Deep Learning for Image Compression

, IST 597: Foundations of Deep Learning, Penn State
RAJEEV BHATT AMBATI, Electrical Engineering
ANKUR MALI, Information Sciences and Technology
MIN-CHUN WU, Mathematics

Abstract

With increase in usage of multimedia device, there is an exponential increase in usage and creation of data leading to data explosion. To deal with such high quality and even higher quantity multimedia object spread across world wide web, we need better data transfer and storage mechanisms. Hence, image compression plays a vital role in achieving this objective. Due to highly complex and randomized correlations between objects or pixels, it is highly complicated to obtain any uniform representations for images, thus it is difficult to retrieve original information from such representations. We aim to achieve comparable performance in terms of compression ratio when compared to standard lossy compression techniques like JPEG and JPEG2000 and also maintain the quality of reconstructed image. We developed an encoder-decoder based model using a Generative Adversarial Network (GAN) that compresses the image in spatial domain and outperforms the standard techniques described above.

Additional Key Words and Phrases: image compression, compression, deep learning

ACM Reference Format:

, Rajeev Bhatt Ambati, Ankur Mali, and Min-Chun Wu. 2017. Deep Learning for Image Compression. *ACM Trans. Graph.* 9, 4, Article 39 (December 2017), 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

In mathematical theory, we consider artificial neural networks as universal function approximators. Due to this property, neural networks were widely used in literature for image compression [Jiang 1999]. However, until recently, we were lacking efficient hardware and algorithms to achieve compression using neural networks. [Toderici et al. 2015] recently proposed efficient architecture to achieve better compression rates using deep neural networks. A Recurrent Neural Network along with 1×1 convolution layer is used to perform compression and achieved better results than JPEG on standard kodak Image Compression benchmark [Toderici et al. 2016].

This paper introduced a recurrent autoencoder where each iteration encodes the residual between the original image and the reconstructed image in the previous iteration. It saves the bits by a progressive encoding of the input image. The bit rates used by [Toderici et al. 2015] are different across iterations by are constant across each image. But in fact the number of bits required to encode an image depends on the entropy of a particular image. [Johnston et al. 2017] takes on this recurrent approach and further improved the number of bits per pixel by using spatially adaptive bit rates. This paper also modified the standard Structural Similarity Index Measure (SSIM) loss and thereby the quality of reconstructed images is increased. It further argues that the initial hidden layer representations are vital for reconstructing high quality images and introduces a concept called "Priming". Hidden state priming essentially performs the first encoder-decoder iteration in k-steps and processes the k^{th} reconstructed image for the further iterations.

Recently a group of Researchers experimented with Generative Adversial Networks and obtained visually pleasing images with higher compression rates [Santurkar et al. 2017]. However, the reconstructed images were different from the original images. Frequent occurrences of particular pixel values in an image determines the most dominant color or luminance within an image. [Theis et al. 2017] proposed a better compressive autoencoder which utilizes entropy coding and also involves representing bits based on how frequent particular pixel values appears within an image.

Quantization is a process of representing a set of continuous inputs with discrete symbols while preserving important information. Depending on how many symbols we use, the size of the signal representation varies and hence, quantization is a vital step in image compression. Specific applications include Discrete Cosine Transform (DCT) data quantization in JPEG and Discrete Wavelet Transform (DWT) in JPEG 2000 [Taubman and Marcellin 2001], two of the standard image compression techniques. A typical video codec works by typically breaking the picture into blocks for example 8×8 blocks in case of [Mitchell et al. 1996]. High frequency components in images has complex structure and hence require effective methods to compress. Human eye is not so good at distinguishing the exact strength of a high frequency component variation in brightness. Many compression techniques exploit this fact by just dividing each high frequency component by an appropriate constant, thereby reducing its strength and the number of bits required to encode it. For example in JPEG encoding, after applying Discrete Cosine Transform (DCT) to an 8×8 block, DCT coefficients are quantized differently depending on their visual importance: high frequency components are typically encoded with more symbols (A sky needs less number of bits to encode than a flower).

There are two types of compression, namely lossy and lossless compression depending on whether the original image can be recovered from the compressed image. Lossy compression capitalizes on discarding redundant information present in the image and thereby decreasing the bits per pixel used to encode and image. Most popular lossy image compression techniques include JPEG and JPEG2000. Many compression techniques focus on the idiosyncrasies of human

Authors' addresses: IST 597: Foundations of Deep Learning, Penn State; Rajeev Bhatt Ambati, Electrical Engineering, rza67@psu.edu; Ankur Mali, Information Sciences and Technology, aam35@psu.edu; Min-Chun Wu, Mathematics, mpw5326@psu.edu.

