SHALLOW SPARSE AUTOENCODERS VERSUS SPARSE CODING ALGORITHMS FOR IMAGE COMPRESSION

Thierry Dumas, Aline Roumy, Christine Guillemot

INRIA Rennes Bretagne-Atlantique
thierry.dumas@inria.fr, aline.roumy@inria.fr, christine.guillemot@inria.fr

ABSTRACT

This paper considers the problem of image compression with shallow sparse autoencoders. We use both a T-sparse autoencoder (T-sparse AE) and a winner-take-all autoencoder (WTA AE). A performance analysis in terms of rate-distortion trade-off and complexity is conducted, comparing with LARS-Lasso, Coordinate Descent (CoD) and Orthogonal Matching Pursuit (OMP). We show that, WTA AE achieves the best rate-distortion trade-off, it is robust to quantization noise and it is less complex than LARS-Lasso, CoD and OMP.

***Index Terms*—** Image compression, sparse autoencoders, shallow architectures, complexity.

1. INTRODUCTION

The *sparse approximation* problem (P_0) corresponds to approximating a signal with the best linear combination of a small number of atoms from a redundant dictionary. At the present time, there exist two approaches to this non-linear approximation. Convex relaxation methods turn (P_0) into a convex program. Greedy methods do a series of locally optimal choices to provide an approximate solution of (P_0). The sparse coding algorithms dedicated to both approaches are somewhat complex. Over the last ten years, considerable effort has been devoted to making their implementation more efficient [1, 2].

Sparse autoencoders also provides a sparse decomposition of a signal but with a much less complex approach. More precisely, the coefficients of the sparse decomposition are computed by multiplying the signal with a fixed matrix and then only the most significant terms are kept. Therefore, sparse autoencoders are similar to a decomposition onto a DCT basis, but here, the matrix is learned from a set of signals. This paper sets up an image compression experiment that tests two shallow sparse autoencoders against three common sparse coding algorithms by evaluating both rate-distortion trade-off and complexity. We analyze in detail the sub-optimal form (shallow architecture) of sparse autoencoders to figure out whether general sparse autoencoders can hold promise for image compression. The rate cost is a key criterion. We therefore pay attention to sparse autoencoders able to create codes containing zero coefficients. For instance, the type of *sparse autoencoder* used for image denoising in [3, 4] is not relevant for image compression.

2. SPARSE CODING

This section lays the basic equations of the methods we will compare and analyze. Let $x \in \mathbb{R}^m$ represent an image patch.

2.1. Shallow sparse autoencoders

An autoencoder is a neural network that takes x as input and provides a reconstruction of x.

T-sparse autoencoder (T-sparse AE) [5]. For $T \in \mathbb{N}_+^*$, $f_T : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is the application that keeps the T largest coefficients in its input vector and sets the rest to 0. Given x, $V \in \mathbb{R}^{n \times m}$ and $b \in \mathbb{R}^n$, the first layer of T-sparse AE computes a sparse vector $z \in \mathbb{R}^n$, see (1). Then, given z, $\Phi \in \mathbb{R}^{m \times n}$ and $c \in \mathbb{R}^m$, the second layer of T-sparse AE gives a reconstruction $\tilde{x} \in \mathbb{R}^m$ of x , see (2).

$$z = f_T \left(Vx + b \right) \tag{1}$$

$$\tilde{x} = \Phi z + c \tag{2}$$

We also introduce a variant of T-sparse AE called T-absolute AE. It is identical to T-sparse AE except that its application $f_T : \mathbb{R}^n \rightarrow \mathbb{R}^n$ keeps the T coefficients in its input vector which absolute values are the largest and sets the rest to 0.

Winner-take-all autoencoder (WTA AE) [6]. p image patches $x^{(1)}, ..., x^{(p)}$ of dimension m are concatenated into a matrix $M_x \in \mathbb{R}^{m \times p}$. p sparse vectors $z^{(1)}, ..., z^{(p)}$ of dimension n are concatenated into a matrix $M_z \in \mathbb{R}^{n \times p}$. p copies of the vector $d \in \mathbb{R}^n$ are concatenated into a matrix $C_d \in \mathbb{R}^{n \times p}$. p copies of the vector $e \in \mathbb{R}^m$ are concatenated into a matrix $C_e \in \mathbb{R}^{m \times p}$. For $\alpha \in]0, 1[$, $g_\alpha : \mathbb{R}^{n \times p} \rightarrow \mathbb{R}^{n \times p}$ is the application that keeps the $\alpha \times n \times p$ largest coefficients in its input matrix and sets the rest to 0. Given M_x, $W \in \mathbb{R}^{n \times m}$ and d, the first layer of WTA AE computes M_z, see (3). Then, given M_z, $U \in \mathbb{R}^{m \times n}$ and e, the second layer of WTA AE calculates a reconstruction $M_{\tilde{x}} \in \mathbb{R}^{m \times p}$ of M_x, see (4).

