Stagewise K-SVD to design efficient dictionaries for sparse representations

Cristian Rusu and Bogdan Dumitrescu, *Member, IEEE*

Abstract

The problem of training a dictionary for sparse representations from a given dataset is receiving a lot of attention mainly due to its applications in the fields of coding, classification and pattern recognition. One of the open questions is how to choose the number of atoms in the dictionary: if the dictionary is too small then the representation errors are big and if the dictionary is too big then using it becomes computationally expensive. In this paper we solve the problem of computing efficient dictionaries of reduced size by a new design method, called Stagewise K-SVD, which is an adaptation of the popular K-SVD algorithm. Since K-SVD performs very well in practice, we use K-SVD steps to gradually build dictionaries that fulfill an imposed error constraint. The conceptual simplicity of the method makes it easy to apply, while the numerical experiments highlight its efficiency for different overcomplete dictionaries.

Index Terms

Sparse representations, dictionary learning, K-SVD

EDICS: DSP-TFSR, IMD-MDSP

The authors are with the Department of Automatic Control and Computers, University Politehnica of Bucharest, Spl. Independenţei 313, Bucharest 060042, Romania (e-mails: cristian.rusu@schur.pub.ro, bogdan.dumitrescu@acse.pub.ro).

B. Dumitrescu is also with Department of Signal Processing, Tampere University of Technology, 33101 Tampere, Finland (e-mail:bogdan.dumitrescu@tut.fi).

I. INTRODUCTION

The field of sparse representations has received continuous attention during the latest years, fueled by recent theoretical developments [1] [2] and applications [3] [4].

In this context, the problem of designing bases for sparse representations is central and has a multitude of application areas, including: image [5] and video [6] coding, pattern recognition and classification [7]. The goal is to find overcomplete bases (dictionaries) in which signals have multiple representations and we seek the sparsest linear combination that gives a good approximation of the signal. The main advantage that this design philosophy brings is the possibility to create extremely good sparse representations for the input data, of course, at the cost of a higher computational time.

Given a dataset $\boldsymbol{Y} \in \mathbb{R}^{n \times m}$ the problem can be formulated in the following manner:

$$\begin{aligned} \underset{\boldsymbol{A},\boldsymbol{X}}{\text{minimize}} \quad & ||\boldsymbol{Y} - \boldsymbol{A}\boldsymbol{X}||_F^2 \\ \text{subject to} \quad & ||\boldsymbol{x}_i||_0 \leq s_0, \ 1 \leq i \leq m \end{aligned} \tag{1}$$

that reads: find the full matrix $\boldsymbol{A} \in \mathbb{R}^{n \times N}$ with $N \ll m$, called here dictionary and its columns called atoms, and the sparse representation matrix $\boldsymbol{X} \in \mathbb{R}^{N \times m}$ with target sparsity s_0 such that $\boldsymbol{Y} \approx \boldsymbol{A}\boldsymbol{X}$, where $||\boldsymbol{E}||_F^2 = \sum_{i=1}^{n} \sum_{1}^{m} E_{ij}^2$ is the Frobenius norm and $||\boldsymbol{x}_i||_0$ is the number of nonzero elements of the i-th column of $\boldsymbol{X}$. For the representation performance of the algorithms described in this paper we consider the root mean squared error (RMSE), a measure that is built upon the Frobenius norm.

A popular and very efficient approach to solve this problem is the K-SVD algorithm [8]. This algorithm employs an alternate optimization process in two steps: keeping fixed the dictionary $\boldsymbol{A}$, create the sparse representation matrix $\boldsymbol{X}$ using the orthogonal matching pursuit (OMP) algorithm [9] and then in the second step, update both the dictionary and the representations using a rank-1 approximation obtained by the singular values decomposition (SVD). In order to speed up the procedure, OMP is implemented in a "bulk" fashion and the SVD step can be approximated with a few iterations of the power method [10]. Even though the results depend strongly on the initialization, in general this approach provides very good results in a reasonable amount of time.

