The Effect of Atom Replacement Strategies on Dictionary Learning

Paul Irofti[1].

[1] University Politehnica of Bucharest, Romania.

Abstract— **The sparse representations field presents a wide set of algorithms for learning overcomplete dictionaries. During the learning process many of the dictionary columns remain unused by the resulting representations. In this paper we present a few replacement strategies and their direct impact on a set of popular algorithms such as K-SVD. Experiments show significant reductions in the representation error and also evidentiate clear differences between the strategies.**

1 Introduction

The signal processing domain presents an increased interest for sparse representations through overcomplete dictionary learning (DL)[1, 2], which showed significant improvements compared to fixed dictionaries built from various transform basis.

Starting with the matrix $Y \in \mathbb{R}^{p\times m}$, built from m training signals of dimension p, and a sparsity target s we aim to solve the optimization problem

$$\begin{aligned} \underset{D,X}{\text{minimize}} \quad & \|Y - DX\|_F^2 \\ \text{subject to} \quad & \|d_j\|_2 = 1,\ 1 \le j \le n \\ & \|x_i\|_0 \le s,\ 1 \le i \le m, \end{aligned} \tag{1}$$

where the variables are the dictionary $D \in \mathbb{R}^{p\times n}$, whose n columns d_j are called atoms, and the sparse representations $X \in \mathbb{R}^{n\times m}$. The columns x_i of X are at most s-sparse as denoted by $\|\cdot\|_0$ which is the l_0 pseudo-norm. The atoms are normalized so that the multiplicative indetermination of the DX factorization is removed.

Equation (1) is non-convex and bilinear which is why DL methods approach it in two stages First, the dictionary D is fixed and the representations are computed. This is a hard combinatorial problem that can be solved through greedy algorithms. The popular choice is orthogonal matching pursuit (OMP)[3]. Next, the resulting representations are kept fixed and the dictionary is updated to reduce the approximation error.

Investigating the sparsity pattern provided by OMP we can see how many times each dictionary atom is used for representation. While popular atoms help with classification and compression, what about the ones that are rarely or never used?

In this paper we focus on different strategies for replacing unused atoms with new ones leading to an overall improvement of the DL process. In Section 2 we briefly present existing dictionary update methods followed by Section 3, where we describe and present numerical results with different replacement strategies, and afterwards we conclude in Section 4.

2 DL algorithms

K-SVD[4] updates the atoms in sequence following the Gauss-Seidel approach: the current atom is refined using information from the atoms that were previously updated in the current stage. K-SVD solves the optimization problem

$$\min_{d_j, X_{j,\mathcal{I}_j}} \left\| \left(Y - \sum_{\ell \neq j} d_\ell X_{\ell,\mathcal{I}_\ell} \right) - d_j X_{j,\mathcal{I}_j} \right\|_F^2, \tag{2}$$

where $\mathcal{I}_j$ is the sparse signals subset that use atom j in their representation, $X_{j,\mathcal{I}_j}$ denotes row j from the sparse representations restricted to the columns in $\mathcal{I}_j$, and all atoms excepting d_j are fixed. Problem (2) is a rank-1 approximation whose solution is given by the singular vectors corresponding to the largest singular value. Note that the solution updates both the atom and the associated representations.

Approximate K-SVD (AK-SVD)[5] is a faster version that computes the singular vectors through a single iteration of the power method

In order to guarantee certain properties, such as clustering, sequential generalization of K-means (SGK) [6] preserves the sparse structure and proposes a similar optimization problem that refines only the atom d_j. This can be reduced to a simple least-squares problem.

NSGK [7] follows the same strategy, but accounts for differences between the previous and current values of the dictionary and representations. Empirical evidence indicates that NSGK provides better results than the former methods.

3 Replacement Strategies

The sequential nature of the dictionary update allows us to replace unused atoms during refinement as soon as we encounter them.

Let us assume that the default action is to leave the atom as it is. Numerical evidence shows that once an atom stops being used it never gets picked up again, so we could just remove it and shrink the dictionary. While this does not affect the approximation it does improve performance and storage.

Given that common practice dictates that we start with a random dictionary, a natural idea is to substitute with a new randomly generated column. Another option is to find the worst represented signal and make it a part of the dictionary (marked ’Worst’ in our tables and figures). In our experiments we also performed this replacement in bulk at the end of the dictionary refinement stage. We label this ’Post’.

We built the signal set Y from random 8×8 image patches taken from the USC-SIPI [8] database. The results presented here are the average of executing 10 runs of each method with the same parametrization. Every algorithm performed 50 DL iterations before stopping.

We compare the approximation of the original signals by the dictionary and sparse representations through the root mean square error RMSE $= \frac{\|Y-DX\|_F}{\sqrt{pm}}$.