$$M_z = g_\alpha \left(W M_x + C_d \right) \tag{3}$$

$$M_{\tilde{x}} = U M_z + C_e \tag{4}$$

2.2. Sparse coding algorithms

LARS-Lasso and Coordinate Descent (CoD) [7]. Given x, $\lambda \in \mathbb{R}_+^*$ and a dictionary $D \in \mathbb{R}^{m \times n}$, they both attempt to solve a relaxed version of (P_0) written in (5) to find a sparse decomposition $z \in \mathbb{R}^n$ of x over D.

$$z = \min_{z_v} \left(\|x - Dz_v\|_2^2 + \lambda \|z_v\|_1 \right) \tag{5}$$

$$\tilde{x} = Dz \tag{6}$$

Orthogonal Matching Pursuit (OMP) [8]. OMP is the canonical greedy algorithm for (P_0). Given x, $K \in \mathbb{N}_+^*$ and a dictionary $D \in \mathbb{R}^{m \times n}$, its goal is to find a vector of coefficients $z \in \mathbb{R}^n$ with at most K non-zero terms so that Dz equals to x approximatively.

$$z = \min_{z_v} \|x - Dz_v\|_2^2 \text{ st. } \|z_v\|_0 \leq K \tag{7}$$

$$\tilde{x} = Dz \tag{8}$$

(1), (3), (5) and (7) illustrate that, unlike T-sparse AE and WTA AE, LARS-Lasso, CoD and OMP go through optimization for encoding.

3. LEARNING FRAMEWORK

Before running any test, T-sparse AE and WTA AE must train their parameters. Similarly, LARS-Lasso, CoD and OMP need pre-trained dictionaries.

3.1. Training data extraction

The CALTECH-256 dataset [9] is a set of 256 objects categories containing 30607 images. The first 200 categories are dedicated to training which makes a total of 22823 images for training. The RGB color space is transformed into the YCbCr color space and we only keep the luminance channel. $N \in \mathbb{N}_+^*$ patches of size $\sqrt{m} \times \sqrt{m}$ are randomly extracted from the 22823 luminance images. We remove the DC component from each patch. It yields a bank of patches $S = \left\{x^{(1)}, x^{(2)}, ..., x^{(N)}\right\}$ where, for $i \in [|1, N|]$, $x^{(i)} \in \mathbb{R}^m$. We define μ_j and σ_j as respectively the mean and the standard deviation of the j^{th} pixel x_j over all the patches in S. For $m \leq 144$ and $N \geq 600000$, we have noticed that, $\forall j \in [|1, m|]$, $\mu_j \approx 0$ and $\sigma_j \approx \sigma \in \mathbb{R}_+^*$. This remark must be kept aside as it will hold importance in section 3.2.

3.2. Shallow sparse autoencoders training

Training T-sparse AE. Given S and $T \in \mathbb{N}_+^*$, V, b, Φ and c are optimized to solve (9).

$$\begin{aligned} &\min_{V,b,\Phi,c} \frac{1}{N} \sum_{i=1}^{N} \left\| x^{(i)} - \tilde{x}^{(i)} \right\|_2^2 \\ &\text{st. } \forall i \in [|1, N|], \left\| f_T \left(V x^{(i)} + b \right) \right\|_0 \leq T \end{aligned} \tag{9}$$

Training WTA AE. Given S, $\alpha \in]0, 1[$ and $p \in \mathbb{N}_+^*$, W, d, U and e are optimized to solve (10).

$$\begin{aligned} &\min_{W,d,U,e} \frac{1}{N} \sum_{i=1}^{N_p} \left\| M_x^{(i)} - M_{\tilde{x}}^{(i)} \right\|_F^2 \\ &\text{st. } \forall i \in [|1, N_p|], \left\| g_\alpha \left(W M_x^{(i)} + C_d \right) \right\|_0 \leq \alpha \times n \times p \\ &N_p = \frac{N}{p} \end{aligned} \tag{10}$$

$\|.\|_F$ stands for the Frobenius norm and $\|.\|_0$ counts the number of non-zero elements in its input matrix. p must be a divisor of N.