In terms of the representation performance, it is clear that a bigger dictionary is better, meaning that it should reach a lower representation error than any strictly smaller dictionary. Since problem (1) is not convex, this does not always happen in practical designs. There is no general procedure and there are no guidelines when considering the values of N. Making the obvious choice of picking a very large N is not good since large dictionaries, even if they provide a very good presentation error, are just not feasible in many real world applications. In general, when K-SVD is applied in practice, the size N is

chosen by a heuristic or after conducting several numerical experiments because there is an important representation error/speed tradeoff that takes place. Consequently, the task of finding a good value of N when the restrictions on the representation errors are known is of great practical importance and the K-SVD method does not address it in any way.

Previous work related to the construction of size-optimized dictionaries includes: a K-SVD approach that gradually prunes the under-utilized or highly correlated atoms by using a competitive agglomeration algorithm approach [11] or one that also incorporates a subtractive clustering method to keep the most important atoms while it also prunes the redundant ones [12], the addition of criteria for eliminating redundant atoms by discarding linearly dependent atoms and adding linear independent features [13], selecting the atoms of the dictionary from a fixed set of candidate dictionaries by a greedy method [14] and segmentation of the training data into subspaces and constructing the dictionary by extracting similar features from multiple subspaces [15].

In this paper we propose a new way of constructing the dictionaries that relies on K-SVD steps but also addresses the issue of the size N. The dictionary is built from the ground up, starting with only s_0 atoms and gradually adding more columns by inspecting the worst constructed data items with the current dictionary configuration. At each step one or more atoms can be added to the current dictionary. This increase in size is followed by an adjustment of the atoms, all or only the new ones, using regular K-SVD steps. The procedure is halted when the desired representation error is reached. This way, the method decides on the size of the dictionary based on the prescribed representation performance. Numerical experiments show that the dictionaries designed this way are much smaller than the dictionaries designed with K-SVD that achieve the same representation errors.

The paper is organized in the following way: Section II details the steps of the proposed training procedure called Stagewise K-SVD, Section III describes the experimental results obtained on image data and Section IV concludes the paper.

II. The training procedure

The proposed method, Stagewise K-SVD, is presented next followed by a detailed discussion on how it builds a dictionary and the parameters that it involves. The main component of the procedure is the forward process that constructs the dictionary starting from a dictionary with only s_0 atoms. The procedure ends with a polishing process that tries to prune the less used atoms until the representation error constraint no longer holds.

Stagewise K-SVD (Given: the dataset $\boldsymbol{Y} \in \mathbb{R}^{n \times m}$, the target sparsity s_0, the target representation error ϵ_0, the number H of atoms to be added at each iteration and the interval R at which full iterations are applied. Return: dictionary $\boldsymbol{A} \in \mathbb{R}^{n \times N}$ and representation matrix $\boldsymbol{X} \in \mathbb{R}^{N \times m}$ such that $||\boldsymbol{Y} - \boldsymbol{A}\boldsymbol{X}||_F \leq \epsilon_0$.)

1) **Initialization**

   a) train initial dictionary $\boldsymbol{A}^0 \in \mathbb{R}^{n \times s_0}$ on the whole dataset using K-SVD with an initial dictionary created by picking s_0 random items from the dataset

