IMAGE COMPRESSION USING LEARNED DICTIONARIES BY RLS-DLA AND COMPARED WITH K-SVD

Karl Skretting and Kjersti Engan

University of Stavanger
Department of Electrical Engineering and Computer Science
4036 Stavanger, Norway. *karl.skretting@uis.no*

ABSTRACT

The recently presented recursive least squares dictionary learning algorithm (RLS-DLA) is tested in a general image compression application. Dictionaries are learned in the pixel domain and in the 9/7 wavelet domain, and then tested in a straightforward compression scheme. Results are compared with state-of-the-art compression methods. The proposed compression scheme using RLS-DLA learned dictionaries in the 9/7 wavelet domain performs better than using dictionaries learned by other methods. The compression rate is just below the JPEG-2000 rate which is promising considering the simple entropy coding used.

Index Terms— dictionary learning, RLS-DLA, sparse approximation, overcomplete dictionary, image compression

1. INTRODUCTION

Signal representation using sparse approximations and overcomplete dictionaries has been given considerable attention in recent years, especially in the area of compressed sensing, and can also be useful in other applications like texture classification, compression and denoising. We define a dictionary as a collection of finite atoms $\{d_k\}_{k=1}^{K}$, where $d_k \in \mathcal{R}^N$, with $K > N$ a redundant dictionary is implied. Letting the atoms be columns in a matrix $D \in \mathcal{R}^{N \times K}$ we can represent or approximate the signal vector as a sparse representation over the overcomplete (redundant) dictionary:

$$\tilde{x} = \sum_{k=1}^{K} w(k) d_k = Dw, \quad r = x - \tilde{x} = x - Dw, \qquad (1)$$

with a sparseness constraint on the coefficient, or weight, vector w, $\|w\|_0 \leq s$ or $\|w\|_1 \leq s_1$. The l_0 pseudo-norm $\|\cdot\|_0$ is the number of non-zero elements. Finding the optimal sparse coefficient vector can be formulated as

$$w_{opt} = \arg\min_{w} \|w\|_0 + \gamma \|x - Dw\|_2^2. \qquad (2)$$

This is what we call the vector selection problem and it can be addressed and approximated in different ways. Changing the l_0 norm with the l_1 norm is one option leading to basis pursuit and related algorithms, matching pursuit techniques are other options. We use order recursive matching pursuit (ORMP) in this work. If it is possible to find a good sparse solution to (1), the signal belongs to the sparseland model according to [1]. There are two main alternative ways to construct the dictionary D; a general chosen dictionary or an application specific learned dictionary. We are concerned with the latter approach.

It is well known that a sparse model fits natural images quite well, this is exploited in DCT and wavelet coding where many of the coefficients are quantized to zero. A sparse model also seems to fit the way the human visual system perceives an image [2]. Redundant dictionaries have successfully been used for sparse representation of images earlier [3], and also for image compression [4, 5, 6]. The contributions of this work are that dictionary learning is done by our new algorithm RLS-DLA [7] which improves the sparse representation capabilities and that dictionaries with overlapping atoms are effectively generated in the wavelet coefficient domain which removes blocking artifacts. The experiments presented show that each idea improves the compression rate, and used together they give good compression results both compared with DCT and wavelet based compression.

2. DICTIONARY LEARNING

Dictionary learning is the task of learning or training a dictionary such that it is well adapted to its purpose, i.e. sparse representation of a class of signals. When the sparse representation is used in an image compression scheme, entropy coding of the quantized coefficients and their positions should follow the sparse representation part. Thus the learned dictionary should give a good sparse approximation of images, and it should also be of moderate size, so that the position information is

acceptable.

A common setup for the dictionary learning problem [1] starts with access to a training set, $\{x_l\}_{l=1}^L,\ x_l \in \mathcal{R}^N$, and the aim to find both a dictionary, $D \in \mathcal{R}^{N\times K}$, and a corresponding coefficient set $\{w_l\}_{l=1}^L,\ w_l \in \mathcal{R}^K$ to satisfy (1). Let X denote a matrix with x_l as columns and W a matrix with the corresponding weights w_l as columns. The dictionary learning problem can be formulated as an optimization problem with respect to W and D:

$$\{D_{opt},\ W_{opt}\} = \arg\min_{D,W}\ \|W\|_0 + \gamma\|X - DW\|_F^2 \quad (3)$$