For T-sparse AE and WTA AE, the solver is mini-batch gradient descent with momentum [10], where all gradients are computed by backpropagation. [11] gives many tricks to run backpropagation. In particular, we must be careful that the mean of each input to a neural network over all training examples be zero and all inputs have the same standard deviation. As explained in section 3.1, S satisfies this requirement.

3.3. Dictionary learning

We use three different dictionary learning methods tailored to respectively LARS-Lasso, CoD and OMP.

Learning D_λ for LARS-Lasso. Given S and $\lambda \in \mathbb{R}_+^*$, the algorithm in [1] creates D_λ [1].

Learning D_δ for CoD. Given S and $\delta \in \mathbb{R}_+^*$, algorithm 1 solves (11).

$$\begin{aligned} &\min_{D_\delta, z^{(1)}, ..., z^{(N)}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\left(x^{(i)}, z^{(i)}, D_\delta, \delta\right) \\ &\mathcal{L}\left(x^{(i)}, z^{(i)}, D_\delta, \delta\right) = \left\| x^{(i)} - D_\delta z^{(i)} \right\|_2^2 + \delta \left\| z^{(i)} \right\|_1 \end{aligned} \tag{11}$$

Algorithm 1 alternates between sparse coding steps that involve CoD and dictionary updates that use stochastic gradient descent. For $j \in [|1, n|]$, $D_{\delta,j}$ is the j^{th} column of D_δ.

Algorithm 1

```
Inputs: $S$, $\delta$ and $\varepsilon \in \mathbb{R}_+^*$.
$D_\delta$ is randomly initialized.
For several epochs do:
    $\forall i \in [|1, N|]$, $z^{(i)} = \min_{z_v} \mathcal{L}\left(x^{(i)}, z_v, D_\delta, \delta\right)$ using CoD.
    $D_\delta \leftarrow D_\delta - \varepsilon \frac{\partial \mathcal{L}\left(x^{(i)}, z^{(i)}, D_\delta, \delta\right)}{\partial D_\delta}$
    $\forall j \in [|1, n|], D_{\delta,j} \leftarrow \frac{D_{\delta,j}}{\|D_{\delta,j}\|_2}$
Output: $D_\delta$.
```

[1] Its implementation is the one from the SPAMS library: http://spams-devel.gforge.inria.fr/.

Learning D_o for OMP. K-SVD [12] is the commonly used dictionary learning algorithm for OMP. But K-SVD is too slow for large N. Unlike K-SVD, algorithm 2 can quickly generate D_o for large N. In section 5.4, we will compare K-SVD with algorithm 2. Given S and $K \in \mathbb{N}_{+}^{*}$, algorithm 2 solves (12). Note the similarities between (9) and (12).

$$\begin{aligned} &\min_{D_o, z^{(1)}, ..., z^{(N)}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{G}\left(x^{(i)}, z^{(i)}, D_o\right) \\ &\text{st. } \forall i \in [|1, N|], \left\|z^{(i)}\right\|_0 \leq K \\ &\mathcal{G}\left(x^{(i)}, z^{(i)}, D_o\right) = \left\|x^{(i)} - D_o z^{(i)}\right\|_2^2 \end{aligned} \tag{12}$$

Algorithm 2 alternates between sparse coding steps that involve OMP and dictionary updates that use stochastic gradient descent. For $j \in [|1, n|]$, $D_{o,j}$ is the j^{th} column of D_o. Algorithm 2 and K-SVD mainly differ in the scheduling of dictionary updates. At epoch t, for a given training patch of S, algorithm 2 updates all the atoms of D_o used in the decomposition of that patch over D_o. At epoch t, K-SVD updates only one time each atom of the dictionary; each atom update takes into account the approximation error of all the training patches of S using that atom in their decomposition.

Algorithm 2

```
Inputs: $S$, $K$ and $\varepsilon \in \mathbb{R}_{+}^{*}$.
$D_o$ is randomly initialized.
For several epochs do:
    $\forall i \in [|1, N|],\ z^{(i)} = \min_{z_v} \mathcal{G}\left(x^{(i)}, z_v, D_o\right) \text{ st. } \|z_v\|_0 \leq K$ : OMP
    $D_o \leftarrow D_o - \varepsilon \frac{\partial \mathcal{G}\left(x^{(i)}, z^{(i)}, D_o\right)}{\partial D_o}$
    $\forall j \in [|1, n|],\ D_{o,j} \leftarrow \frac{D_{o,j}}{\|D_{o,j}\|_2}$
Output: $D_o$.
```

4. IMAGE COMPRESSION SCHEME

After training in section 3, each competitor undergoes the same image compression experiment described below.

