Image Compression with Stochastic Winner-Take-All Auto-Encoder

Thierry Dumas, Aline Roumy, Christine Guillemot

IMAGE COMPRESSION WITH STOCHASTIC WINNER-TAKE-ALL AUTO-ENCODER

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 addresses the problem of image compression using sparse representations. We propose a variant of auto-encoder called Stochastic Winner-Take-All Auto-Encoder (SWTA AE). "Winner-Take-All" means that image patches compete with one another when computing their sparse representation and "Stochastic" indicates that a stochastic hyperparameter rules this competition during training. Unlike auto-encoders, SWTA AE performs variable rate image compression for images of any size after a single training, which is fundamental for compression. For comparison, we also propose a variant of Orthogonal Matching Pursuit (OMP) called Winner-Take-All Orthogonal Matching Pursuit (WTA OMP). In terms of rate-distortion trade-off, SWTA AE outperforms auto-encoders but it is worse than WTA OMP. Besides, SWTA AE can compete with JPEG in terms of rate-distortion.

***Index Terms*—** Image compression, sparse representations, auto-encoders, Orthogonal Matching Pursuit.

1. INTRODUCTION

Auto-encoders are powerful tools for reducing the dimensionality of data. Deep fully-connected auto-encoders [1] are traditionally used for this task. However, two issues have so far prevented them from becoming efficient image compression algorithms: they can only be trained for one image size and one compression rate [2, 3].

[4] attempts to solve both issues. The authors train an auto-encoder on image patches so that images of various sizes can be compressed. Their auto-encoder is a recurrent [5] residual auto-encoder that performs variable rate image compression after a single training. But all image patches have the same rate and therefore different distortions due to the texture complexity variety in image patches. In addition, recurrence, which is equivalent to scalability in image compression, is not optimal in terms of rate-distortion trade-off [6, 7].

Instead, we propose to perform learning on whole images under a global rate-distortion constraint. This is done through Winner-Take-All (WTA), which can be viewed as a competition between image patches when computing their representation. Furthermore, auto-encoders architecture must adapt to different rates. Therefore, during training, the WTA parameter that controls the rate is stochastically driven. These contributions give rise to Stochastic Winner-Take-All Auto-Encoder (SWTA AE).

1.1. Notation

Vectors are denoted by bold lower case letters and matrices by upper case ones. $\mathbf{X}_j$ denotes the j^{th} column of a matrix $\mathbf{X}$. $\|\mathbf{X}\|_F$ is the Frobenius norm of $\mathbf{X}$. $\|\mathbf{X}\|_0$ counts the number of non-zero elements in $\mathbf{X}$. The support of a vector $\mathbf{x}$ is $\text{supp}\,(\mathbf{x}) = \{i \mid \mathbf{x}_i \neq 0\}$.

2. STOCHASTIC WINNER-TAKE-ALL AUTO-ENCODER (SWTA AE)

We now present our Stochastic Winner-Take-All Auto-Encoder (SWTA AE) whose architecture is shown in Figure 1. SWTA AE is a type of auto-encoder. An auto-encoder is a neural network that takes an input and provides a reconstruction of this input. We justify below two of the most critical choices for the SWTA AE architecture.