where $\|W\|_0 = \sum_{l=1}^L \|w_l\|_0$ is the total number of non-zero weights and $\|X - DW\|_F^2 = \sum_{l=1}^L \|r_l\|_2^2$. This is a very hard optimization problem, and a practical relaxation is to split the problem into two parts, iteratively solved: 1) Keeping D fixed, find W, and 2) Keeping W fixed, find D. This strategy is adopted in both the iterative least squares dictionary learning algorithms (ILS-DLA)[1] and partly in K-SVD[2] [5]. Part 1) is reduced to a vector selection problem (2). Part 2) has W fixed and the problem is reduced to the minimization of $\|X - DW\|_F^2$, so the ILS-DLA update is the least squares solution:

$$D = (XW^T)(WW^T)^{-1}. \quad (4)$$

This optimization strategy has two major drawbacks. First, the choice of an initial dictionary, D_0, is crucial as the method "converges", if it converges at all, towards a local minimum close to D_0, and there is no obvious good way to select D_0. Secondly, having few training vectors, L small, gives a risk of overtraining and having L large gives large execution times.

The RLS-DLA [7] addresses these problems by using a scheme which continuously updates the dictionary. In the derivation of RLS-DLA a 'time step' i is introduced and the matrices $X_i = [x_1, x_2, \ldots, x_i]$ of size $N \times i$, $W_i = [w_1, w_2, \ldots, w_i]$ of size $K \times i$ and $C_i = (W_i W_i^T)^{-1}$ are defined, as well as the dictionary D_i which is the least squares minimization of $\|X_i - DW_i\|_F^2$, i.e. $D_i = (X_i W_i^T)(W_i W_i^T)^{-1}$. At each step a new training vector x_i is supplied and the corresponding weights w_i are found using the previous (most recent) dictionary D_{i-1} and a vector selection algorithm. Using the matrix inversion lemma (Woodbury matrix identity) on C_i we get the following simple updating rules

$$C_i = C_{i-1} - \alpha u u^T, \quad (5)$$
$$D_i = D_{i-1} + \alpha r_i u^T, \quad (6)$$

$u = C_{i-1}w_i$ and $\alpha = 1/(1 + w_i^T u)$. $r_i = x_i - D_{i-1}w_i$ is the representation error.

[1]The family of MOD algorithms is denoted ILS-DLA in [8]

[2]here the non-zero positions in W remain fixed, but the coefficient values are changed

Introducing an adaptive *forgetting factor* $\lambda_i \leq 1$ in step i, as described in [7], makes the dictionary a lot less dependent on the initial dictionary as well as improving convergence properties. This scheme is denoted the Search-Then-Converge scheme. It will change the update in (5) to $C_i = (\lambda_i^{-1}C_{i-1}) - \alpha u u^T$ and $u = (\lambda_i^{-1}C_{i-1})w_i$, while equation (6) and α are unchanged.

For images it is (usually) no problem to get access to a lot of training vectors, and the initial dictionary is as important here as for any other application. This makes the RLS-DLA a reasonable choice for dictionary learning. For example, the images in Fig. 1 have 32768 non-overlapping 8×8 patches, and more than 2 million if the patches may overlap. Each patch is made into a training vector simply by lexicographically ordering of the pixels. The training vectors are picked in a random order from the set of training images and presented for the RLS-DLA algorithm. The K first random training vectors are used to make the initial dictionary, which soon will be forgotten, and to set the initial C_i matrix. Then, each new training vector x_i is processed in two parts. First the weights w_i are found by a Matching Pursuit algorithm (or any other vector selecting algorithm), using as stop criterion that the approximation error is below a limit ϵ. This limit can initially be calculated based on a target PSNR and it may be adjusted while learning progress. The second part updates the dictionary and the C_i matrix. The RLS-DLA equations (5) and (6) preferable including a forgetting factor λ_i, are applied. Once in a while we may also want to normalize the dictionary, i.e. scale each atom to get 2-norm equal to 1. This must be done as described in [7] and it also includes a scaling of the C_i matrix.

In [3] it was shown that a better sparse approximation is possible if the dictionary atoms are allowed to overlap with the neighboring patches, another benefit of overlapping is that blocking artifacts are avoided. This corresponds to how the wavelet basis functions overlap (JPEG-2000) while the DCT basis functions are block-oriented (JPEG). One easy way to design a dictionary with overlapping atoms is to generate the vectors in the coefficient domain of a wavelet. For example by transforming the image using a three level two dimensional 9/7 wavelet transform and by forming the vectors from the 8×8 patches in the coefficient domain. This is what we call the 9/7 wavelet domain or simply 9/7 in the rest of this work.

3. THE COMPRESSION SCHEME

A dictionary learned by the RLS-DLA scheme described in the previous section, or by another method, can be used in an image compression scheme with the following parts

1. Form the set of vectors from non-overlapping patches of the image, or from patches of the wavelet coefficients when the dictionary is learned in the wavelet domain. This set is denoted $\{x_j\}_{j=1}^{L}$ or X. As in other compression schemes (JPEG) the DC components, which themselves form a downsampled version of the original image, are quantized and compressed separately. A predictive method similar to DPCM, followed by Huffman coding of the prediction errors, is used here.

