Convolutional Matching Pursuit and Dictionary Training

Arthur Szlam, Koray Kavukcuoglu, and Yann LeCun

1 Introduction

One of the most succesful recent signal processing paradigms has been the sparse coding/dictionary design model [8, 4]. In this model, we try to represent a given $d \times n$ data matrix X of n points in $\mathbb{R}^d$ written as columns via a solution to the problem

$$\{W_*, Z_*\} = \{W_*(K, X, q), Z_*(K, X, q)\}$$
$$= arg \min_{Z \in \mathbb{R}^{K \times n}, W \in \mathbb{R}^{d \times K}} \sum_k ||Wz_k - x_k||^2, \ ||z_k||_0 \leq q, \tag{1.1}$$

or its Z coordinate convexification

$$\{\tilde{W}_*, \tilde{Z}_*\} = \{\tilde{W}_*(K, X, \lambda), \tilde{Z}_*(K, X, \lambda)\}$$
$$= arg \min_{Z \in \mathbb{R}^{K \times n}, W \in \mathbb{R}^{d \times K}} \sum_k ||Wz_k - x_k||^2 + \lambda ||z_k||_1. \tag{1.2}$$

Here, $\{W, Z\}$ are the dictionary and the coefficients, respectively, and z_k is the kth column of Z. K, q, and λ are user selected parameters controlling the power of the model.

More recently, many models with additional structure have been proposed. For example, in [9, 2], the dictionary elements are arranged in groups and the sparsity is on the group level. In [3, 5, 7], the dictionaries are constructed to be translation invariant. In the former work, the dictionary is constructed via a non-negative matrix factorization. In the latter two works, the construction is a convolutional analogue of 1.2 or an l^p variant, with $0 < p < 1$. In this short note we work with greedy algorithms for solving the convolutional analogues of 1.1. Specifically, we demonstrate that sparse coding by matching pursuit and dictionary learning via K-SVD [1] can be used in the translation invariant setting.

2 Matching Pursuit

Matching pursuit [6] is a greedy algorithm for the solution of the sparse coding problem

$$\min_z ||Wz - x||^2,$$
$$||z||_0 \leq q,$$

where the $d \times k$ matrix W is the dictionary, the $k \times 1$ z is the code, and x is an $d \times 1$ data vector.

1. Set $e = x$, and z the k-dimensional zero vector.
2. Find $j = \arg\max_i ||W_i^T e||_2^2$.
3. Set $a = W_j^T x$.
4. Set $e \leftarrow e - aW_j$, and $z_j = z_j + a$.
5. Repeat for q steps

Note that with a bit of bookkeeping, it is only necessary to multiply W against x once, instead of q times. This at a cost of an extra $O(K^2)$ storage: set e_r and a_r be e and a from the rth step above. Then:

$$W^T e_0 = W^T x;$$

$$W^T e_1 = W^T x - a_0 W^T W_{j_0},$$

and so on. If the Gram matrix for W is stored, this is just a lookup.

2.1 Convolutional MP

We consider the special case

$$\min_z ||\sum_{j=1}^{k} w_j * z_j - x||^2,$$

$$||\overline{z}||_0 \leq q,$$

where each w_j is a filter, and $\overline{z}$ is all of the responses.

Note that the Gram matrix of the "Toeplitz" dictionary consisting of all the shifts of the w_j is usually too big to be used as a lookup table. However, because of the symmetries of the convolution, it is also unnecessary; we only need store a $4 * h_f \times w_f \times k^2$ array of inner products, where h_f and w_f are the dimensions of the filters.

With this additional storage, to run q basis pursuit steps with k filters on an $h \times w$ image costs the computation of one application of the filter bank plus $O(kqhw)$ operations.

3 Learning the filters

Given a set of x, we can learn the filters and the codes simultaneously. Several methods are available. A simple one is to alternate between updating the codes and updating the filters, as in K-SVD [1]:

1. Initialize k $h_f \times w_f$ filters $\{w_1, ..., w_k\}$.
2. Solve for z as above.

3. For each filter w_j,

   - find all locations in all the data images where w_j is activated
   - extract the $h_f \times w_f$ patch E_p from the reconstruction via z at each activated point p.
   - remove the contribution of w_j from each E_p (i.e. $E_p \leftarrow E_p - c_{(p,j)} w_j$, where $c_{(p,j)}$ was the activation determined by z).
   - update $w_j \leftarrow \text{PCA}(E_p)$

4. Repeat from step 2 until fixed number of iterations.

We note that the forward subproblem (finding Z with W fixed) is not convex, and so the alternation is not guaranteed to decrease the energy or to converge to even a local minimum. However, in practice, on image and audio data, this method generates good filters.

4 Some experiments

We train filters on three data sets: the AT&T face database, the motorcycles from a Caltech database, and the VOC PASCAL database. For all the images in all our experiments, we perform an additive contrast normalization: each image x is transformed into $x' = x - x * b$, where b is a 5×5 averaging box filter. This is very nearly transforming $x' = \nabla^2 x$, that is, using the discrete Laplacian of the image instead of the image. Using the Laplacian would correspond to using the energy

$$\sum_x ||\nabla \left(\sum_j w_j * z_j - x \right) ||^2,$$

that is, the energy sees the difference between gradients, not intensities.

4.1 Faces

The AT&T face database, available at `http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase` is a set of 400 images of 40 individuals. The faces are centered in each image. We resize each image to 64×64 and contrast normalize. We train 8 16×16 filters. After training the filters we find the feature maps of each image in the database, obtaining a new set of 400 8 channel images. We take the elementwise absolute value of each of the 8 channel images, and then average pool over 8×8 blocks. We then train a new 16 element dictionary on the subsampled images. In figure 1 we display the first level filters, and the second level filters up to shifts of size 8 and sign changes of the first level filters..