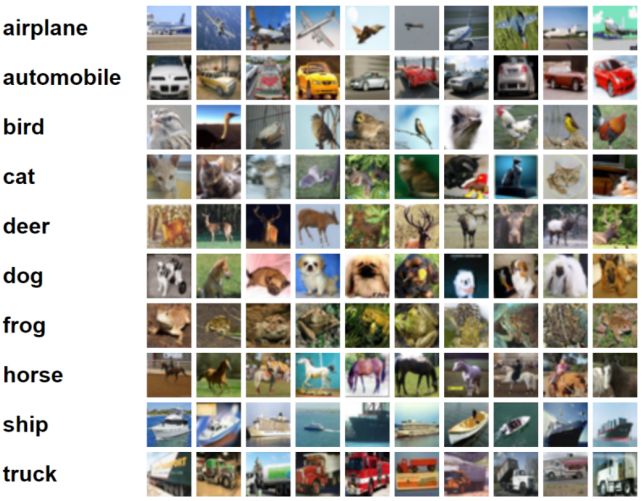

Fig. 1. 10 random images from each of the 10 classes are arranged row-wise.

physiology, taking into account, for instance, that the human eye can see only certain wavelengths of light. Even though, flaws are encountered in lossy compression which are well noticeable to the human eye for example the JPEG compression artifacts. Though we might be able to compress the data effectively in terms of bit rate, sometimes recovering the original image is important for example in case of compressing X-ray scans. This type of compression is called lossless compression. We will be performing lossy image compression for our project.

We experimented with several Deep Learning methodology on image compression problem. We primarily experimented with architecture like autoencoder and GANs. Since model works in encoderâĂŞdecoder fashion, we obtained better results than JPEG in terms of reconstructed image quality. We have used CIFAR-10 dataset for experiments. It consists of 60000 32×32 color images in 10 classes, with 6000 images per class divided into 50000 training images and 10000 test images. These 10 classes are mutually exclusive without any overlap between trucks and automobiles which include sedans, SUVs and the likes. A few samples from the dataset are shown in Figure 1 [Alex Krizhevsky and Hinton [n. d.]].

2 IMAGE QUALITY ASSESSMENT

While achieving higher compression rates is the goal of image compression, the quality of the uncompressed image should also be visually pleasing to the human eye. Since human subjective evaluation of the entire dataset is a tedious task, there is a necessity for using an objective measure for evaluating the quality of reconstructed image. This section covers existing image quality assessment (IQA) techniques and discusses their suitability for our problem.

Earlier IQA techniques include Mean Square Error (MSE), and Peak Signal to Noise Ratio (PSNR). For two images I_{orig}, the original image and I_{recon}, the reconstructed image, both of size $m \times n$, the above metrics are defined as follows:

$$MSE = \frac{1}{mn} \sum_{i=0}^{i=m-1} \sum_{j=0}^{j=n-1} [I_{orig}(i,j) - I_{recon}(i,j)]^2 \quad (1)$$

$$PSNR = 10 \times \log_{10}(\frac{MAX^2_{I_{orig}}}{MSE}) \quad (2)$$

Where $MAX_{I_{orig}}$ is the maximum possible pixel value of the image. When pixels are represented using 8 bit per sample, this is 255. Typical values for the PSNR in lossy image and video compression are between 30 and 50 dB, provided the bit depth is 8 bits, where higher is better. Both MSE and PSNR measure absolute error whereas, we need a metric for measuring the change in perceptual quality. SSIM [Wang et al. 2004] is a metric that views image degradation as a perceived change in structural information, while also incorporating important perceptual phenomena, including both luminance masking and contrast masking terms. The expression for measuring SSIM quality score is given below:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (3)$$

where μ_x, μ_y are the averages and σ_x^2, σ_y^2 are the variances of x and y respectively. σ_{xy} is the covariance of x and y. c_1, c_2 are two constants to avoid division by zero. The resultant SSIM index is a decimal value between -1 and 1, and value 1 is obtained only when comparing identical images. A high SSIM score indicates a good quality image. Figure 2 shows a visualization of an example comparing MSE and SSIM. Since PSNR is just a logarithm of MSE excluding the constant, it's reasonable to assume PSNR also performs similarly. Moreover, [Hore and Ziou 2010] analyses the difference between SSIM and PSNR. So far in many applications SSIM turned out to be a better perceptual similarity measure than MSE and PSNR.