4.1. Testing data extraction

In the CALTECH-256 dataset, the 250th category gathers 96 images of zebras. We pick them out of the dataset, convert the RGB color space into the YCbCr color space and only keep the luminance channel. It yields the bank of luminance images denoted $\Gamma = \left\{X^{(1)}, ..., X^{(96)}\right\}$. We choose the images of zebras for testing because they have disparate backgrounds. This makes them difficult to reconstruct. Note that the training set S does not contain any patches of zebras.

4.2. Performance measures

For each $i \in [|1, 96|]$, $X^{(i)}$ passes through the system displayed in figure 1. Note that the 96 images in Γ have different sizes. For $i \in [|1, 96|]$, $N^{(i)}$ varies. The quantizer block applies a scalar uniform quantization on the values of the non-zero coefficients in the set $\{z^{(1)}, ..., z^{(N^{(i)})}\}$. The "coding" part draws on the CODEC in [13]. The set of indexes $\{a_k^{(j)}\}_{k \in [|1, \|z^{(j)}\|_0|]}^{j \in [|1, N^{(i)}|]}$ is encoded using a fixed length code. The set of quantized values $\{\tilde{\gamma}_k^{(j)}\}_{k \in [|1, \|z^{(j)}\|_0|]}^{j \in [|1, N^{(i)}|]}$ is entropy coded with an Huffman code. We call PSNR (in dB) the average PSNR over the 96 images in Γ. The mean rate over the 96 images in Γ, denoted R (in bpp), is defined as:

$$\begin{aligned} R &= \frac{1}{96} \sum_{i=1}^{96} R^{(i)} \\ R^{(i)} &= \left(E^{(i)} + \log_2(n)\right) \frac{1}{m \times N^{(i)}} \sum_{j=1}^{N^{(i)}} \|z^{(j)}\|_0 \end{aligned}$$

For $i \in [|1, 96|]$, $E^{(i)}$ refers to the entropy of the distribution of the set of quantized values $\{\tilde{\gamma}_k^{(j)}\}_{k \in [|1, \|z^{(j)}\|_0|]}^{j \in [|1, N^{(i)}|]}$. Note that R does not take into account the encoding cost of the DC components, as this has to be encoded for each evaluated method. R should also include the encoding cost of the number of non-zero coefficients. Note that, for T-sparse AE and OMP, $\forall j \in [|1, N^{(i)}|]$, $\|z^{(j)}\|_0 = \|z\|_0$ independent of both i and j. For WTA AE, LARS-Lasso and CoD, for each $j \in [|1, N^{(i)}|]$, $\|z^{(j)}\|_0$ varies. However, we neglect this contribution as its cost is small with respect to the cost of encoding the set of indexes $\{a_k^{(j)}\}_{k \in [|1, \|z^{(j)}\|_0|]}^{j \in [|1, N^{(i)}|]}$ and the set of quantized values $\{\tilde{\gamma}_k^{(j)}\}_{k \in [|1, \|z^{(j)}\|_0|]}^{j \in [|1, N^{(i)}|]}$.

Moreover, for OMP, R can be further optimized. Indeed, the coefficients associated to the first iterations of OMP have larger absolute values than the coefficients associated to the last iterations of OMP. Therefore, an entropy coding can be performed per iteration of OMP. For $k \in [|1, \|z\|_0|]$, let $E_k^{(i)}$ be the entropy of the distribution of the set of quantized values $\{\tilde{\gamma}_k^{(j)}\}^{j \in [|1, N^{(i)}|]}$. For OMP only, R becomes R_o.

