leads to a very good approximation and often leads to the sparsest representation. The resulting optimization problem is convex and unconstrained, and hence can be solved using several methods like gradient descent.

4.2 Greedy (Matching pursuit)

Greedy methodology, on the other hand, solves the problem in (2) by sequentially selecting the atom that best explains the data. Upon convergence,

5 Summary of sparse coding techniques used:

5.1 FISTA

The objective function that we have as defined above is:

$$F(x) = \tfrac{1}{2}\|y - Dx\|^2 + \lambda \|x\|_1$$

Here the first term, $f(x)$ is a smooth, convex function with Lipschitz continuous gradient $D^T(Dx - y)$ and $L_f = \|D^T D\|$.

FISTA, proposed by Beck [2] has a faster convergence rate as compared to ISTA. The main difference between the two is that the iterative shrinkage operator is not applied to the previous point alone but to another point which uses a specific linear combination of the previous two points. The algorithm in this case becomes:

1. L_f
2. **Step 0.** $j_1 = x_0 \, \epsilon \, \mathbb{R}^n, t_1 = 1$
3. **Step** k**.**$(k \geq 1)$
4. $x_k = soft(j_k - \frac{1}{L_f}\nabla f(j_k), \frac{\lambda}{L_f})$
5. $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$
6. $y_{k+1} = x_k + \frac{t_k - 1}{t_k + 1}(x_k - x_{k-1})$

5.2 MP

Matching pursuit, proposed by Mallat and Zhang et al, in 1993 is a well known greedy algorithm for learning sparse code of a signal over a given dictionary. The basic idea of the algorithm is outlined below.
At every step of refining the sparse code, the algorithm picks an atom of the dictionary that best represents a portion of the input signal.

The following pseudo-code briefly describes the algorithm:

1. **Input:** Training signal y, Dictionary - D
2. **Initialization:** $R_1 = y, x = 0, n = 1$

3. **while** $R_n >$ Threshold **do**
4. Find $d_i \in$ D with max correlation with residual i.e. $< R_n, d_i >$
5. $x_i = x_i + < R, d_i >$
6. $R_{n+1} = R_n - x_i d_i$
7. $n = n + 1$
8. **end while**
9. **Output:** Sparse code of y over D, $x^* \leftarrow x_n$

The algorithm evaluates the best matching atom at every step by evaluating the inner product between the Residual, Rn and the individual atoms from the dictionary. The coefficient vector x is updated with the atom that has maximum inner product and its contribution to the signal is deducted from the signal. The process repeats until the residual falls below a threshold. The sparse coefficients thus learned is believed to be the sparsest approximation of the decomposition of the signal over the dictionary.

5.3 OMP

A variation of the Matching Pursuit is the Orthogonal Matching Pursuit, also a greedy method that iteratively selects the atom having highest correlation with the residual. The difference is that after adding the best matched atom to the coefficient vector, the residual is updated with the orthogonal projection of the input signal y onto the subspace of previously selected atoms. This set of selected atoms is called the active set and since the resulting residual is orthogonal to each of the atoms in the set, none of those atoms are selected in the future iterations.

The OMP algorithm is as below:

1. **Input:** Training signal y, Dictionary - D
2. **Initialization:** $R_0 = y$, $X(c_0) = 0, n = 0$
3. **while** not converged **do**
4. Solve for X_{t_i} by minimizing,
   $\max_t {X_t}^T R_{n-1}$
   Add X_{t_i} to the set of selected atoms; $c_n = c_{n-1} \cup t_i$
5. Let $P_n = X(c_n)(X(c_n)^T X(c_n))^{-1} X(c_n)^T$ denote the projection onto the linear space spanned by the elements of $X(c_n)$.
6. Update $R_n = (I - P_n)y$.
7. $n = n + 1$
8. **end while**
9. **Output:** Sparse code of y over D, $x^* \leftarrow x_n$