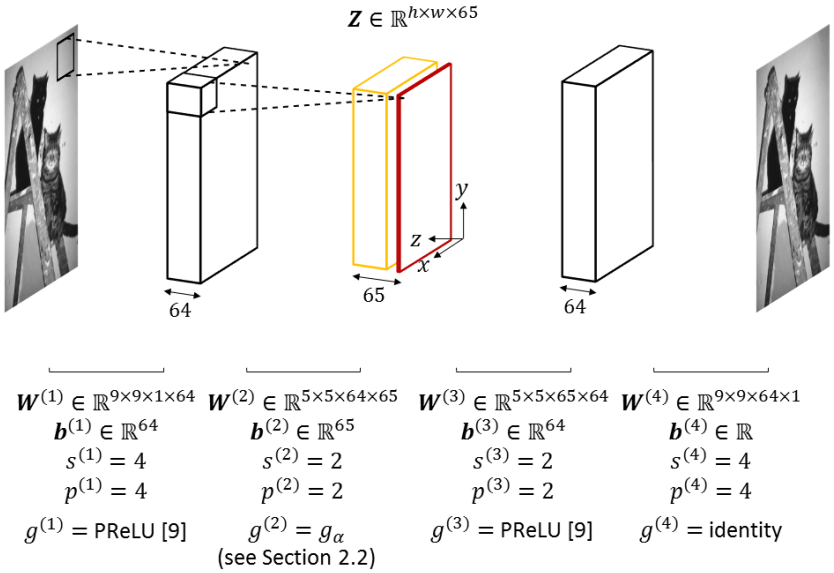
2.1. Strided convolution

A compression algorithm must process images of various sizes. However, the most efficient neural networks [8, 9] require that all images have the same size. Indeed, they include both convolutional layers and fully-connected layers, and the number of parameters of the latters directly depends on the image size. This imposes to train one architecture per image size. That is why our proposed SWTA AE only contains convolutional layers. Its encoder has two convolutional layers and its decoder has two deconvolutional layers [10]. Each layer $i \in [\![1, 4]\!]$ consists in convolving the layer input with the bank of filters $\mathbf{W}^{(i)}$, adding the biases $\mathbf{b}^{(i)}$ and applying a mapping $g^{(i)}$, producing the layer output. For the borders of the layer input, zero-padding of width $p^{(i)}$ is used.

Max-pooling is a core component of neural networks [11] that downsamples its input representation by appling a max

This work has been supported by the French Defense Procurement Agency (DGA).

Fig. 1: SWTA AE architecture.



filter to non-overlapping sub-regions. But max-pooling increases the rate. Indeed, if the encoder contains a max-pooling layer, the locations of maximum activations selected during pooling operations must be recorded and transmitted to the corresponding unpooling layer in the decoder [12, 13]. Instead, for $i \in [\![1, 2]\!]$, we downsample using a fixed stride $s^{(i)} > 1$ for convolution, which does not need any signaling.

2.2. Semi-sparse bottleneck

The bottleneck is the stack of feature maps denoted $\mathbf{Z} \in \mathbb{R}^{h\times w\times 65}$ in Figure 1. $\mathbf{Z}$ is the representation of the input image that is processed in Section 2.3 to give the bitstream.

We propose to apply a global sparse constraint that provides control over the coding cost of $\mathbf{Z}$. This is called Winner-Take-All (WTA). Let us define WTA via a mapping $g_\alpha : \mathbb{R}^{h\times w\times 64} \to \mathbb{R}^{h\times w\times 64}$, where $\alpha \in]0, 1[$ is the WTA parameter. g_α keeps the $\alpha \times h \times w \times 64$ most representative coefficients in its input tensor, i.e. those whose absolute values are the largest, and sets the rest to 0. g_α only applies to the output of the convolution in the second layer involving the first 64 filters in $\mathbf{W}^{(2)}$, producing the first 64 sparse feature maps in $\mathbf{Z}$. Figure 1 displays these sparse feature maps in orange. Varying α leads to various coding costs of $\mathbf{Z}$. Note that [14] uses WTA, but our WTA rule is different and g_α does not apply to specific dimensions of its input tensor as this constraint is not relevant for image compression.

A patch of the input image might be represented by a portion of the first 64 sparse feature maps in $\mathbf{Z}$ that only contains zeros. We want to ensure that each image patch has a minimum code in $\mathbf{Z}$ to guarantee a sufficient quality of reconstruction per patch. That is why the last feature map in $\mathbf{Z}$ is not sparse. Figure 1 displays it in red. We have noticed that, during the training in Section 4.2, SWTA AE learns to store in the last feature map a subsampled version of its input image.

2.3. Bitstream generation

The coefficients of the non-sparse feature map in $\mathbf{Z}$ are uniformly quantized over 8-bits and coded with a Huffman code. The non-zero coefficients of the 64 sparse feature maps in $\mathbf{Z}$ are uniformly quantized over 8-bits and coded with a Huffman code while their position is coded as explained hereafter. Figure 1 defines a coordinate system (x, y, z) for $\mathbf{Z}$. The non-zero coefficients in $\mathbf{Z}$ are scanned along (x, y, z) where z changes the fastest. The position along z is coded with a fixed-length code and, for each pair (x, y), the number of non-zero coefficients along z is coded with a Huffman code. This unequivocally characterizes the position of each non-zero coefficient in $\mathbf{Z}$. We have observed that this processing is effective in encoding the position of the non-zero coefficients.