In Table 1 we show the DL results on a set of $m = 2048$ signals of size $p = 64$ each, with a sparsity constraint $s = 8$,

Table 1: Final RMSE with various replacement strategies

n	Method	*Replacement*			
		None	Random	Worst	Post
128	K-SVD	0.029406	0.026723	**0.019096**	0.019355
	AK-SVD	0.029497	0.026876	**0.019134**	0.019369
	SGK	0.029402	0.026800	**0.019079**	0.019612
	NSGK	0.025004	0.024707	**0.020558**	0.019804
192	K-SVD	0.030805	0.027013	0.018812	**0.018803**
	AK-SVD	0.031010	0.026661	**0.018804**	0.018835
	SGK	0.031328	0.026938	**0.018799**	0.018885
	NSGK	0.023911	0.025168	0.021471	**0.019271**
256	K-SVD	0.029146	0.023979	0.016773	**0.016730**
	AK-SVD	0.029291	0.024300	**0.016843**	0.016858
	SGK	0.029836	0.024201	0.016928	**0.016923**
	NSGK	0.022099	0.022590	0.020913	**0.017563**
384	K-SVD	0.024984	0.018467	0.012522	**0.011930**
	AK-SVD	0.024721	0.018697	0.012665	**0.011972**
	SGK	0.024693	0.019048	0.012791	**0.011946**
	NSGK	0.016995	0.016820	0.018473	**0.012650**
512	K-SVD	0.025699	0.017744	0.012343	**0.011034**
	AK-SVD	0.025318	0.018016	0.012436	**0.011078**
	SGK	0.025617	0.017668	0.012579	**0.011072**
	NSGK	0.016795	0.016769	0.017951	**0.012128**

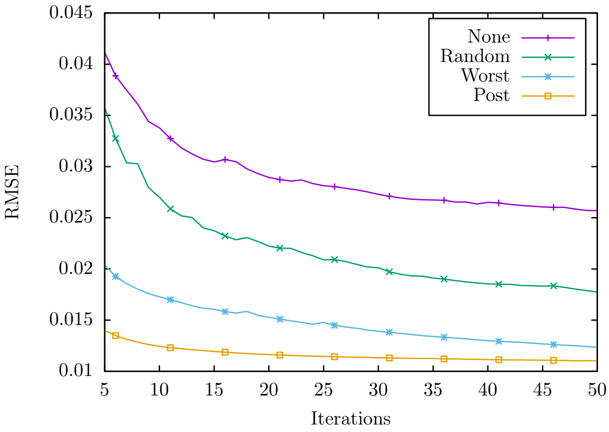

Figure 1: Error descent averaged over 10 runs for K-SVD ($n = 512$).

while varying the dictionary size. We can see that, except for a few NSGK results, replacing the unused atoms provides much better approximations. While using random atoms for substitution improves the situation, choosing the worst represented data item instead is clearly the best choice. The difference comes with dictionary size: smaller dictionaries prefer substitution during refinement ($n = 128$ and part of $n = 256$), while larger ones show smaller errors by replacing all unused atoms post-refinement.

Figure 1 shows the average error evolution of K-SVD with different atom replacement strategies. Employing post-refinement substitution is the clear winner at every iteration, followed by performing signal substitution during dictionary update. Random replacement comes in last, but is clearly ahead of the plain K-SVD version.

In Figure 2 we present the average error curve for AK-SVD dictionaries of size $n = 256$. This case is particularly interesting because post-refinement substitution starts ahead but is caught up and beaten by substitution during dictionary update. Even though both are well behind, we can see that random replacement is better than no replacement.

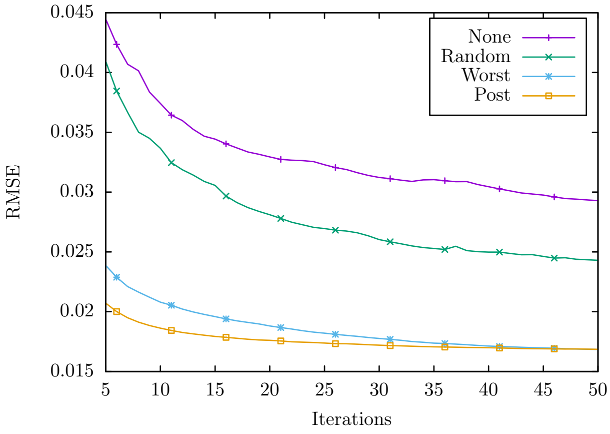

Figure 2: Error descent averaged over 10 runs for AK-SVD ($n = 256$).

4 Conclusion

We presented the impact of different atom replacement techniques on the DL process. Numerical results have shown, with small exceptions, that substitution has a significant impact on the approximation error. Our experiments suggest that replacing unused atoms with the worst represented signals from the training set is the best approach.

Acknowledgements

This work was supported by the Romanian National Authority for Scientific Research, CNCS - UEFISCDI, project number PN-II-RU-TE-2014-4-2713.

References

[1] R. Rubinstein, A.M. Bruckstein, and M. Elad, “Dictionaries for Sparse Representations Modeling,” *Proc. IEEE*, vol. 98, no. 6, pp. 1045–1057, June 2010.

[2] I. Tosic and P. Frossard, “Dictionary Learning,” *IEEE Signal Proc. Mag.*, vol. 28, no. 2, pp. 27–38, Mar. 2011.

[3] Y.C. Pati, R. Rezaiifar, and P.S. Krishnaprasad, “Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition,” in *27th Asilomar Conf. Signals Systems Computers*, Nov. 1993, vol. 1, pp. 40–44.

[4] M. Aharon, M. Elad, and A.M. Bruckstein, “K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation,” *Signal Processing, IEEE Transactions on*, vol. 54, no. 11, pp. 4311–4322, 2006.

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

[6] S.K. Sahoo and A. Makur, “Dictionary training for sparse representation as generalization of k-means clustering,” *Signal Processing Letters, IEEE*, vol. 20, no. 6, pp. 587–590, 2013.

[7] M. Sadeghi, M. Babaie-Zadeh, and C. Jutten, “Dictionary Learning for Sparse Representation: a Novel Approach,” *IEEE Signal Proc. Letter*, vol. 20, no. 12, pp. 1195–1198, Dec. 2013.

[8] A.G. Weber, “The USC-SIPI Image Database,” 1997.