2) **Forward iterative process**

   repeat from i = 0:

   a) create set $\mathbb{W}$ of the worst 5% reconstructed items from the dataset

   b) eliminate the less used atom to get $\tilde{\boldsymbol{A}}^i$

   c) compute H new atoms (columnwise concatenated in the matrix $\boldsymbol{U} \in \mathbb{R}^{n \times H}$) obtained by applying the SVD to the set $\mathbb{W}$ and keeping the left singular vectors associated with the H largest singular values, $\boldsymbol{A}^{i+1} = [\tilde{\boldsymbol{A}}^i \;\; \boldsymbol{U}]$

   d) at each R-th iteration apply the full K-SVD procedure on the dataset $\boldsymbol{Y}$ with the initial dictionary $\boldsymbol{A}^{i+1}$, otherwise apply the K-SVD by training only the new atoms (from $\boldsymbol{U}$) in the SVD step and reconstruct only the items from $\mathbb{W}$ in the OMP step (but use all available atoms)

   e) stop if $||\boldsymbol{Y} - \boldsymbol{A}^{i+1}\boldsymbol{X}^{i+1}||_F \leq \epsilon_0$

   f) $i \leftarrow i + 1$

3) **Backward iterative process**

   repeat:

   a) eliminate the less used atom of $\boldsymbol{A}^i$ to obtain $\boldsymbol{A}^{i+1}$ and train again using the full K-SVD

   b) stop if $||\boldsymbol{Y} - \boldsymbol{A}^{i+1}\boldsymbol{X}^{i+1}||_F > \epsilon_0$

   c) $i \leftarrow i + 1$

The proposed method tries to construct an efficient dictionary with a small number of atoms by an iterative process that increases the size of the dictionary until a specified error is reached. The idea is to extend the current dictionary by a fixed number of atoms such that the representation error is likely to be reduced substantially. The principle of Stagewise K-SVD seems to be the most natural, since the K-SVD steps work better in practice for small size dictionaries.

The initial dictionary $\boldsymbol{A}^0$ has the minimum number of atoms allowed such that a full representation in the OMP algorithm is possible with the given sparsity target s_0. The initial atoms are chosen at

random from the dataset. Another initialization point was constructed by considering the first s_0 principal components from the SVD of $\boldsymbol{Y}$, but the results did not show any significant improvement.

The SVD is applied to compute H new atoms, from the worst 5% reconstructed data items, in terms of the relative representation error $||\boldsymbol{y_i} - \boldsymbol{A}\boldsymbol{x_i}||/||\boldsymbol{y_i}||,\ 1 \leq i \leq m$; the 5% value was chosen after conducting numerical experiments. The new atoms are added to the current dictionary. Obviously, a smaller H is better in terms of representation performance but worse in terms of speed, since the iterative process runs more times. The K-SVD method is applied next on the extended dictionary in order to reduce the representation error. These runs will have only a few iterations since the representations change dramatically for only a fraction of the total items. Because of this we propose a parameter R that balances the performance and the speed of Stagewise K-SVD. For most of the iterations, a reduced K-SVD step is applied such that the training is made only for the H new atoms and reconstruction is made only for the items from the set $\mathbb{W}$. For large values of the parameter R, the size of the resulting dictionary is bigger but the design cost is relatively low, although the optimization procedure runs for more steps, since most iterations are reduced iterations that are very fast.

In the backward process we try to further decrease the size of the dictionary by choosing the less used atom and eliminating it. This atom is chosen by looking at the squared mean of its coefficients in the representations of the training vectors; a similar choice is made in step (b) of the forward process. The iterations stop as soon as the representation error is larger than the prescribed value. This step is necessary only for larger values of R and H.

Numerical experiments that validate the method are presented in the next section.

III. Experimental results

Following the training details provided in the literature we apply the Stagewise K-SVD method on the publicly available test images database [16]. From the dataset we extract randomly 11000 non-overlapping blocks of 8×8 pixels and apply the K-SVD algorithm with 441 atoms, a random initial dictionary and a target sparsity level of $s_0 = 6$ that is kept in all numerical experiments, achieving a RMSE of 0.04 as described in [1].

By comparison, the method proposed in [11] reaches the same error level with a dictionary trained with 179 atoms. The method, called EK-SVD, uses a completely different approach than the one presented in this paper. While Stagewise K-SVD constructs a dictionary by gradually increasing its size, EK-SVD starts with the maximum number of dictionary atoms and gradually prunes the under-utilized or highly correlated atoms by using a competitive agglomeration algorithm approach.

