The Adaptive Dictionary Learning Toolbox

Cristian Rusu
LCSL – IIT & MIT
Via Morego 30, 16163 Genova GE
cristian.rusu@iit.it

Karin Schnass
University of Innsbruck
Technikerstraße 13, A-6020 Innsbruck
karin.schnass@uibk.ac.at

I. Introduction

Since Olshausen and Field, [1], MOD, [2] and K-SVD, [3] a lot of research has been invested in the development of dictionary learning algorithms. Most proposed algorithms naturally display modular structure and have numerous parameters that need to be chosen such as dictionary size and sparsity level. A popular technique for dictionary learning is alternating optimization, which updates the dictionary atoms and the sparse representations alternatively while the other is fixed. Due to the complexity of the learning problem, over time numerous dictionary update rules and, independently from the dictionary learning problem, sparse approximation algorithms have been developed without a clearly superior solution for all learning scenarios. This leads to choice overload.

As such, it is becoming increasingly hard to decide which choices are good, because in many instances researchers compare apples and oranges, for example: different dictionary sizes (comparing to a K-SVD dictionary with redundancy 4, as done in the original paper, to a basis), not removing the mean of the data (which for sure works better on some data), and replacing modules (mostly in K-SVD) without regard to the trade-off between computational cost and precision, e.g. from a numerical perspective, it is not always clear where the computational bottleneck lies: for example, in K-SVD, replacing the SVD update with a power iteration improves the running time since SVDs are more expensive but it is actually Orthogonal Matching Pursuit (OMP) that dominates the complexity of the algorithm. On the other hand, replacing OMP with mixed integer programming for increased precision is like dropping an atomic bomb on a flea.

II. An adaptive dictionary learning toolbox

It is high time to provide a toolbox that makes everything comparable, to see which parts of newly proposed dictionary learning methods are faster, more accurate, etc. Our focus is on batch alternating optimization algorithms (sparse coding step alternates with dictionary update step) but it will be possible to integrate any algorithm into the proposed framework. The main features of the toolbox are:

- A plug-and-play system for the dictionary update and sparse approximation steps:
  - sparse coding algorithms: OMP (with various stopping criteria), thresholding, Hard Thresholding Pursuit (HTP), Adaptive Pursuit (AP) a mix of OMP and HTP algorithms designed for increased speed and without the need of the sparsity level as input.
  - classic dictionary update routines: Olshausen-Field, MAP, MOD, SVD, ITKrM and structured dictionary update rules.
  - several dictionary initialization techniques.

  Therefore, the toolbox is able to reproduce the most popular dictionary learning algorithms and allows programmers to easily test new ideas to improve them.
- Inspired by [4], [5] we also provide the possibility to make all the above algorithms dictionary size adaptive. To this end, we include several routines to remove and train replacement/additional atoms.
- For fixed-sized dictionaries, adaptive identification, and update of atoms that perform badly.
- Adaptive sparsity for the whole dataset and for data item separately (of course dependent on the algorithm used in the sparse approximation step).
- Learning with coherence constraints on the dictionary.
- Synthetic dataset generation and recovery (to evaluate the recovery rate of a particular dictionary learning algorithm).

Some algorithms might have a sequential (or online) implementation (for example, ITKrM or power iteration K-SVD) since this is a MATLAB toolbox, we decided against sequential implementations and always use batch processing of the dataset.

The proposed toolbox addresses two communities:

1) The dictionary learning community for which we hope to provide a framework where new ideas can be easily developed and tested, with an eye towards reproducibility.
2) General researchers who want to use dictionary learning techniques in their applications for which we hope to provide a toolkit that includes advanced learning methods (beyond just the alternating optimization steps) in a relatively simple way.

III. Results and conclusions

We provide experimental results in the figures below, combining several dictionary update rules and sparse approximation algorithms together with adaptive sparsity and dictionary size selection.

For the future, we hope that other dictionary learning researchers will contribute their ideas and source code to the toolbox and use it to compare against previous approaches.

Our immediate goal is to also clone the toolbox for the python programming language. All implementations are publicly available on the GitHub page https://github.com/cristian-rusu-research/adl-toolbox

References

[1] D. Field and B. Olshausen, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, pp. 607–609, 1996.

[2] K. Engan, S. Aase, and J. Husoy, "Method of optimal directions for frame design," in *ICASSP99*, vol. 5, 1999, pp. 2443 – 2446.

[3] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing.*, vol. 54, no. 11, pp. 4311–4322, November 2006.

[4] C. Rusu and B. Dumitrescu, "Stagewise K-SVD to design efficient dictionaries for sparse representations," *IEEE Signal Processing Letters*, vol. 19, no. 10, pp. 631–634, 2012.

[5] K. Schnass, "Dictionary learning - from local towards global and adaptive," *arXiv:1804.07101*, 2018.

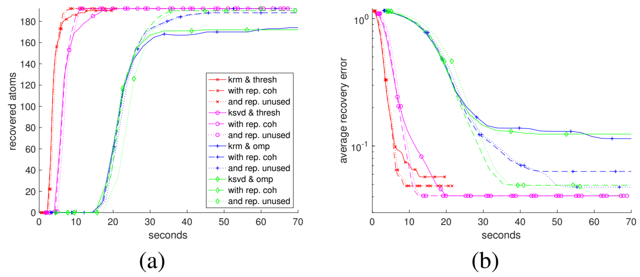

Fig. 1. Learning dictionaries on synthetic data, consisting of 10000 signals that are 6-sparse with respect to a dictionary of 192 randomly chosen atoms from the DCT and Dirac bases in $\mathbb{R}^{128}$. Each algorithm runs for 50 iterations, either not replacing atoms, replacing coherent atoms $\mu > 0.7$ or additionally atoms that are used less than 50 times. The sparse coding step uses either thresholding or OMP, and the dictionary update step uses K SVDs or K residual means. Replacement atoms are produced by learning a dictionary of 6 atoms with sparsity one via K-SVD/KRM on the residuals. An atom is considered recovered if has inner product > 0.99 with a learned atom.

