EK-SVD: Optimized Dictionary Design for Sparse Representations

Raazia Mazhar, Paul D. Gader
Department of Computer and Information Science and Engineering, University of Florida
rmazhar@cise.ufl.edu

Abstract

Sparse representations using overcomplete dictionaries are used in a variety of field such as pattern recognition and compression. However, the size of dictionary is usually a tradeoff between approximation speed and accuracy. In this paper we propose a novel technique called the Enhanced K-SVD algorithm (EK-SVD), which finds a dictionary of optimized size for a given dataset, without compromising its approximation accuracy. EK-SVD improves the K-SVD dictionary learning algorithm by introducing an optimized dictionary size discovery feature to K-SVD. Optimizing strict sparsity and MSE constraints, it starts with a large number of dictionary elements and gradually prunes the under-utilized or similar-looking elements to produce a well-trained dictionary that has no redundant elements. Experimental results show the optimized dictionaries learned using EK-SVD give the same accuracy as dictionaries learned using the K-SVD algorithm while substantially reducing the dictionary size by 60%.

1. Introduction

Sparse signal representations using overcomplete dictionaries are used in a variety of fields such as, pattern recognition [3], image [1] and video coding [2]. Overcompleteness of a set means that it has more members than the dimensionality of the its members. Given an overcomplete set of basis signals, called the dictionary, the goal is to express input signals as sparse linear combinations of the dictionary members. The advantage of overcompleteness of a dictionary is its robustness in case of noisy or degraded signals. Also, it introduces greater variety of shapes in the dictionary, thus leading to sparser representations of a variety of input signals.

Overcompleteness of dictionaries for sparse representations is certainly desirable. However, there are no set guidelines about choosing the optimal size of a dictionary. For example, for an n-dimensional input signals, both a size $n + 1$ and $2n$ dictionary may be considered overcomplete. A bigger dictionary may seem to give more variety of shapes, it also adds to approximation speed. Also, bigger may not always be better as the dictionary can contain some similar looking elements or some elements that are seldom used for representation. Excluding such elements can enhance the encoding speed of the dictionary but will not compromise its approximation accuracy.

In this paper, we propose a novel dictionary learning technique that discovers an optimized number of dictionary elements by reducing redundancies in the learned dictionary. We call this method Enhanced K-SVD (EK-SVD) as it improves upon a state-of-the-art dictionary learning algorithm, K-SVD [4]. For a given dataset, K-SVD learns an excellent dictionary while enforcing a strict sparsity constraint. However, the drawback of K-SVD is that the total number of elements K is heuristically chosen by human interpretation. Therefore, the problem of training a good dictionary using K-SVD boils down to the cumbersome process of manually selecting a good value of K.

This drawback of K-SVD is removed by EK-SVD as it automatically discovers the value of K during the dictionary learning process. For this purpose, EK-SVD uses an approach similar to the Competitive Agglomeration (CA) algorithm [6] to update the dictionary coefficients. CA is a clustering algorithm that discovers the optimal partitioning as well as the optimal number of clusters for a given dataset. Using CA-style approach for updating dictionary coefficients helps prune seldom used elements. If there are many similar-looking elements, this approach helps retain only those elements that are frequently used or can be used in place of the others. Once the correct number of clusters has been discovered, EK-SVD uses the Matching Pursuits (MP) [5] algorithm to learn a truly sparse and accurate dictionary.

The rest of the paper is organized as follows: Section

2 introduces some of the algorithms used by EK-SVD. Section 3 describes the K-SVD algorithm. EK-SVD algorithm is defined in section 4. Exprimental results are reported in section 6 and section 7 concludes the paper.

2. Preliminaries

2.1 Matching Pursuits Algorithm

Matching pursuits is a greedy algorithm that finds sparse representation of a given signal $x \in \Re^n$. Given a set of dictionary elements $D = \{g_j\}_{j=1}^M$ in a Hilbert space H, such that $g_j \in \Re^n$ and $\|g_j\| = 1$, MP approximates x by iteratively projecting it onto D:

$$x = \sum_{j=0}^{t-1} w_j g_j + R_t \tag{1}$$

where R_t is called the t^{th} residue and $R_0 \equiv x$. Also, $w_j = \langle R_j, g_j \rangle$ is called the dictionary coefficient. At each iteration, the j^{th} dictionary element and its coefficient is chosen by projecting the j^{th} residue onto D and choosing the one with the maximum projection: $g_j = g_d$, where $d = \arg\max \langle x, g_k \rangle$, $k \in \{1 \dots M\}$. The residue is updated as: $R_{j+1} = R_j - w_j g_j$.

Final residue after $t-1$ iterations is denoted by R_x. The ordered list of dictionary elements g_j is called the projection sequence $G_x = \{g_1, \dots, g_{t-1}\}$ and the corresponding coefficients w_j are denoted by $W_x = \{w_1, \dots, w_{t-1}\}$.