TABLE I

DICTIONARY SIZES ACHIEVED USING THE STAGEWISE K-SVD METHOD FOR TARGET RMSE ERROR 0.04. K-SVD REACHES THIS RMSE WITH 441 ATOMS WHILE EK-SVD NEEDS 179 ATOMS. THE BEST RESULT OF STAGEWISE K-SVD IS 175 ATOMS WHICH TRANSLATES INTO A 60% REDUCTION OF THE INITIAL DICTIONARY.

		R				
		1	2	3	4	5
H	2	175	197	212	222	219
	3	189	197	205	211	226
	4	194	225	243	226	258
	5	199	224	238	239	246
	6	199	228	244	241	318

The results obtained when applying the Stagewise K-SVD method are shown in Table I and it is clear that Stagewise K-SVD substantially outperforms K-SVD in terms of the quality of the produced dictionary and is a good alternative to EK-SVD.

When Stagewise K-SVD is applied with the larger values for the involved parameters, $R = 5$ and $H = 6$, it outperforms K-SVD both in the running time, which is 60% of the running time of K-SVD, and in the representation capabilities of the constructed optimized dictionary, that is still 30% smaller. In the case where the best performance is obtained, for parameters $R = 1$ and $H = 2$, the optimized dictionary is 60% smaller but at the expense of a larger running time, 4.5 longer than the running time of K-SVD. A comparison with EK-SVD is impossible, since the running times are not reported in [11].

Further numerical experiments are made to check the efficiency of Stagewise K-SVD when compared to dictionaries of different lengths designed with K-SVD. The results in Table II describe one set of runs and they show how Stagewise K-SVD behaves when compared to dictionaries that are k times overcomplete ($\boldsymbol{A} \in \mathbb{R}^{n \times kn}$) while achieving lower representation errors, under various values of H and R.

We compare the results of Stagewise K-SVD with two variants of the K-SVD algorithm: one where K-SVD is initialized by a random matrix and the other where the initialization matrix is constructed by extracting random items from the dataset. In our context, when designing smaller dictionaries (2-3 times overcomplete) the initialization using random data items outputs better performance since atoms will contain relevant directions while for larger dictionaries initialization with a random matrix adds more diversity to the dictionary and thus better performance is reached. The performance is also computed as the ratio between the actual size achieved by Stagewise K-SVD and the initial dictionary size (smaller

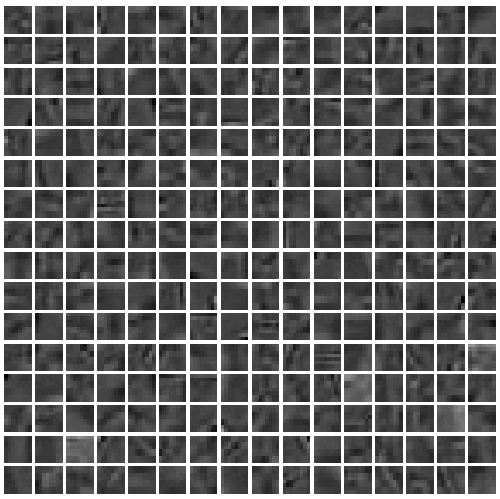

Fig. 1. Atoms trained using the K-SVD algorithm for $N_0 = 256$.

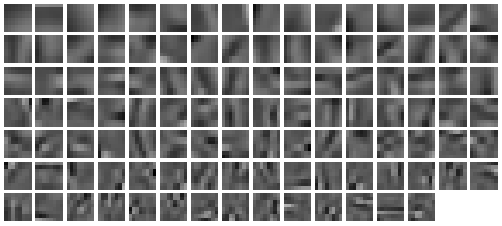

Fig. 2. Atoms trained using Stagewise K-SVD with $N = 110$ and the target representation error of K-SVD for $N_0 = 256$.

means better). The dictionaries computed by Stagewise K-SVD are clearly smaller than their K-SVD counterparts overall and, in the best case scenario, they are smaller by approximately 50%.