2. Find the sparse matrix W using sparse approximation of X with the dictionary D. We use ORMP for these L vector selection problems (2), with the representation error as stopping criterion, the limit ϵ is found from a given target PSNR.

3. The non-zero weights are quantized, introducing some more error and thus decreasing the PSNR. We use a uniform quantizer with thresholding; zero-bin is twice the size of the other bins. Based on some preliminary tests we found that setting the bin size to a value that gives a decrease of 0.4 to 0.5 dB in PSNR is appropriate.

4. The quantized W matrix is entropy coded. First the non-zeros entries of W (taken columnwise) are put into one sequence and the position information in another sequence, i.e. the number of zeros preceeding each non-zero entry is stored. Then these two sequences are Huffman coded, using the recursive splitting algorithm in [9], and formed into a bit sequence.

Note that the dictionary is not included in the compressed file, as it is supposed to be a general dictionary and an internal part of the compression scheme. In this work all learned dictionaries are intended to be general for the large class of grayscale natural images which here are represented by the training images in Fig. 1.

The advantage of step 2 above, compared with DCT or wavelet decomposition with the same error (PSNR), is that the representation is sparser. On the other hand, a disadvantage is that the coefficient matrix (including the zeros) is larger, and that the structure in the coefficients is lost or hidden, i.e. the non-zero positions in W are or seem to be almost random. JPEG, SPIHT and JPEG-2000 exploit the structure in the quantized coefficients in their advanced entropy coding schemes; especially the EBCOT scheme in JPEG-2000 is advanced and effective.

4. EXPERIMENTS AND RESULTS

To prove learned dictionaries and sparse representations to be interesting in compression we need to demonstrate that what we gain on getting *sparser representations* is not all lost during coding. We will show that comparing dictionary schemes with DCT and wavelet based schemes using similar entropy coding, but exploiting the coefficient structures in the DCT and the wavelet cases, is indeed in favor of the dictionary based schemes in the wavelet domain. 6 dictionaries were learned from the set of training images depicted in Fig. 1. This was done by the three methods MOD, K-SVD and RLS-DLA, both in the pixel domain and in the 9/7 wavelet domain. In all cases the dictionary size is set to 64×440, including the DC atom we get 441 atoms. The same size was used in [5]. For MOD and K-SVD a fixed training set is needed. We randomly pick 1500 patches from each of the training images, giving $L = 12000$. For RLS-DLA a new training vector is randomly selected in each iteration. In learning, as well as later in compression, ORMP is used for vector selection. The error limit ϵ is adjusted during learning to match a target PSNR equal to 38 dB for the training images. 1000 epochs, each processing $L = 12000$ training vectors, were done for MOD and K-SVD and 6 million iterations were done for RLS-DLA.

Fig. 1. The training set consists of 8 images, each have size 512×512 pixels, 256 gray levels.

Fig. 2. A detail from the test image `lena`. Original to the left, RLS-DLA (PSNR = 35.26) in the middle and SPITH (PSNR = 34.78) to the right. 0.25 bits per pixel. Some differences can be seen in the lower eyelash.

Compression results on two test images, `lena` and `boat`, are shown in Tab. 1 and 2. Probably, the `lena` image best fits the image class implicit given by the training images, the `boat` image has more distinct lines than most of the training images.

The proposed dictionary compression scheme is compared with DCT and wavelet based schemes using similar entropy coding. The results labeled "DCT" and "9/7 wavelet" in Tab. 1 and 2 were achieved using the same recursive Huffman coder [9], here the quantized coefficients

Table 1. Achieved PSNR on `lena`

bit rate	0.25	0.50	0.75	1.00	1.50
MOD (pix)	34.25	37.51	39.42	40.81	42.93
K-SVD (pix)	34.19	37.52	39.48	40.92	43.14
RLS (pix)	34.22	37.76	39.82	41.32	43.65
DCT	34.02	37.41	39.54	41.09	43.44
MOD (9/7)	35.12	38.21	39.98	41.28	43.34
K-SVD (9/7)	35.13	38.26	40.07	41.41	43.56
RLS (9/7)	35.26	38.57	40.42	41.78	44.02
9/7 wavelet	35.08	38.29	40.19	41.61	43.90
JPEG	31.64	35.85	37.77	39.14	41.16
SPIHT	34.78	38.11	40.12	41.40	43.26
JPEG-2000	35.29	38.60	40.48	41.91	44.13