2.2. Competitive Agglomeration Algorithm

The CA algorithm combines the benefits of the hierarchical and partitional clustering algorithms as it discovers the optimal number of clusters during the clustering process. Given $X = \{x_i\}_{i=1}^N$, $x_i \in \Re^n$ and cluster centres $B = (\beta_1, \dots, \beta_c)$, CA minimizes the following objective function:

$$J(B, U, X) = \sum_{j=1}^{C} \sum_{i=1}^{N} u_{ij}^2 d^2(x_i, \beta_j) - \eta \sum_{j=1}^{C} \left[\sum_{i=1}^{N} u_{ij} \right]^2 \tag{2}$$

Subject to $\sum_{j=1}^{C} u_{ij} = 1$, for $i \in \{1, \dots, N\}$.

$d^2(x_i, \beta_i)$ measures the dissimilarity of the data point x_i to the centre β_j, u_{ij} represents the degree of membership of x_i in cluster β_j. The first component of (2) tries to minimize the dissimilarity between a cluster centre and its members and the second component tries to minimize the total number of clusters. When both components are combined and η is chosen properly, the final partition will minimize the sum of intra-cluster distances, while partitioning the dataset into the smallest possible number of clusters. Hence the CA algorithm reduces redundancies in cluster representatives by merging together similar clusters, while preserving dissimilar clusters.

2.3 MP Based Similarity Measure

In [3] the Matching Pursuits based Similarity Measure (MPSM) is proposed and used with the CA algorithm. MPSM uses the residue and coefficient information provided by the MP algorithm to compare two signals:

$$d^2(x_1, x_2) = \alpha \|R_{x_1}^{x_2} - R_{x_2}\|^2 + (1-\alpha) \|W_{x_1}^{x_2} - W_{x_2}\|^2, \tag{3}$$

where $R_{x_1}^{x_2}$ and $W_{x_1}^{x_2}$ respectively represent the residue and coefficients of x_1 when projected onto the projection sequence G_{x_2} of the signal x_2.

3. The K-SVD Algorithm

The K-SVD algorithm generalizes the K-means clustering algorithm by relaxing the K-means constraint of using only one cluster to represent each x_i. It treats the dictionary elements g_j as the cluster centers and the coefficents w_{ij} as membership of a signal x_i into cluster g_j. K-SVD minimizes the following:

$$\min_{D,W} \{\|X - DW\|_F^2\} \tag{4}$$

Subject to $\|w_i\|_0 \leq T_0$, for $i \in \{1, \dots, N\}$.

$X \in \Re^{nxN}$ is a column matrix of all signals x_i: $X = [x_1 | \dots | x_N]$. $D \in \Re^{nxM}$ is a column matrix of all dictionary elements: $D = [g_1 | \dots | g_M]$ and $W \in \Re^{MxN}$ is a matrix of all coefficients of x_i corresponding to dictionary element g_j and w_i is one row of W corresponding to coefficients of x_i. $\|.\|_0$ is called the l_0 norm and counts the number of non-zero elements in a vector. Thus K-SVD enforces a strict sparsity constraint on w_i. $\|A\|_F$ is called the Frobenius norm and is defined as $\|A\|_F = \sqrt{\sum_{ij} A_{ij}^2}$.

Like K-means, K-SVD uses a two phase approach to update the values of W and D. In the first phase, the dictionary coefficients are updated using MP. In the second phase, D and W are assumed to be fixed and only one column g_k of D is updated at a time. Let the k^{th} row in W that gets multiplied with g_k be denoted by w^k. Then $g_k w^k$ can be separated from (4) as follows:

$$\|X - DW\|_F^2 = \left\|E_k - g_k w^k\right\|_F^2 \tag{5}$$

where $E_k = (X - \sum_{j \neq k} g_j w^j)$ is the approximation error of all x_i when the kth atom is removed. The Singular Value Decomposition (SVD) of E_k will produce

the closest rank-1 matrix that minimize the above error. After removing columns from E_k that do not use g_k, SVD of E_k yields $E_k = U\Delta V^T$. The g_k is replaced with the first column of U and w^k with the first column of V. All dictionary elements g_j are updated using the same method. Iterating through the two phases of K-SVD produces dictionary that approximates given x_i sparsely and accurately.