Figure 1: First and second layer filters from faces

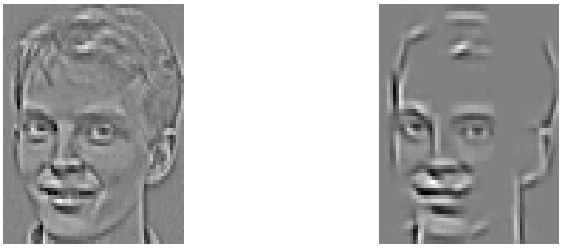

Figure 2: a contrast normalized face, and its reconstruction from 40 filter responses.

4.2 Caltech motorcycles

We also train on the motorbikes-side dataset, available at `http://www.vision.caltech.edu/html-files/arch` which consists of color images of various motorcycles. The motorcycles are centered in each image. We convert each image to gray level, resize to 64×64, and contrast normalize. We train 8 16×16 filters. As before, we then train a new 16 element dictionary on the subsampled absolute value rectified responses of the first level. In figure 3 we display the first level filters, and the second level filters up to shifts of size 8 and sign changes of the first level filters..

Figure 3: First and second layer filters from motorcycles

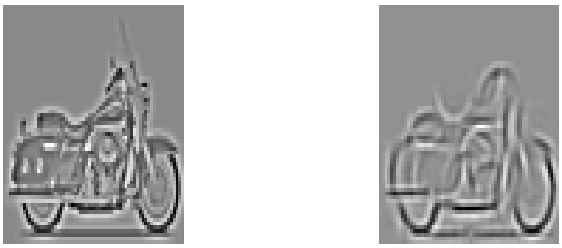

Figure 4: A contrast normalized motorcycle, and its reconstruction from 40 filter responses.

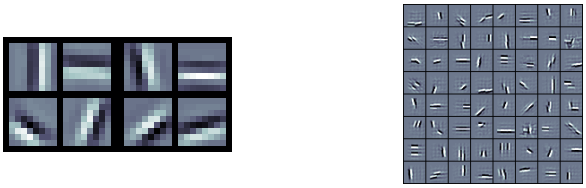

Figure 5: First and second layer filters from natural images

4.3 Images from PASCAL VOC

We also show results trained on "unclassified" natural images from the PASCAL visual object challenge dataset available at `http://pascallin.ecs.soton.ac.uk/challenges/VOC/`. We randomly subsample 5000 grayscaled images by a factor of 1 to 4, and then pick from each image a 64×64 patch, and then contrast normalize. We train 8 8×8 filters. We then train a new 4×4 64 element dictionary on the subsampled absolute value rectified responses of the first level. In figure 5 we display the first level filters, and the second level filters up to shifts of size 8 and sign changes of the first level filters.

In order to show the dependence of the filters on the number of filters used, in figure 6 we display an 8, 16, and 64 element 16×16 dictionary trained on the same set as above.

References

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

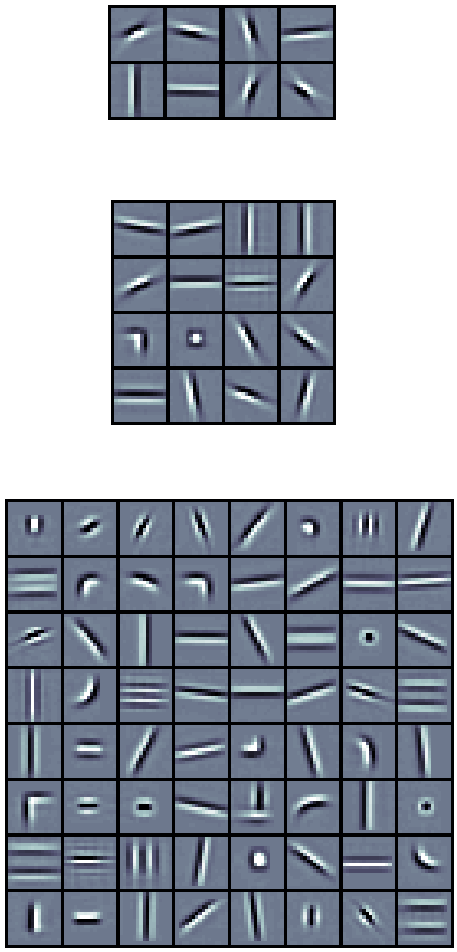

Figure 6: Dictionaries with varying numbers of elements trained on natural images.

[2] Francis R. Bach. Consistency of the group lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1225, 2008.

[3] Sven Behnke. Discovering hierarchical speech features using convolutional non-negative matrix factorization. In *IJCNN*, pages 7–12, 2008.

[4] Alfred M. Bruckstein, David L. Donoho, and Michael Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM Review*, 51(1):34–81, 2009.

[5] Y. Boureau K. Gregor M. Mathieu Y. LeCun K. kavukcuoglu, P. Sermanet. Learning convolutional feature hierarchies for visual recognition. *Advances in NIPS*, 2010.

[6] Stephane Mallat and Zhifeng Zhang. Matching pursuit with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41:3397–3415, 1993.

[7] Graham Taylor Matthew Zeiler, Dilip Krishnan and Rob Fergus. Hierarchical convolutional sparse image decomposition. In *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, June 2010.

[8] B. Olshausen and D. Field. Sparse coding with an overcomplete basis set: A strategy employed by v1?, 1997.

[9] Ming Yuan, Ming Yuan, Yi Lin, and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.