Simulations indicate that all the results are heavily dependent on the initialization points. This makes the usage of K-SVD and its comparison to other methods very difficult. The utilization of Stagewise K-SVD eliminates the need to find a good initialization point while it is able to provide good performance.

Since each step of Stagewise K-SVD helps the reconstruction of the worst data items, an extra benefit is that the maximum representation error and the mutual coherence of the atoms are also lower when compared with the regular K-SVD. For example, in the first test case, the dictionary designed by the full

K-SVD has mutual coherence 0.998 while the one designed with Stagewise K-SVD achieves a lower mutual coherence of 0.782. As Stagewise K-SVD gradually builds the dictionary the mutual coherence increases in order for the dictionary to improve its representation capabilities. The computed atoms for one of the test cases are displayed in Images 1 and 2.

From the simulations it can be observed that worst performance is generally reached when values of H and R increase (a fact that is expected) but the performance is better as k increases, showing that for large overcomplete dictionaries K-SVD fails to design a dictionary with good representation error as compared with smaller dictionaries. This observation validates the approach taken by Stagewise K-SVD to start building large dictionaries from the ground up.

IV. Conclusions

This paper presents a new and conceptually simple method called Stagewise K-SVD to design dictionaries for sparse representations with an optimized number of atoms. The method does not rely on being supplied the number of atoms but rather the target representation error, which is available in practice. The core of the method relies on K-SVD steps that have shown to work very well in practical applications. The efficiency of the method is shown in numerical experiments on image data where Stagewise K-SVD designs dictionaries much smaller than K-SVD while keeping lower representation errors.

V. Acknowledgements

This work was supported by the Romanian National Authority for Scientific Research, CNCS UEFISCDI, project number PN-II-ID-PCE-2011-3-0400 and by the Sectoral Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labour, Family and Social Protection through the Financial Agreement POSDRU/88 /1.5/S/61178.

References

[1] A. M. Bruckstein, D. L. Donoho and M. Elad, From Sparse Solutions of Systems of Equations to Sparse Modeling of Signals and Images, SIAM Review, Vol. 51, pp. 34–81, 2009.

[2] E. J. Candes and T. Tao, Decoding by linear programming, IEEE Trans. Inform. Theory, Vol. 51, pp. 4203–4215, 2004.

[3] R. G. Baraniuk, E. Candes, M. Elada and M. Yi, Applications of Sparse Representation and Compressive Sensing, Proceedings of the IEEE, Vol. 98, pp. 906–909, 2010.

[4] J. Mairal, M. Elad and G. Sapiro, Sparse Representation for Color Image Restoration, IEEE Transactions on Image Processing, Vol. 17, pp. 53–69, 2008.

[5] R. Neff and A. Zakhor, Very Low Bit-Rate Video Coding Based on Matching Pursuits, IEEE Trans. Circuits and Systems, Vol. 7, pp. 158–171, 1997.

[6] Y. Zang, S. Mei, Q. Chen and Z. Chen, A novel image/video coding method based on Compressed Sensing theory, IEEE International Conference on Acoustics Speech and Signal Processing, pp. 1361 – 1364, 2008.

[7] Z. Qiang and L. Baoxin, Discriminative K-SVD for dictionary learning in face recognition, IEEE Conference on Computer Vision and Pattern Recognition, pp. 2691–2698, 2010.

[8] M. Aharon, M. Elad and A. Bruckstein, K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation, IEEE Trans. Signal Processing, Vol. 54, pp. 4311–4322, 2006.

[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 Comput., vol.1, pp. 40-44, 1993.