4. The Enhanced K-SVD Algorithm

The dictionary learning capabilities of K-SVD and pruning capabilities of the CA algorithm can be combined to learn optimal dictionaries. The idea is that in the first phase of K-SVD, instead of using MP to update coefficients g_k, use the CA algorithm. The dictionary elements g_j are treated as the cluster centers and the memberships u_{ij} of signal x_i in the cluster g_j are derived from the coefficients w_{ij}. Memberships of dispensable dictionary elements are progressively made smaller and thus diminished over a few iterations. Such elements can then be pruned. Thus the optimal dictionary design algorithm is defined by the following objective function:

$$J(D,U,X) = \sum_{j=1}^{M}\sum_{i=1}^{N} u_{ij}^2 d^2(x_i,g_j) - \eta \sum_{j=1}^{C}\left[\sum_{i=1}^{N} u_{ij}\right]^2 \tag{6}$$

Subject to $\sum_{j=1}^{C} u_{ij} = 1 - \hat{R}_{x_i}$, for $i \in \{1,\ldots,N\}$.

Note that (6) is same as the CA objective function, except the constraint. In (6)$\hat{R}_{x_i}$ is the normalized residue of the signal x_i and is defined as:

$$\hat{R}_{x_i} = \frac{\|R_{x_i}\|^2}{\|x_i\|^2} \tag{7}$$

MPSM is used as the dissimilarity measure between x_i and g_j. Since g_j is a dictionary element, its projection sequence contains only itself, i.e. $W_{g_j} = \{g_j\}$. Therefore MPSM reduces to:

$$d^2(x_1,x_2) = \alpha\|x_i - w_{ij}g_j\|^2 + (1-\alpha)\|w_{ij}-1\|^2. \tag{8}$$

The interpretation of u_{ij} is interesting as they build the bridge between CA and K-SVD algorithms. The relationship between u_{ij} and w_{ij} is defined as:

$$u_{ij} = \frac{|w_j|^2}{\|x_i\|^2} \tag{9}$$

Therefore the constraint in (6) states that the sum of normalized coefficients w_{ij} and the normalized residue $\hat{R}_{x_i}$ should sum to one. However the values of u_{ij} are updated by minimizing (6). For this purpose, we apply Lagrange multipliers to obtain:

$$\begin{aligned} J &= \sum_{j=1}^{M}\sum_{i=1}^{N} u_{ij}^2 d^2(x_i,g_j) - \eta \sum_{j=1}^{C}\left[\sum_{i=1}^{N} u_{ij}\right]^2 \\ &- \sum_{i=1}^{N}\lambda_i\left(\sum_{j=1}^{C} u_{ij} - 1 + \hat{R}_{x_i}\right) \end{aligned}$$

Assuming D and X fixed, we obtain:

$$\frac{\partial J(D,U,X)}{\partial u_{st}} = 2u_{st}d^2(x_s,g_t) - 2\eta\sum_{i=1}^{N} u_{it} - \lambda_s = 0 \tag{10}$$

$$\Rightarrow u_{st} = \frac{2\eta\tau_t + \lambda_s}{2d^2(x_s,g_t)} \tag{11}$$

where $\tau_t = \sum_{i=1}^{N} u_{it}$. For computational ease, τ_t from previous iteration is used as the values are assumed to not change drastically over consecutive iterations. To solve for λ_s, apply the constraint in (6) to (11):

$$\sum_{t=1}^{M}\frac{2\eta\tau_t + \lambda_s}{2d^2(x_s,g_t)} = 1 - \hat{R}_{x_s} \tag{12}$$

By rearranging terms we get:

$$\lambda_s = 2\left(\frac{1-\hat{R}_{x_s}}{\sum_{t=1}^{M}\delta_{st}} - \eta\frac{\sum_{t=1}^{M}\delta_{st}\tau_t}{\sum_{t=1}^{M}\delta_{st}}\right) \tag{13}$$

where $\delta_{st} = \frac{1}{d^2(x_s,g_t)}$. Putting (13) in (11):

$$u_{ij} = (1-\hat{R}_{x_i})\frac{\delta_{ij}}{\sum_{t=1}^{M}\delta_{ij}} + \eta\delta_{ij}\left(\tau_j - \frac{\sum_{t=1}^{M}\delta_{st}\tau_t}{\sum_{t=1}^{M}\delta_{st}}\right) \tag{14}$$

$$\Rightarrow u_{ij} = (1-\hat{R}_{x_i})u^{(Update)} + \eta u^{(Prune)} \tag{15}$$

where the definitions of $u^{(Update)}$ and $u^{(Prune)}$ follow from (14). When $u^{(Update)}$ dominates (15), the value of u_{ij} is updated as usual and when $u^{(Prune)}$ dominates, the value of u_{ij} may be increased or decreased based on the usage of g_j. A method for calculating η is discussed in [6].