Table 2. Achieved PSNR on `boat`.

bit rate	0.50	0.75	1.00	1.50	2.00
MOD (pix)	32.39	34.22	35.55	37.53	39.17
K-SVD (pix)	32.42	34.30	35.67	37.80	39.64
RLS (pix)	32.62	34.63	36.10	38.44	40.48
DCT	32.33	34.37	35.89	38.47	40.87
MOD (9/7)	32.84	34.60	35.88	37.83	39.50
K-SVD (9/7)	32.88	34.67	36.00	38.11	39.94
RLS (9/7)	33.11	35.03	36.42	38.73	40.75
9/7 wavelet	32.78	34.67	36.04	38.48	40.95
JPEG	30.90	32.99	34.46	36.44	38.05
SPIHT	32.63	34.68	35.96	38.35	40.42
JPEG-2000	33.32	35.27	36.74	39.20	42.01

were processed by both End-Of-Block coding and Run-Length coding to exploit the structure. Though simple, this scheme is quite effective. Matlab-files, more test images and results are presented on the web page[3]. Finally, JPEG, JPEG-2000 and SPIHT compression were done on the same images. The JPEG and JPEG-2000 implementation is the MATLAB `imwrite` command. For SPIHT (Set Partitioning in Hierarchical Trees) a public MATLAB implementation[4] was used.

Dictionaries in the wavelet domain achieves about 0.5 dB better PSNR than dictionaries in the pixel domain, slightly more evident at low bit rates. This is parallel to the improvement going from DCT to 9/7 wavelet in general. Going from JPEG to JPEG-2000 has in addition a significant contribution from the entropy coding. Comparing the dictionaries from the different dictionary learning algorithms the RLS-DLA dictionary performs best in the pixel domain as well as in the wavelet domain for both test images. Thus the best dictionary based result is from wavelet domain, RLS-DLA dictionary. These results are comparable but slightly worse than JPEG2000.

[3] `http://www.ux.uis.no/~karlsk/ICTools/ictools.html`

[4] `http://www.cipr.rpi.edu/research/SPIHT/spiht3.html`

5. CONCLUSION

The purpose of this work was to explore the compression capability of sparse approximations with dictionaries learned by RLS-DLA, both in the pixel domain and in the 9/7 wavelet domain. The experiments have demonstrated that the proposed compression scheme which uses learned dictionaries, preferable learned with RLS-DLA, performs quite well, just below the JPEG-2000 results, but better than "straight-forward" compression of the 9/7 wavelet coefficients which in turn is better than SPIHT. The ultimate goal is for the total scheme to perform better than state-of-the-art (JPEG2000), and future work includes improving the entropy coding part, for example by looking for exploitable structure in the coefficient position information.

6. REFERENCES

[1] Michael Elad, *Sparse and Redundant Representations, from Theory to Applications in Signal and Image Processing*, Springer, New York, USA, 2010.

[2] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed in V1," *Vision Research*, vol. 37, pp. 3311–3325, 1997.

[3] K. Skretting, K. Engan, J. H. Husøy, and S. O. Aase, "Sparse representation of images using overlapping frames," in *Proc. 12th Scandinavian Conference on Image Analysis, SCIA 2001*, Bergen, Norway, June 2001, pp. 613–620, available at `http://www.ux.uis.no/~karlsk/`.

[4] Joseph F. Murray and Kenneth Kreutz-Delgado, "Sparse image coding using learned overcomplete dictionaries," in *Proc. of the 14th IEEE Workshop on Machine Learning for Signal Processing*, Sao Luis, Brazil, Sept. 2004, pp. 579–588.

[5] M. Aharon, M. Elad, and A. 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.

[6] Ori Bryt and Michael Elad, "Compression of facial images using the K-SVD algorithm," *J. Vis. Comun. Image Represent.*, vol. 19, no. 4, pp. 270–282, 2008.

[7] K. Skretting and K. Engan, "Recursive least squares dictionary learning algorithm," *IEEE Transactions on Signal Processing*, vol. 58, pp. 2121–2130, Apr. 2010, Digital object identifier: 10.1109/TSP.2010.2040671.

[8] K. Engan, K. Skretting, and J. H. Husøy, "A family of iterative LS-based dictionary learning algorithms, ILS-DLA, for sparse signal representation," *Digital Signal Processing*, vol. 17, pp. 32–49, Jan. 2007.

[9] K. Skretting, J. H. Husøy, and S. O. Aase, "Improved Huffman coding using recursive splitting," in *NORSIG-99*, Asker, Norway, sep 1999, pp. 92–95, available at `http://www.ux.uis.no/~karlsk/`.