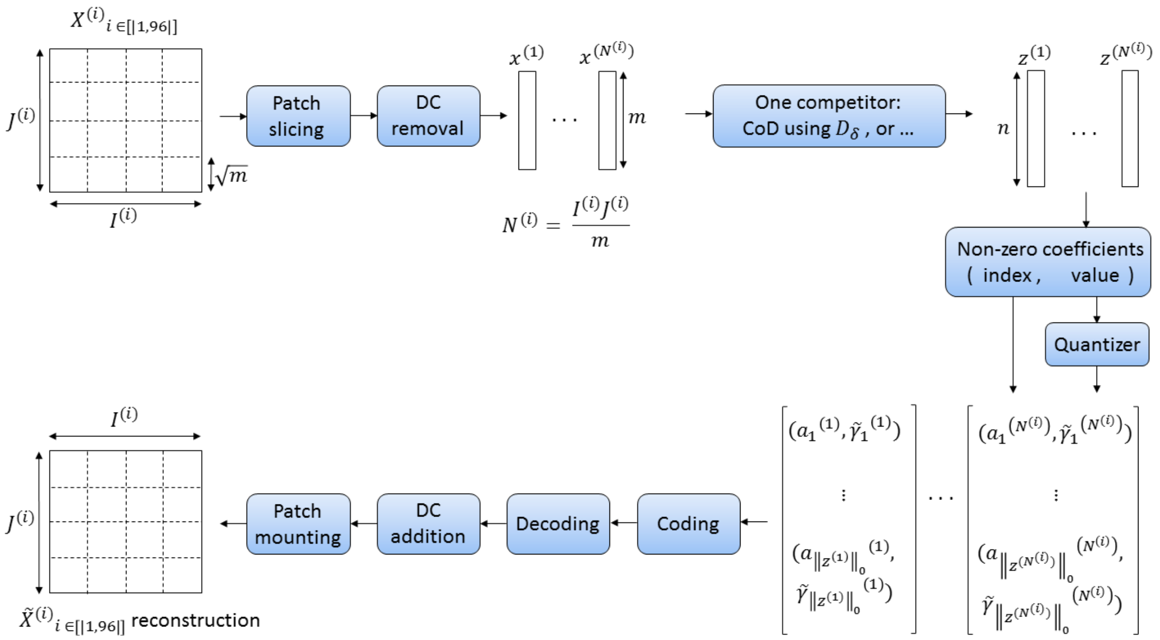
$$R_o = \frac{1}{96 \times m} \sum_{i=1}^{96} \sum_{k=1}^{\|z\|_0} \left(E_k^{(i)} + \log_2(n)\right)$$

5. ANALYSIS

5.1. Curves creation

For each value $n \in \mathbb{N}_{+}^{*}$, we start with the training in section 3. $N = 600000$, $m = 144$, $T = K = 10$, $p = 20000$, $\alpha = \frac{10}{n}$ and $\lambda = \frac{\delta}{2} = 0.1$. The relationship between λ and δ comes from the fact that the minimization problem in [1] differs from (11) by a factor 0.5. We have observed that these values for T, K, p, α, λ and δ bring to competitors the best models for section 4.

Fig. 1: Image compression scheme on Γ.



$X^{(i)}_{i\in[|1,96|]}$
$J^{(i)}$
$I^{(i)}$
$\sqrt{m}$
Patch slicing
DC removal
$x^{(1)}$
$x^{(N^{(i)})}$
m
$N^{(i)} = \frac{I^{(i)}J^{(i)}}{m}$
One competitor: CoD using D_δ , or ...
$z^{(1)}$
$z^{(N^{(i)})}$
n
Non-zero coefficients (index , value)
Quantizer
$(a_1{}^{(1)}, \tilde{\gamma}_1{}^{(1)})$
$(a_{\|z^{(1)}\|_0}{}^{(1)}, \tilde{\gamma}_{\|z^{(1)}\|_0}{}^{(1)})$
$(a_1{}^{(N^{(i)})}, \tilde{\gamma}_1{}^{(N^{(i)})})$
$(a_{\|z^{(N^{(i)})}\|_0}{}^{(N^{(i)})}, \tilde{\gamma}_{\|z^{(N^{(i)})}\|_0}{}^{(N^{(i)})})$
Coding
Decoding
DC addition
Patch mounting
$\tilde{X}^{(i)}_{i\in[|1,96|]}$ reconstruction

At test time, several values for the sparsity parameters of T-sparse AE, WTA AE, LARS-Lasso, CoD and OMP are used to draw figure 2. Note that, at test time, for WTA AE, for $i \in [|1, 96|]$, $p = N^{(i)}$ varies.