5.4 ALM

The Augmented Lagrangian Method was proposed individually by Powell [9] and Hestenes [5] and is explained in depth by Nocedal, Wright [12]. It is an extension to the quadratic penalty function method proposed by Courant [4]. In the penalty function method, we add a quadratic penalty term for each of the constraints in a constrained optimisation problem. That is for:

$$\min_x f(x) \text{ subj. to } c(x) = 0$$

the formulation becomes,

$$\min_x f(x) + \tfrac{\mu}{2} \|c(x)\|^2$$

where μ is the penalty parameter and is always positive. We can now progressively increase μ towards ∞ and try to find a minimizer for the objective function. Thus we can convert the constrained minimisation problem into an unconstrained one. However, an ill-conditioning for the Hessian in the formulation can lead to significantly poor results for the iterative methods. To reduce this possibility, we include a Lagrange multiplier to get the Augmented Lagrangian as:

$$\min_x f(x) + \tfrac{\mu}{2}\|c(x)\|^2 + \lambda c(x)$$

We look at both the primal and the dual formulations of this approach.

5.4.1 Primal ALM

In our case, the problem boils down to:

$$L_\mu(x, e, \lambda) = \|x\|_1 + \|e\|_1 + \tfrac{\mu}{2}\|y - Dx - e\|^2 + \lambda(y - Dx - e)$$

For the primal problem, Bertsekas [3] showed that there exists a λ^* and μ^* such that

$$\begin{aligned} e_{k+1} &= arg\min_e L_\mu(x_k, e, \lambda_k) \\ x_{k+1} &= arg\min_x L_\mu(x, e_{k+1}, \lambda_k) \\ \lambda_{k+1} &= \lambda_k + \mu(y - Dx_{k+1} - e_{k+1}) \end{aligned}$$

Of the above equations, the one for e has a closed form solution. As for the update of x, it is a standard L_1 minimisation problem which we solve using the FISTA method explained above.

So the overall algorithm can be summarized as in [13]:

1. **Input:**$y \,\epsilon\, \mathbb{R}^m$, $D \,\epsilon\, \mathbb{R}^{mxn}$, $x_1 = 0$, $e_1 = y$, $\lambda_1 = 0$
2. **while** not converged(k = 1,2,...) **do**
3. $\quad e_{k+1} \leftarrow shrink(y - Dx_k + \frac{1}{\mu}\lambda_k, \frac{1}{\mu})$
4. $\quad t_1 \leftarrow 1, z_1 \leftarrow x_k, w_1 \leftarrow x_k$
5. $\quad$ **while** not converged (l = 1,2,...) **do**
6. $\quad\quad w_{l+1} \leftarrow shrink(z_l + \frac{1}{l}D^T(y - Dz_1 - e_{k+1} + \frac{1}{\mu}\lambda_k), \frac{1}{\mu L})$
7. $\quad\quad t_{l+1} \leftarrow \frac{1}{2}(1 + \sqrt{1 + 4t_l^2})$
8. $\quad\quad z_{l+1} \leftarrow w_{l+1} + \frac{t_l - 1}{t_l + 1}(w_{l+1} - w_l)$
9. $\quad$ **end while**
10. $\quad x_{k+1} \leftarrow w_l, \lambda_{k+1} \leftarrow \lambda_k + \mu(y - Dx_{k+1} - e_{k+1})$
11. **end while**
12. **Output:** $x^* \leftarrow x_k, e^* \leftarrow e_k$

While in the general case of Augmented Lagrangian methods, the value of μ is incremented after every iteration, we are holding it fixed to the initialisation value.

5.4.2 Dual ALM

The dual Augmented Lagrangian method for efficient sparse reconstruction was proposed by Tomioka [11]. It tries to solve the dual of the problem we have been tackling so far as:

$$\max_j y^T j \text{ subj. to } D^T j \,\epsilon\, \mathbb{B}_1^\infty$$

where $\mathbb{B}_1^\infty = \{x \,\epsilon\, \mathbb{R}^n : \|x\|_\infty \leq 1\}$

The associated Lagrangian function becomes:

$$\min_{j,z} -y^T j - \lambda^T (z - D^T j) + \frac{\mu}{2}\|z - D^T j\|^2 \text{ subj. to } z \,\epsilon\, \mathbb{B}_1^\infty$$