3 COMPRESSION USING AUTOENCODERS

Autoencoders are special type of neural networks which represents input image into least possible feature maps as compared with original image feature map. This process is termed as dimensionality reduction, but we have to make sure that our model is able to recover or approximate original features from the reduced feature representation. This basically means that reconstruction error should be as low as possible. A typical autoencoder consists of two networks: one serves as an encoder, which projects the input into a latent space, a lower dimensional manifold in our case and the other serves as a decoder, which reconstructs the original input from the representation in latent space. Recent advancements have shown that plain autoencoders are comparatively less efficient when compared with standard compression techniques and the ones using deep learning architectures. Hence we propose a complex architecture which reduces information in spatial domain and can also recover data from these compressed signals. Following are a few architectures we have implemented:

3.1 Convolutional Autoencoder

Convolution Neural networks (CNN) are widely accepted neural network architectures which capture spatial and non-spatial information within an image. Hence, we tried to combine CNN along

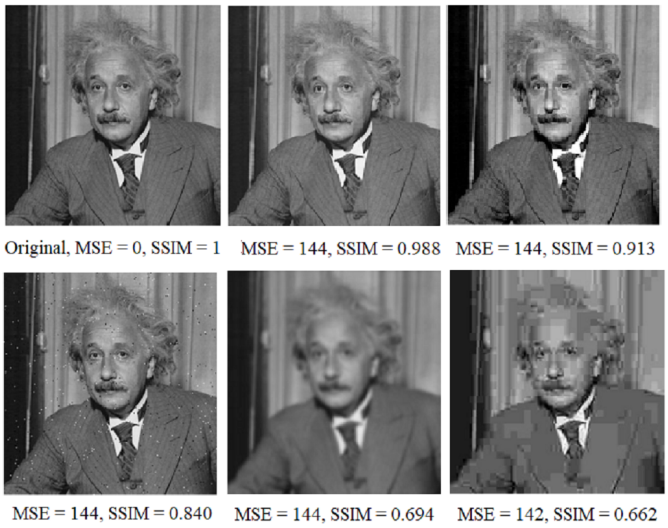

Fig. 2. A sample image showing a comparison of SSIM and MSE

with autoencoders and developed a fully convolutional autoencoder architecture(CNN-AE). Feature Representation or obtained features are termed as latent representation of the input signal. Recent advancements have shown that plain autoencoders are comparatively less efficient when compared with standard compression and the ones using deep learning architectures. Hence we have used a complex architecture which reduces information in spatial domain and can also recover data from these compressed signals. Convolution Neural networks are widely accepted neural network architectures which capture spatial and non-spatial information within the image. Hence, we tried to combine CNN along with autoencoders and developed a fully convolutional autoencoders architecture(CNN-AE). In Encoder our main objective is to reduce dimension of the image using max pooling operator. Whereas, to get better reconstruction and minimal loss of information in decoder, we used interpolation functions, followed by convolution operator to perform deconvolution for input latent representation. The general layout of architecture is shown in Figure 3. It is inspired from the standard CNN architecture introduced by [Krizhevsky et al. 2012] with a few modifications.

In the encoder network, every convolution layer is followed by a Rectified Linear Unit (ReLU) activation layer. Though we use a shallow architecture with only a few layers, ReLU activation would prevent the gradients from saturating and help the training process to converge faster within fewer iterations. Since our input size is small with just 32×32 images, we have used only two maxpooling layers. In the decoder, deconvolution is performed by using an unpooling layer followed by a convolution layer. Initially two convolution layers are used back-to-back to give the model enough capacity in recovering the original image and also to avoid checkerboard artifacts often seen in a deconvolution operation.

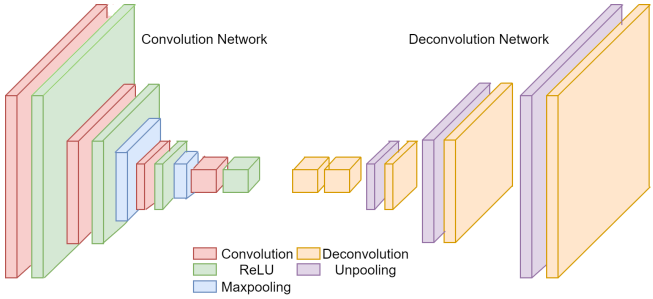

Fig. 3. Architecture of the convolutional autoencoder.