Once all u_{ij} have been updated, equation (9) can be used to find the corresponding w_{ij}. Sign of w_{ij} can be resolved by inspecting the sign of projection of x_i onto g_j. The algorithm then updated the dictionary elements g_j using the K-SVD algorithm and these computed w_{ij}. $\hat{R}_{x_i}$ is assumed to be constant during the iteration of CA. Once the dictionary has been updated, the value of $\hat{R}_{x_i}$ is updated using the MP algorithm. CA iterations continue while the average $\hat{R}_{x_i}$ stays below the desired threshold. Average $\hat{R}_{x_i}$ can go up during an iteration

where a few dictionary elements are dropped simultaneously. But it comes down in successive iterations. In the early pruning stages of the algorithm, the coefficients chosen by the CA algorithm may not be optimal in terms of sparse representation. But once the correct number of dictionary elements has been discovered, the coefficients w_{ij} can be chosen according to the strict sparsity constraint of the K-SVD algorithm using the standard MP algorithm.

5. Experimental Results

The goal of our experiments is to show that a smaller dictionary learned using EK-SVD can achieve the performance of a bigger dictionary learned using the K-SVD algorithm. The training data consisted of 11000 block patches of size 8x8 pixels taken from facial images from a publicaly available face images database [7]. Random face images from database that were not used for training were chosen as the test images.

Since in [4], K was set to 441, we also trained K-SVD with $K = 441$ and $T_0 = 6$. EK-SVD was also initialized at $M = 441$ and $T_0 = 6$. For demonstration purposes, EK-SVD was allowed to run till $M = 1$ and at each pruning step, test images were approximated using the intermediate dictionary. Figure 1 shows the average root mean square error (RMSE) of all test images against the dictionary size M. The value at $M = 441$ is the performance of K-SVD. The RMSE stays almost constant till the dictionary size is reduced to about 40% of its size before it starts increasing. During actual training, accuracy goals were set same as K-SVD and EK-SVD learned the dictionary with $M = 179$.

Figure 2 shows two test images approximated using K-SVD and EK-SVD dictionary. For images 1 and 2, both K-SVD and EK-SVD produce the RMSE of 0.03 and 0.02 respectively. The EK-SVD dictionary thus gives the approximation accuracy similar to the K-SVD dictionary. But it has a huge speed advantage as EK-SVD dictionary is 60% smaller than the K-SVD dictionary.

6. Conclusion

In this paper we introduced a novel dictionary learning technique EK-SVD that discovers the correct number of dictionary elements during dictionary learning, for a given dataset. We demonstrated its enhancement over K-SVD by achieving the same approximation accuracy with a much smaller dictionary.

References

[1] A. E. Moghadam, S. Shirani. Matching Pursuit-Based Region-of-Interest Image Coding. *IEEE Trans. Image Processing*, 16(2), February 2007.

[2] R. Neff, A. Zakhor. Very Low Bit-Rate Video Coding Based on Matching Pursuits *IEEE Trans. Circuits and Systems*, 7(1), February 1997.

[3] R. Mazhar, P. D. Gader, J. N. Wilson. A Matching Pursuit Based Similarity Measure for Fuzzy Clustering and Classification of Signals. *IEEE intl. conf. Fuzzy Systems*, June 2008.

[4] M. Aharon, M. Elad, A. Bruckstein. K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation. *IEEE Trans. Signal Processing*, 54(11), November 2006.

[5] S. G. Mallat, Z. Zhang. Matching Pursuits With Time-Frequency Dictionaries. *IEEE Trans. Signal Processing*, 41(12), December 1993.

[6] H. Frigui, R. Krishnapuram. Clustering by Competitive Agglomeration. *Pattern Recognition*, 30(7): 1109–1119, 1997.

[7] Yale Face Database. http://cvc.yale.edu/projects/ yalefaces/yalefaces.html

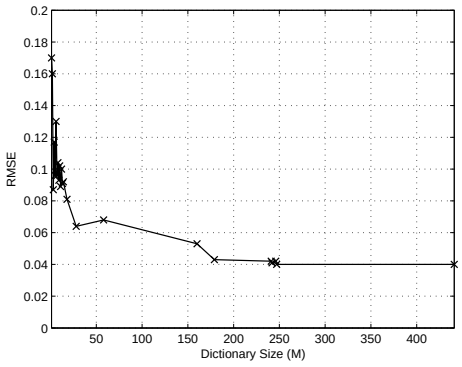

Figure 1. EK-SVD Iterations

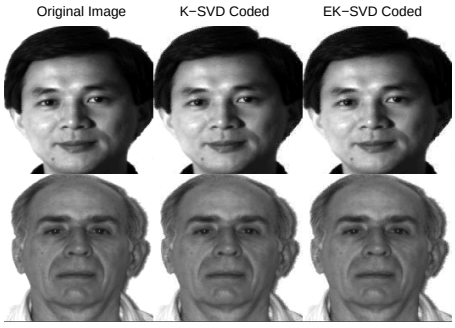

Figure 2. Dictionaries Comparison