Here, again there is a simultaneous minimization w.r.t j,λ and z. So we adopt an alternation strategy to get the following algorithm as in [13]:

1. **Input:** $y \,\epsilon\, \mathbb{R}^{\geqslant}, B = [A, I] \,\epsilon\, \mathbb{R}^{mx(n+m)}, w_1 = 0, j_1 = 0$
2. **while** not converged (k = 1,2,...) **do**
3. $z_{k+1} = \mathbb{P}_{\mathbb{B}_1^\infty}(B^T j_k + \frac{w_k}{\mu})$
4. $j_{k+1} = (BB^T)^{-1}(Dz_{k+1} - (Bw_k - y)/\mu)$
5. $w_{k+1} = w_k - \mu(z_{k+1} - D^T j_{k+1})$
6. **end while**
7. **Output:** $\lambda^* \leftarrow w_k[1:n], e^* \leftarrow w_k[n+1:n+m], j^* \leftarrow j_k$

5.5 Feature Sign

- Feature Sign

5.6 L1LS

- L1LS

6 Experimental Setup

We have used the K-SVD approach suggested in the previous sections for Dictionary Learning, and have used above mentioned 7 solvers for benchmarking.

8 iterations of K-SVD were performed. For each of the solvers, the following parameters have been varied:

1. Starting Image Noise Levels: 10, 20, 50 dB
2. Dictionary Size: 64, 128, 256

And the comparison is based on:

1. Execution Time
2. SNR of the de-noised image

All other supplied parameters were fixed across approaches, although tuning them specifically to a particular method might have resulted in better results, in order to maintain consistency across the experiment.

K-SVD toolbox [10] and L1 Solvers [13], [6] were used for computation, and all the processes were run using matlab in nodesktop mode.

Pertinent values, images and dictionaries at each of the intermediate steps were recorded and are hosted at AML Project Results under the respective results folders.

7 Findings

8 Analysis

9 Conclusion

References

[1] Michal Aharon, Michael Elad, and Alfred Bruckstein. -svd: An algorithm for designing overcomplete dictionaries for sparse representation. *Signal Processing, IEEE Transactions on*, 54(11):4311–4322, 2006.

[2] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[3] Dimitri P Bertsekas. Nonlinear programming. 1999. *Athena Scientific*.

[4] Richard Courant et al. Variational methods for the solution of problems of equilibrium and vibrations. *Bull. Amer. Math. Soc*, 49(1):1–23, 1943.

[5] Magnus R Hestenes. Multiplier and gradient methods. *Journal of optimization theory and applications*, 4(5):303–320, 1969.

[6] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y Ng. Efficient sparse coding algorithms. *Advances in neural information processing systems*, 19:801, 2007.

[7] Stéphane G Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *Signal Processing, IEEE Transactions on*, 41(12):3397–3415, 1993.

[8] Yagyensh Chandra Pati, Ramin Rezaiifar, and PS Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Signals, Systems and Computers, 1993. 1993 Conference Record of The Twenty-Seventh Asilomar Conference on*, pages 40–44. IEEE, 1993.

[9] Michael JD Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The computer journal*, 7(2):155–162, 1964.

[10] Ron Rubinstein, Michael Zibulevsky, and Michael Elad. Efficient implementation of the k-svd algorithm using batch orthogonal matching pursuit. *CS Technion*, page 40, 2008.

[11] Ryota Tomioka and Masashi Sugiyama. Dual-augmented lagrangian method for efficient sparse reconstruction. *Signal Processing Letters, IEEE*, 16(12):1067–1070, 2009.

[12] SJ Wright and J Nocedal. *Numerical optimization*, volume 2. Springer New York, 1999.

[13] Allen Y Yang, Zihan Zhou, Arvind Ganesh, S Shankar Sastry, and Yi Ma. Fast l1-minimization algorithms for robust face recognition. *arXiv preprint arXiv:1007.3753*, 2010.

10 Appendix

10.1 Appendix-1

10.2 Appendix2