5.2. Analysis for 8-bits uniform quantization

The experiments in figure 2 are conducted when LARS-Lasso uses D_λ, CoD uses D_δ and OMP uses D_o. We also ran other experiments where LARS-Lasso, CoD and OMP shared a common dictionary. We noticed insignificant differences from what is displayed in figure 2. The domination of OMP over LARS-Lasso and CoD has nothing to do with pre-trained dictionaries. It must be related to the nature of OMP. Note that this observation coincides with the conclusions of Adam Coates and al. [14]. Their framework differs from ours as it is classification. They also suggest that the quality of pre-trained dictionaries has little impact as long as the dictionaries express relatively well the diversity of image patches. However, the sparse coding mechanism matters a lot.

For OMP, we previously compared R with its optimized variant R_o used in figure 2. Over the range of PSNRs in figure 2, $R - R_o$ typically belonged to $[0.01, 0.05]$ dB. It was not large. The good rate-distortion trade-offs of OMP compared to those of LARS-Lasso and CoD cannot be attributed to the fact that the measure of rate for OMP has been optimized.

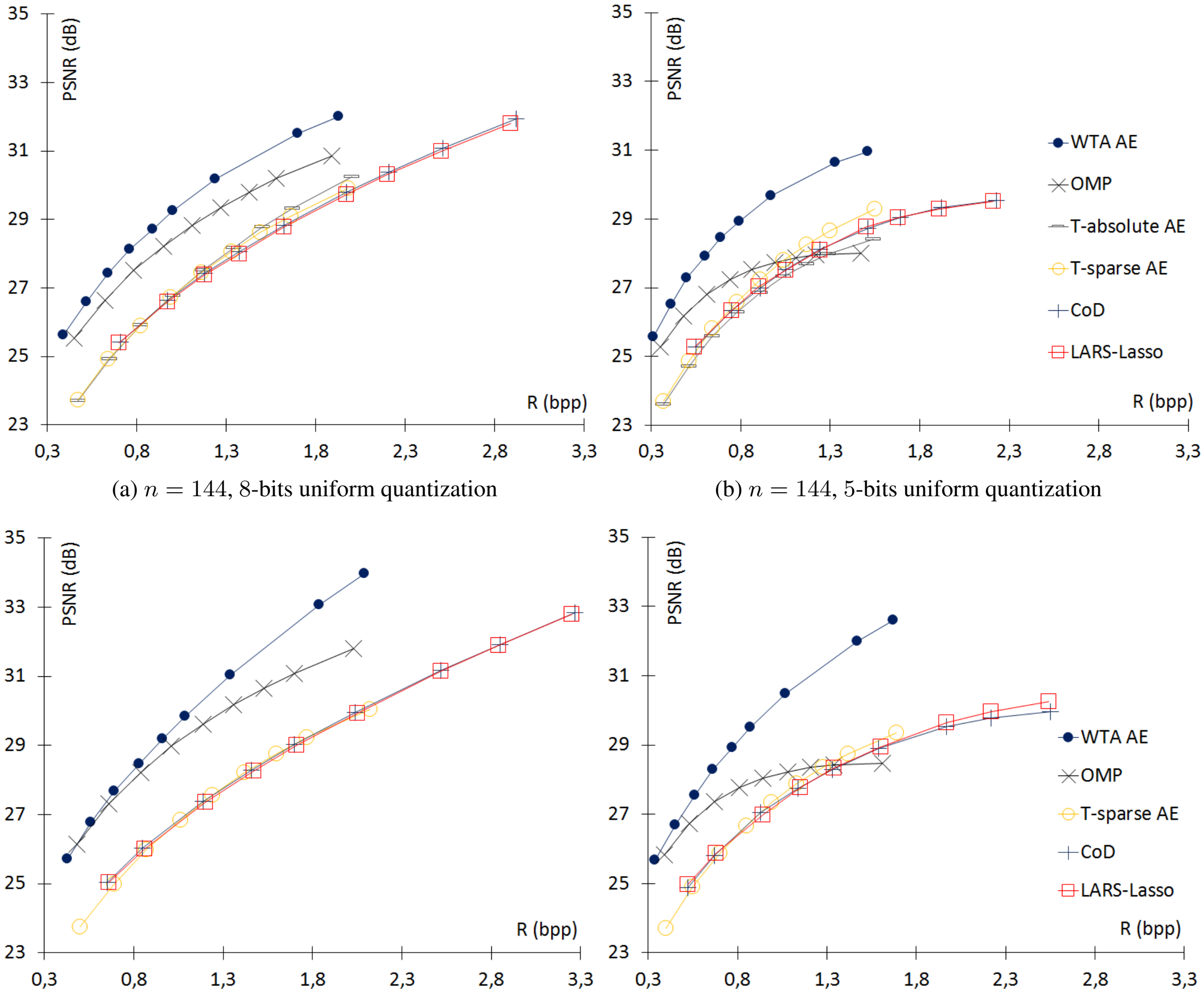
We observe that, for equivalent rates, the PSNRs provided by T-sparse AE are below those of OMP. We try to put forward an explanation. The main difference between autoencoders and sparse coding algorithms lies in the relation linking an image patch $x \in \mathbb{R}^m$ to its sparse vector $z \in \mathbb{R}^n$. For autoencoders, this relation simply couples a linear combination and a point-wise non-linear function, see (1). However, for sparse coding algorithms, it is an iterative process. Given $x \in \mathbb{R}^m$, autoencoders put more restrictions on the form z can take. This limited diversity of sparse vectors leads to poorer reconstruction of image patches.