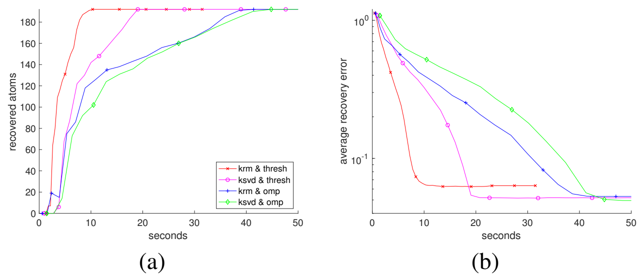

Fig. 2. In a set-up similar to 1 but sparsity varying between 4 and 8 we learn the dictionaries adaptively starting with initial size d and initial sparsity 1. The sparse approximation routines are slightly modified versions of thresholding and OMP. Coherent and unused atoms are pruned, and promising candidate atoms added. All routines recover the correct dictionary and size.

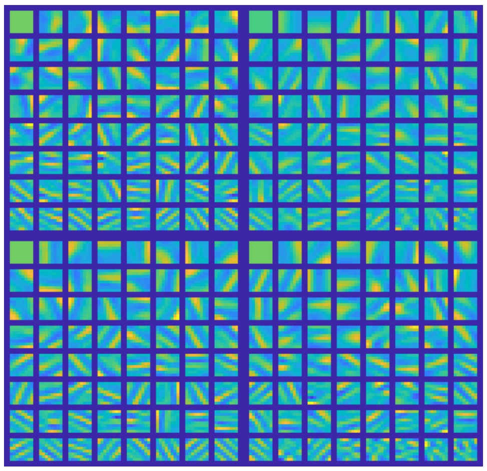

Fig. 3. Atoms from learning dictionaries on image data, consisting of 62001 patches from the mandrill image that are 3-sparse with respect to a dictionary of size 64. Training is done with four algorithms: KRM & thresholding (upper left), KRM & OMP (upper right), SVD & thresholding (lower right) and SVD & OMP (i.e., K-SVD in the lower left). All algorithms run for 50 iterations, replacing coherent atoms $\mu > 0.7$ and unused atoms.

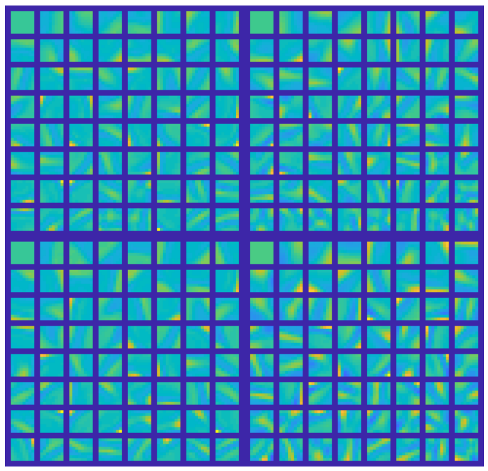

Fig. 4. Analogous to Figure 3 but for the peppers image.

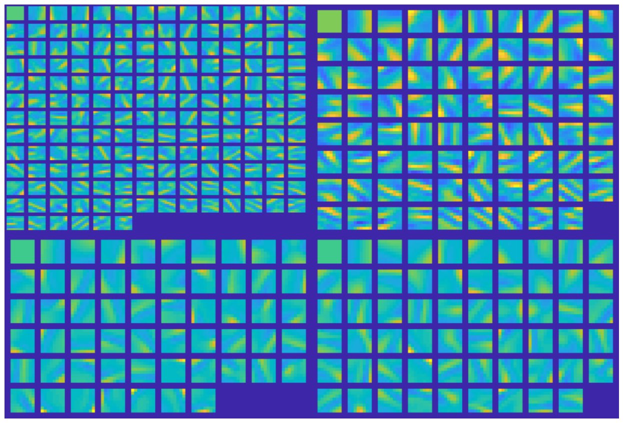

Fig. 5. Atoms from learning dictionaries on image data: mandrill (top) and peppers (bottom). Two algorithms used are KRM + adaptive pursuit (left) and K-SVD (right). Each algorithm runs for 60 iterations, replacing coherent atoms $\mu > 0.7$, using adaptive sparsity and dictionary size following ideas from [4], [5]. Average sparsity is: 3.3 for mandrill with KRM & AP, 1.9 for mandrill with K-SVD; 5.1 for peppers with KRM & AP, 3.2 for peppers with K-SVD.

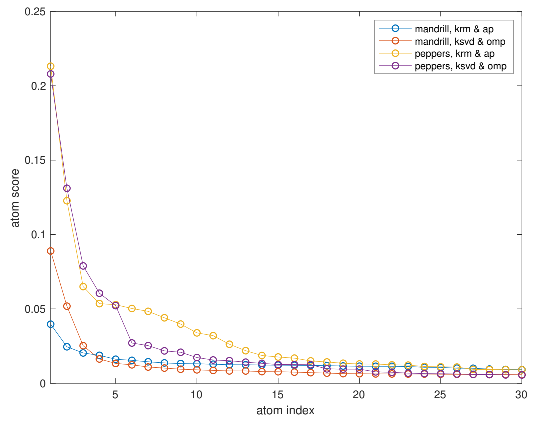

Fig. 6. Atom scores for the first 30 atoms of each of the four dictionaries in Figure 5, sorted decreasing. The scores are normalized squared sums of the coefficients used by the atoms in the sparse representations.