3.2 Recurrent Convolutioinal Autoencoder

Though, CNN-AE Architecture showed some promising results, the gradients would saturate after N number of iterations; thus not contributing in better reconstruction error. A combination of CNN with Recurrent Neural Networks are widely used architectures in literature showing state of the art performance in many Computer Vision tasks. Thus based on similar idea, we experimented with adding recurrence to our network. While compressing images, we do introduce some error in the ouput and our aim is to minimize this reconstruction error; thus we apply recurrence after each step treating each reconstruction loss as a sequence loss in each time step and applying our CNN-AE to minimize this sequence loss. Our approach thus introduces certain degree of trade-off between image quality and image compression ratio. We trained our model on cifar -10 dataset containing 32*32 images and evaluated our model performance on cifar-10 Test set along with Few Higher Resolution images down sampled to be divisible by 32.

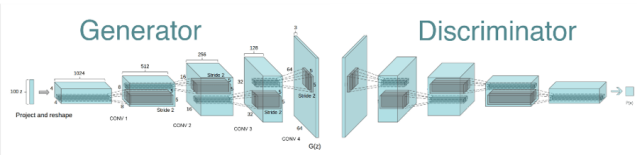

Fig. 4. Architecture of the Generative Adversarial Network.

3.3 Generative Adversarial Network Autoencoder

Even though we have similar images in terms of contents, their feature map or latent representation is different. We can model our problem statement based on this understanding and can try to create an architecture which can map two latent representations. In the recent past, Generative Adversarial Networks have shown promising results in generating realistic natural images, this means they can learn underlying latent representation of images and thus generate more realistic images. However, we have to decide the trade-off in terms of input signals and restrict our model to obtain sufficient information. We do not need our model to generate realistic but different images. If input signal is a Gaussian distribution over entire sample, then the end result would be different than the original signal. Hence, we provide Gaussian distribution over compressed image as our input feature map. Our encoder-decoder architecture using GAN is shown in Figure 4. We experimented with 2 GAN architectures, namely Deep Convolutional Generative Adversarial Network (DCGAN) [Radford et al. 2015] and Wasserstein Generative Adversarial Network (WGAN) [Arjovsky et al. 2017]. Figure 5 shows the typical generator architecture modified to meet our objective.

However, based on our initial experiments, we found that simple GAN loss is not efficient of our objective due to two reasons. First, the gradients saturated after a few epochs and thus unsuccessful in reducing reconstruction error. Second, the GAN architecture suffers with instability issues while trying to reduce Jensen-Shannon distance; thus fails in producing continuous functions when input probabilities are in low-dimension feature space. Based on these two observations, we experimented with hybrid loss since Wasserstein loss helps in reducing stability issues in GANs. Hence, we combined Wasserstein loss with Mean Squared Error (MSE) loss to train our generative model. We combined our stable GAN with autoencoder, which generates compressed representation of the input image and then fed as input to the generative model. We evaluated our model on various hybrid loss functions and found the best performing architecture for our objective. Figure 5 shows sample images generated using our network on CIFAR-10 test set.

4 EXPERIMENTS

In this project, we worked on block based compression methods by dividing input image into 32×32 non-overlapping blocks and then fed them to several autoencoder and GAN architectures for lossy compression. To have fair comparison, we compressed JPEG by 16:1 compression ratio and our models are also reduced in similar fashion.

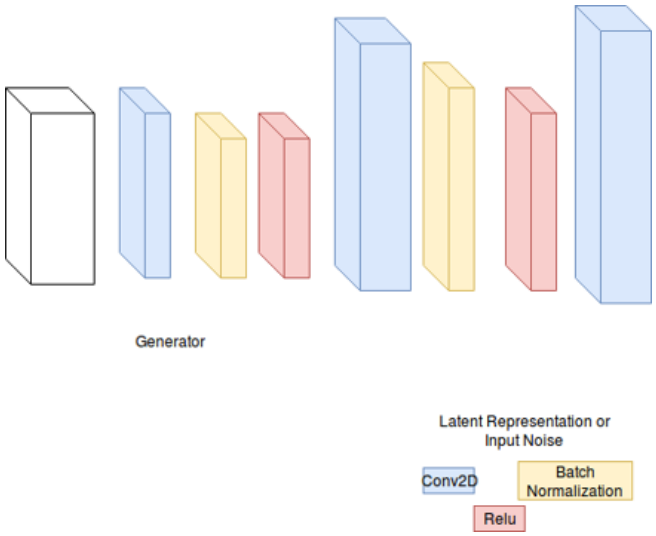

Fig. 5. Architecture of the Generative Adversarial Network.