WTA AE beats all its competitors in terms of rate-distortion trade-off. This can be explained by the way the sparsity constraint is handled. For OMP and T-sparse AE, the number of non-zero coefficients in the sparse decomposition is fixed per image patch, see (1) and (7). Instead, for WTA AE, it is fixed per set of patches, see (3). This allows to spread the ressources over the patches, using less non-zero coefficients for patches with less complex texture. This WTA AE approach can be viewed as a suboptimal form of the “block sparsity selection using a global rate-distortion criterion” for OMP [13], where the coefficients leading to the largest distortion decrease are kept. Yet, the approach in [13] is highly complex.

5.3. Analysis for variable uniform quantization

Unlike OMP, T-sparse AE and WTA AE seem to be rarely affected by the quantization level. For $i \in [|1, 96|]$, the non-zero coefficients in the set $\{z^{(1)}, ..., z^{(N^{(i)})}\}$ created by OMP are either positive or negative and most of them gather around zero. However, T-sparse AE produces a set $\{z^{(1)}, ..., z^{(N^{(i)})}\}$ whose non-zero coefficients are positive and are spreaded over a wide range. We assume that the impact of quantization

Fig. 2: Evolution of PNSR with R.



(a) $n = 144$, 8-bits uniform quantization

(b) $n = 144$, 5-bits uniform quantization

(c) $n = 288$, 8-bits uniform quantization

(d) $n = 288$, 5-bits uniform quantization

arises from this difference in the distribution of the non-zero coefficients in the set $\{z^{(1)}, ..., z^{(N^{(i)})}\}$. To verify this, we look at the behavior of T-absolute AE. Basically, the only distinction between T-sparse AE and T-absolute AE comes from the fact that T-absolute AE generates a set $\{z^{(1)}, ..., z^{(N^{(i)})}\}$ that contains both positive and negative non-zero coefficients. Like OMP, the 5-bits uniform quantization harms the PSNRs in the T-absolute AE curve. For OMP and T-absolute AE, the quantization operation might cause many confusions between the small positive and negative non-zero coefficients in the set $\{z^{(1)}, ..., z^{(N^{(i)})}\}$. The reconstruction of image patches is damaged by these confusions.

5.4. Proof of the equivalence K-SVD/algorithm 2

We restart the entire scheme in sections 3 and 4. $N = 50000$ and $n = 288$. This way, K-SVD quickly trains a dictionary. Figure 3 compares two cases: OMP uses the dictionary from K-SVD and OMP uses the dictionary from algorithm 2.

5.5. Complexity analysis

Table 1 summarizes the complexity per image patch for OMP based on a fast QR-1 decomposition [2], T-sparse AE and WTA AE. The basic implementation of LARS-Lasso [15] has complexity $\mathcal{O}(n^3)$, assuming that $m \leq n$. [1] suggests an efficient Cholesky-based implementation of LARS-Lasso and processes batches of $\eta \in \mathbb{N}_+^*$ image patches. Using these optimizations, LARS-Lasso becomes only slightly slower than OMP. One iteration of CoD has complexity $\mathcal{O}(n)$ [7] and the number of iterations is related to a threshold. CoD can be either faster than OMP or much slower depending on this threshold so we do not show the complexity of CoD. Note that, in our experiments, CoD is much slower than OMP as

Fig. 3: Evolution of PSNR with R, $n = 288$, 8-bits uniform quantization.