[10] R. Rubinstein, M. Zibulevsky and M. Elad, Efficient Implementation of the K-SVD Algorithm using Batch Orthogonal Matching Pursuit Technical Report - CS Technion, 2008.

[11] R. Mazhar and P. D. Gader, EK-SVD: Optimized dictionary design for sparse representations, 19th International Conference on Pattern Recognition, pp. 1–4, 2008.

[12] J. Feng, L. Song, X. Yang and W. Zhang, Sub clustering K-SVD: Size variable Dictionary learning for Sparse Representations, 16th IEEE International Conference on Image Processing, pp. 2149–2152, 2009.

[13] L. Rebollo-Neira, Dictionary redundancy elimination, IEEE Proc., Vis. Image Process., Vol. 151, pp. 31–34, 2004.

[14] V. Cevher and A. Krause, Greedy Dictionary Selection for Sparse Representation, IEEE Journal of Selected Topics in Signal Processing, Vol. 5, pp. 979–988, 2011.

[15] J. Feng, L. Song, X. Yang and W. Zhang, Learning dictionary via subspace segmentation for sparse representation, 18th IEEE International Conference on Image Processing, pp. 1245–1248, 2011.

[16] Yale Face Database. [Online]. Available: http://cvc.yale.edu/projects/ yalefaces/yalefaces.html.

TABLE II

DICTIONARY SIZES N REACHED USING THE STAGEWISE K-SVD METHOD WITH SEVERAL VALUES OF H AND R THAT ACHIEVE THE SAME REPRESENTATION ERRORS AS FOR DIFFERENT k OVERCOMPLETE ($N_0 = k \times 64$) DICTIONARIES TRAINED WITH K-SVD STARTED WITH A RANDOM DICTIONARY. THE REPRESENTATION ERRORS ACHIEVED BY THE K-SVD ALGORITHM, IN INCREASING ORDER OF DICTIONARY SIZE ARE: $\{0.059, 0.051, 0.045, 0.033\}$ FOR THE RANDOM START RUN AND $\{0.053, 0.049, 0.047, 0.037\}$ FOR THE RUN HAVING AS STARTING POINT RANDOM ITEMS FROM THE DATASET.

			random start		random data item start	
N_0	H	R	N	N/N_0	N	N/N_0
128	2	1	71	55.74%	91	**71.09%**
		2	76	59.38%	94	73.44%
		3	73	57.03%	97	75.78%
	3	1	77	60.16%	101	78.91%
		2	70	**54.69%**	93	72.66%
		3	74	57.81%	102	79.69%
	4	1	74	57.81%	92	71.88%
		2	84	65.63%	107	83.59%
		3	84	65.63%	109	85.16%
192	2	1	98	**51.04%**	109	56.77%
		2	114	59.38%	112	58.33%
		3	105	54.69%	120	62.50%
	3	1	101	52.60%	107	**55.72%**
		2	110	57.29%	112	58.33%
		3	114	59.38%	114	59.38%
	4	1	102	53.13%	111	57.81%
		2	119	61.98%	119	61.98%
		3	129	67.19%	130	67.70%
256	2	1	110	**42.97%**	114	44.53%
		2	129	50.39%	112	**43.75%**
		3	129	50.39%	122	47.66%
	3	1	117	45.70%	123	48.05%
		2	122	47.66%	119	46.48%
		3	146	57.03%	137	53.52%
	4	1	125	48.83%	117	45.70%
		2	139	54.30%	127	49.61%
		3	149	58.20%	132	51.56%
512	2	1	212	**41.41%**	207	**40.43%**
		2	236	46.09%	229	44.73%
		3	248	48.44%	245	47.85%
	3	1	239	46.68%	229	44.73%
		2	254	47.66%	227	44.34%
		3	282	49.22%	252	49.22%
	4	1	251	47.07%	228	44.53%
		2	279	50.59%	253	49.41%
		3	273	53.32%	260	50.78%