3. WINNER-TAKE-ALL ORTHOGONAL MATCHING PURSUIT (WTA OMP)

SWTA AE is similar to Orthogonal Matching Pursuit (OMP) [15], a common algorithm for image compression using sparse representations [16]. The difference is that SWTA AE computes the sparse representation of an image by alternating convolutions and mappings whereas OMP runs an iterative decomposition of the image patches over a dictionary. More precisely, let $\mathbf{x} \in \mathbb{R}^m$ be an image patch. Given $\mathbf{x}$ and a dictionary $\mathbf{D} \in \mathbb{R}^{m\times n}$, OMP finds a vector of coefficients $\mathbf{y} \in \mathbb{R}^n$ with $k < m$ non-zero coefficients so that $\mathbf{Dy}$ equals to $\mathbf{x}$ approximatively.

For the sake of comparison, we build a variant of OMP called Winner-Take-All Orthogonal Matching Pursuit (WTA OMP). More precisely, let $\mathbf{X} \in \mathbb{R}^{m\times p}$ be a matrix whose columns are formed by p image patches of dimension m and $\mathbf{Y} \in \mathbb{R}^{n\times p}$ be a matrix whose columns are formed by p vec-

tors of coefficients of dimension n. WTA OMP first decomposes each image patch over $\mathbf{D}$, see (1). Then, it keeps the $\gamma \times n \times p$ coefficients with largest absolute value for the n-length sparse representation of the p patches and sets the rest to 0, see (2). The support of the sparse representation of each patch has therefore been changed. Hence the need for a final least-square minimization, see (3).

Algorithm 1 : WTA_OMP

Inputs: $\mathbf{X} \in \mathbb{R}^{m \times p}$, $\mathbf{D} \in \mathbb{R}^{m \times n}$, $k < m$ and $\gamma \in]0, 1[$.

$$\text{For each } j \in [\![1, p]\!], \mathbf{Y}_j = \text{OMP}\left(\mathbf{X}_j, \mathbf{D}, k\right) \tag{1}$$

$$\mathbf{I} = f_\gamma\left(\mathbf{Y}\right) \tag{2}$$

$$\text{For each } j \in [\![1, p]\!], \mathbf{Z}_j = \min_{\mathbf{z} \in \mathbb{R}^n} \|\mathbf{X}_j - \mathbf{D}\mathbf{z}\|_2^2 \text{ st. } \text{supp}\left(\mathbf{z}\right) = \text{supp}\left(\mathbf{I}_j\right) \tag{3}$$

Output: $\mathbf{Z} \in \mathbb{R}^{n \times p}$.

4. TRAINING

Before moving on to the image compression experiment in Section 5, SWTA AE needs training. Similarly, a dictionary $\mathbf{D} \in \mathbb{R}^{m \times n}$ must be learned for WTA OMP.

4.1. Training data extraction

We extract 1.0×10^5 RGB images from the ILSVRC2012 ImageNet dataset [17]. The RGB color space is transformed into YCbCr and we only keep the luminance channel.

For SWTA AE, the luminance images are resized to 321×321. $\mathbf{M} \in \mathbb{R}^{321 \times 321}$ denotes the mean of all luminance images. $\sigma \in \mathbb{R}_+^*$ is the mean of the standard deviation over all luminance images. Each luminance image is subtracted by $\mathbf{M}$ and divided by σ. These images are concatenated into a training set $\mathbf{\Delta} \in \mathbb{R}^{321 \times 321 \times \left(1.0 \times 10^5\right)}$.

For WTA OMP, $\eta = 1.2 \times 10^6$ image patches of size $\sqrt{m} \times \sqrt{m}$ are randomly sampled from the luminance images. We remove the DC component from each patch. These patches are concatenated into a training set $\mathbf{\Gamma} \in \mathbb{R}^{m \times \eta}$.

4.2. SWTA AE training

As explained in Section 2.2, α tunes the coding cost of $\mathbf{Z}$. If α is fixed during training, all the filters and the biases of SWTA AE are learned for one rate. That is why we turn α into a stochastic hyperparameter during training. This justifies the prefix "Stochastic" in SWTA AE. Since there is no reason to favor some rates during training, we sample α according to the uniform distribution $\mathcal{U}[\mu - \epsilon, \mu + \epsilon]$, where $\mu - \epsilon > 0$ and $\mu + \epsilon < 1$. We select $\mu = 1.8 \times 10^{-1}$ and $\epsilon = 1.7 \times 10^{-1}$ to make the support of α large. At each training epoch, α is drawn for each training image of $\mathbf{\Delta}$.