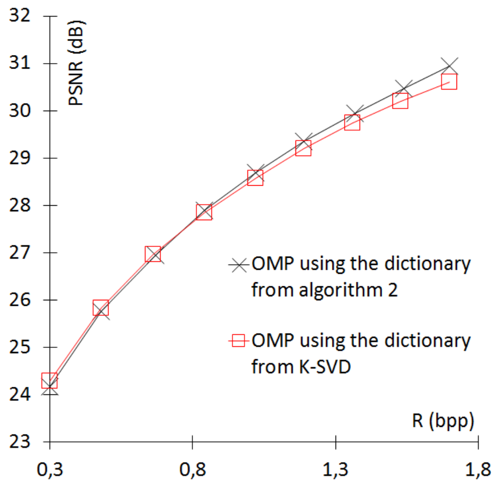


Table 1: complexity per image patch assuming that $\|z\|_0 \ll m \leq n$.

Method	Complexity
OMP	$\mathcal{O}\left(mn\,\|z\|_0\right)$ [2]
T-sparse AE	$\mathcal{O}\left(mn\right)$
WTA AE	$\mathcal{O}\left(mn\right)$

we set a low threshold to get the best PSNRs from CoD.

We clarify the complexity of WTA AE displayed in 1. In (3), given a set of images patches $M_x \in \mathbb{R}^{m \times p}$ and $W \in \mathbb{R}^{n \times m}$, the complexity of the matrix product WM_x is $\mathcal{O}(nmp)$. In (3), g_α requires a sorting operation of complexity $\mathcal{O}(np\log{(np)})$. At test time, $200 \leq p \leq 8000$ so $\mathcal{O}(nmp)$ dominates $\mathcal{O}(np\log{(np)})$. The complexity per image patch of WTA AE is $\mathcal{O}(nm)$.

6. CONCLUSION

We show that, for image compression, WTA AE provides the best rate-distortion trade-off, it is robust to quantization noise and less complex than LARS-Lasso, CoD and OMP. Interestingly, these first results show that the quadratic complexity of shallow neural networks together with a specific training can outperform the classical sparse coding algorithms for image compression. They are potential avenues to do better image compression at low complexity.

7. REFERENCES

[1] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro, "Online learning for matrix factorization and sparse coding," *JMLR*, vol. 11(1), pp. 19–60, 2010.

[2] Bob L. Sturm and Mads G. Christensen, "Comparaison of orthogonal matching pursuit implementations," *Proc. 20th IEEE EUSIPCO*, pp. 220–224, 2012.

[3] Junyuan Xie, Linli Xu, and Enhong Chen, "Image denoising and inpainting with deep neural networks," in *NIPS*, 2012.

[4] Forest Agostinelli, Michael R. Anderson, and Honglak Lee, "Adaptive multi-column deep neural networks with application to robust image denoising," in *NIPS*, 2013.

[5] Alireza Makhzani and Brendan Frey, "k-sparse autoencoders," in *ICLR*, 2014.

[6] Alireza Makhzani and Brendan Frey, "A winner-take-all method for training sparse convolutional autoencoders," in *NIPS*, 2015.

[7] YingYing Li and Stanley Osher, "Coordinate descent optimization for l1 minimization with application to compressed sensing; a greedy algorithm," *Inverse Problem and Imaging*, vol. 3(3), pp. 487–503, 2009.

[8] Joel A. Tropp, "Greed is good: algorithmic results for sparse approximation," *IEEE Trans Inf. Theory*, vol. 50, no. 10, pp. 2231–2242, 2004.

[9] Greg Griffin, Alex Holub, and Pietro Perona, "Caltech-256 object category dataset," Tech. Rep., California Institute of Technology, 2007.

[10] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton, "On the importance of initialization and momentum in deep learning," in *Proceedings of the 30th International Conference on Machine Learning*, pp. 1139-1147, 2013.

[11] Yann LeCun, Leon Bottou, Genevieve B. Orr, and Klaus-Robert Muller, "Efficient backprop," *Neural Networks: Tricks of the Trade*, pp. 9–50, Springer, 1998.

[12] Ori Bryt and Michael Elad, "Compression of facial images using the k-svd algorithm," *J. Visual Common. Image Representation*, vol. 19, no. 4, pp. 270–283, 2008.

[13] Joaquin Zepeda, Christine Guillemot, and Ewa Kijak, "Image compression using sparse representations and the iteration-tuned and aligned dictionary," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, September 2011.

[14] Adam Coates and Andrew Y. Ng, "The importance of encoding versus training with sparse coding and vector quantization," in *ICML*, 2011.

[15] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani, "Least angle regression," *The Annals of Statistics*, vol. 32, no. 2, pp. 407–499, 2004.