As shown in Section 2.1, SWTA AE can process images of various sizes. During training, we feed SWTA AE with random crops of size 49×49 of the training images of $\mathbf{\Delta}$. This accelerates training considerably. The training objective is to minimize the mean squared error between these cropped images and their reconstruction plus l_2-norm weights decay. We use stochastic gradient descent. The gradient descent learning rate is fixed to 2.0×10^{-5}, the momentum is 0.9 and the size of mini-batches is 5. The weights decay coefficient is 5.0×10^{-4}. Our implementation is based on Caffe [18]. It adds to Caffe the tools introduced in Sections 2.2 and 2.3.

4.3. Dictionary learning for WTA OMP

Given $\mathbf{\Gamma}$, $k < m$ and $\gamma \in]0, 1[$, the dictionary learning problem is formulated as (4).

$$\begin{aligned} \min_{\mathbf{D}, \mathbf{Z}_1, ..., \mathbf{Z}_\eta} & \frac{1}{\eta} \sum_{j=1}^{\eta} \|\mathbf{\Gamma}_j - \mathbf{D}\mathbf{Z}_j\|_2^2 \\ & \text{st. } \forall j \in [\![1, \eta]\!], \|\mathbf{Z}_j\|_0 \leq k \\ & \text{st. } \sum_{i=j}^{\eta} \|\mathbf{Z}_j\|_0 \leq \gamma \times n \times \eta \end{aligned} \tag{4}$$

(4) is solved by Algorithm 2 which alternates between sparse coding steps that involve WTA OMP and dictionary updates that use stochastic gradient descent. Given $\mathbf{\Gamma}$ and $p \in \mathbb{N}_+^*$, let ϕ be a function that randomly partitions $\mathbf{\Gamma}$ into $\eta_p = \eta \,/\, p$ mini-batches $\left\{\mathbf{X}^{(1)}, ..., \mathbf{X}^{(\eta_p)}\right\}$, where, for $i \in [\![1, \eta_p]\!]$, $\mathbf{X}^{(i)} \in \mathbb{R}^{m \times p}$. Mini-batches make learning very fast [19].

Algorithm 2 : dictionary learning for WTA OMP.

Inputs: $\mathbf{\Gamma} \in \mathbb{R}^{m \times \eta}$, $k < m$, $\gamma \in]0, 1[$, $p \in \mathbb{N}_+^*$ and $\varepsilon \in \mathbb{R}_+^*$.
$\mathbf{D} \in \mathbb{R}^{m \times n}$ is randomly initialized.
$\forall j \in [|1, n|],\ \mathbf{D}_j \leftarrow \mathbf{D}_j / \|\mathbf{D}_j\|_2$
For several epochs do:

$$\left[\mathbf{X}^{(1)}, ..., \mathbf{X}^{(\eta_p)}\right] = \phi\left(\mathbf{\Gamma}, p\right)$$

$$\forall i \in [|1, \eta_p|],\ \mathbf{Z}^{(i)} = \text{WTA\_OMP}\left(\mathbf{X}^{(i)}, \mathbf{D}, k, \gamma\right)$$

$$\mathbf{D} \leftarrow \mathbf{D} - \varepsilon \frac{\partial \left\|\mathbf{X}^{(i)} - \mathbf{D}\mathbf{Z}^{(i)}\right\|_F^2}{\partial \mathbf{D}}$$

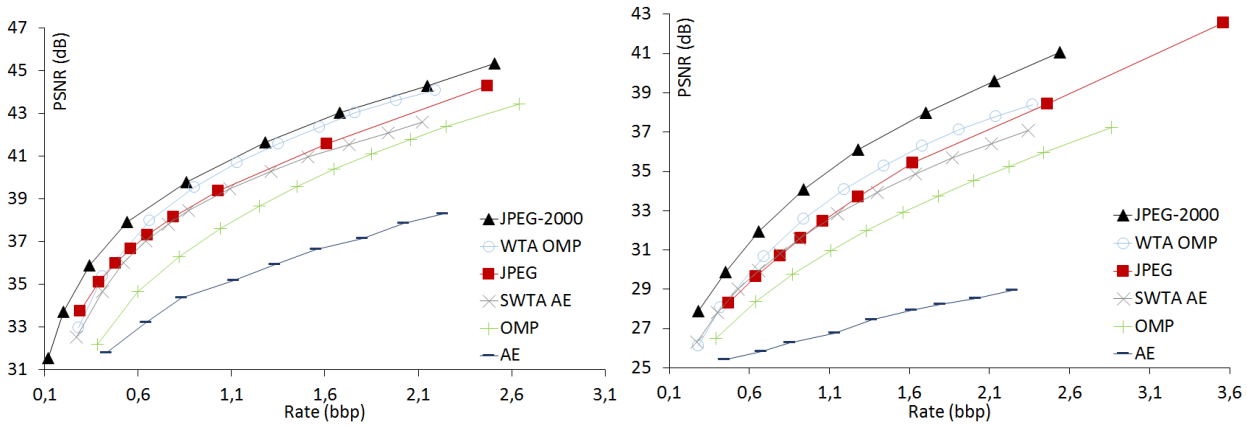
$$\forall j \in [|1, n|],\ \mathbf{D}_j \leftarrow \mathbf{D}_j / \|\mathbf{D}_j\|_2$$

Output: $\mathbf{D} \in \mathbb{R}^{m \times n}$.

For OMP, given $\mathbf{\Gamma}$, a dictionary $\mathbf{D}' \in \mathbb{R}^{m \times n}$ is learned using K-SVD [16] [1], and the parameters m and n are optimized with an exhaustive search. This leads to $m = 64$ and $n = 1024$. For SWTA AE, the same values for m and n are used for training $\mathbf{D}$ via Algorithm 2. Moreover, $k = 15$, $\gamma = 4.5 \times 10^{-3}$, $p = 10$ and $\varepsilon = 2.0 \times 10^{-2}$.

[1] K-SVD code: http://www.cs.technion.ac.il/ elad/software/

Fig. 2: Evolution of PNSR with the rate.



(a) LENA luminance 512×512.

(b) BARBARA luminance 480×384.

5. IMAGE COMPRESSION EXPERIMENT

After training in Section 4, we compare the rate-distortion curves of OMP, WTA OMP, SWTA AE, JPEG and JPEG2000 on test luminance images.

5.1. Image CODEC for SWTA AE

Each input test luminance image is pre-processed similarly to the training in Section 4.1. The mean learned image $\mathbf{M}$ is interpolated to match the size of the input image. Then, the input image is subtracted by this interpolated mean image and divided by the learned σ. The encoder of SWTA AE computes $\mathbf{Z}$. The bitstream is obtained by processing $\mathbf{Z}$ as detailed in Section 2.3.

5.2. Image CODEC for OMP and WTA OMP

A luminance image is split into 8×8 non-overlapping patches. The DC component is removed from each patch. The DC components are uniformly quantized over 8-bits and coded with a fixed-length code. OMP (or WTA OMP) finds the coefficients of the sparse decompositions of the image patches over $\mathbf{D}'$ (or $\mathbf{D}$). The non-zero coefficients are uniformly quantized over 8-bits and coded with a Huffman code while their position is coded with a fixed-length code.

Then, for WTA OMP only, the number of non-zero coefficients of the sparse decomposition of each patch over $\mathbf{D}$ is coded with a Huffman code.

5.3. Comparison of rate-distortion curves

In the literature, there is no reference rate-distortion curve for auto-encoders. We compare SWTA AE with JPEG and JPEG2000 [2] even though the image CODEC of SWTA AE is less optimized. Furthermore, we compare SWTA AE with its non-sparse Auto-Encoder counterpart (AE). AE has the same architecture as SWTA AE but its $\mathbf{Z}$ only contains non-sparse feature maps. Note that, to draw a new point in the AE rate-distortion curve, AE must be first re-trained with a different number of feature maps in $\mathbf{Z}$.

Figure 2 shows the rate-distortion curves of OMP, WTA OMP, AE, SWTA AE, JPEG and JPEG2000 for two of the most common images: LENA and BARBARA. In terms of rate-distortion trade-off, SWTA AE outperforms AE and WTA OMP is better than OMP. This highlights the value of WTA for image compression. When we compare SWTA AE with WTA OMP, we see that iterative decomposition is more efficient for image compression using sparse representations. Moreover, SWTA AE can compete with JPEG. We also ran this image compression experiment on several crops of LENA and BARBARA and observed that the relative position of the six rate-distortion curves was comparable to the relative positioning in figure 2. The size of the test image does not affect the performance of SWTA AE. More simulation results and a complexity analysis for OMP, WTA OMP and SWTA AE can be found on the web page `https://www.irisa.fr/temics/demos/NeuralNets/AutoEncoders/swtaAE.htm`.

6. CONCLUSIONS AND FUTURE WORK

We have shown that, SWTA AE is more adaptated to image compression than auto-encoders as it performs variable rate image compression for any size of image after a single training and provides better rate-distortion trade-offs.

So far, our work has focused on the layer of auto-encoders which is dedicated to coding. Yet, many avenues of research are still to be explored to improve auto-encoders for image compression. For instance, [20] proves that removing a max-pooling layer and increasing the stride of the previous convolution, as we do, harms neural networks. This has to be addressed.

[2] JPEG and JPEG2000 code: http://www.imagemagick.org/script/index.php

7. REFERENCES

[1] Geoffrey Hinton and Ruslan Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313 (5786), pp. 504–507, July 2006.

[2] Alex Krizhevsky and Geoffrey Hinton, "Using very deep autoencoders for content-based image retrieval," in *ESANN*, 2011.

[3] L. Deng, M. Seltzer, D. Yu, A. Acero, A. Mohamed, and G. Hinton, "Binary coding of speech spectrograms using a deep auto-encoder," in *Interspeech*, 2010.

[4] George Toderici, Sean O'Malley, Sung Jin Hwang, Damien Vincent, David Minnen, Shumeet Baluja, Michele Covell, and Rahul Sukthankar, "Variable rate image compression with recurrent neural networks," in *ICLR*, 2016.

[5] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton, "Speech recognition with deep recurrent neural networks," in *ICASSP*, 2013.

[6] Heiko Schwarz, Detlev Marpe, and Thomas Wiegand, "Overview of the scalability video coding extension of the H.264/AVC standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17 (9), pp. 1103–1120, September 2007.

[7] Gary J. Sullivan, Jim M. Boyce, Ying Chen, Jens-Rainer Ohm, C. Andrew Segal, and Anthony Vetro, "Standardized extensions of high efficiency video coding (HEVC)," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7 (6), pp. 1001–1016, December 2013.

[8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, "ImageNet classification with deep convolutional neural networks," in *NIPS*, 2012.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Delving deep into rectifiers: surpassing human-level performance on imagenet classification," in *ICCV*, 2015.

[10] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han, "Learning deconvolution network for semantic segmentation," in *ICCV*, 2015.

[11] Dominik Scherer, Andreas Müller, and Sven Behnke, "Evaluation of the pooling operations in convolutional architectures for object recognition," in *ICANN*, 2010.

[12] Matthew Zeiler and Rob Fergus, "Visualizing and understanding convolutional networks," in *ECCV*, 2014.

[13] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla, "SegNet: a deep convolutional encoder-decoder architecture for image segmentation," *arXiv preprint arXiv:1511.00561*, 2015.

[14] Alireza Makhzani and Brendan Frey, "Winner-take-all autoencoders," in *NIPS*, 2015.

[15] Mark Davenport and Michael Wakin, "Analysis of orthogonal matching pursuit using the restricted isometry property," *IEEE Transactions on Information Theory*, vol. 56 (9), pp. 4395–4401, September 2010.

[16] Ori Bryt and Michael Elad, "Compression of facial images using the K-SVD algorithm," *Journal of Visual Communication and Image Representation*, vol. 19 (4), pp. 270–282, May 2008.

[17] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li, "ImageNet: a large-scale hierarchical image database," in *CVPR*, 2009.

[18] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell, "Caffe: convolutional architecture for fast feature embedding," in *ACM MM*, 2014.

[19] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky, "Lecture 6a, overview of mini-batch gradient descent," `http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf`.

[20] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller, "Striving for simplicity: the all convolutional net," in *ICLR*, 2015.