4.1 Autoencoders

We experimented with various autoencoder architectures and observed that fully convolutional models are better suitable to our purpose in terms of performance. The compression ratio 16:1 was kept consistent across all our experiments. The vital part of designing experiments using autoencoder is that we cannot use large number of filters before our latent input. Thats because these features represent the compressed image. We had to experiment with the trade-off between number of filters and latent representation. Second, our input size being small with just 32×32 images, very deep networks will overfit the dataset and can introduce bias trade-off on the training dataset. Thus, based on our experiments, a 3 layer CNN-AE with 3 recursive steps, which we call CNN-AE-RNN performed better on CIFAR-10 test set. In our experiments data was normalized between 0 and 1, and gradients were clipped between -0.015 to 0.015. For all the variations, we used Adam Optimizer with a learning rate 0.00002 and standard MSE was used as loss function to train our network. Early stopping is used in all the networks and are trained for 1000 epochs. CNN-AE-RNN architecture showed promising results on CIFAR-10 test set and were better than the ones obtained using standard JPEG algorithm.

4.2 Generative Adversarial Network

The first GAN network has two units: a Compression unit(Generator), which takes input from the autoencoder and a discriminator network, which outputs the probability of the generated image being real. Our architecture is inspired from Resnet network with a few changes in the internal parameters. Since out input dimensions are small with just 32×32 images, we adopted a shallow network. We initialized our parameters using Xavier initialization and also clipped the weights between -0.015 and 0.015. All our networks are trained for 500 epochs. For WGAN, we used a learning rate of 0.00002 and RMSProp as the optimizer. Our model showed stability when we used LeakyRELU in discriminator. For DCGAN, we found

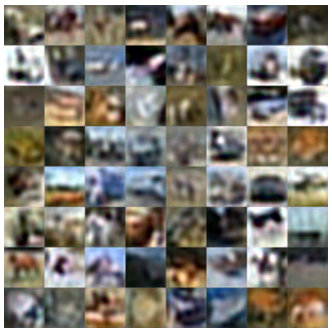

Fig. 6. Sample images generated using GAN-AE after 10,000 iterations

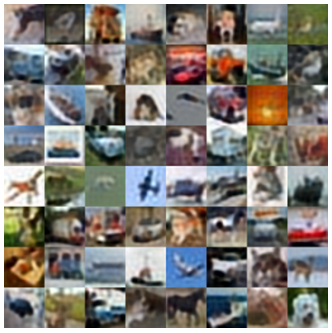

Fig. 7. Sample images generated using GAN-AE after 500,000 iterations

that, a learning rate 0.0002 along with Adam Optimizer achieved optimal performance. We experimented with classical GAN loss like DC loss, Wasserstein loss and hybrid losses such as DC loss + L1 loss, DC Loss + L2 Loss, WLoss + L1 Loss and Wloss + L2 Loss. Based on our initial experiments, we observed that L2 loss with Wasserstein metrics stabilized the training procedure of GAN thereby, leading to better reconstruction error. If performed the training for enough iterations, GAN-AE produced promising results. Figure 6 and Figure 7 shows a sample of images generated in the due process of training.

5 RESULTS

We compared our algorithms with the widely used JPEG algorithm. Even though there are many algorithms out there which claims to be better than JPEG, we can hardly find any open sourced implementation of them. Even Google's full end-to-end compression model is not available for training and also the testing module requires heavy amount of computing resources which wasnâĂŹt doable in assigned time frame. In this paper, we compared our results with JPEG and have shown better results than our baseline. For lossy image compression, we compared our results using both CIFAR-10 and higher resolution images including few from Kodak benchmark dataset. For our best models, we have shown the results in Table 1 and Table 2. All results are obtained using fair comparison with a compression ratio 16:1 on both test Dataset. Table 1 shows that our approach of using hybrid loss thus plays a vital role and outperforms JPEG by a large margin. GAN-AE with L1 + DC loss performs worst than JPEG but, rest of our models outperform JPEG in terms of PSNR. More training time guaranteed sharper images but, in some cases, we observed blurriness in output image. Hence, after successive experiments, we found a trade-off between optimal training rate and hyperparameters. A comparison of the reconstructed images of CIFAR-10 using JPEG and GAN is shown in Figure 8. The performance of GAN-AE on high resolution image and Kodak benchmark dataset is shown in Figures 9 and 10 respectively.

Table 1. Results on CIFAR-10 dataset.

Method	PSNR
JPEG	**28.244**
GAN-AE (L1, DC)	28.236
GAN-AE (L1, W)	30.048
GAN-DC	30.457
GAN-WGAN	30.9602
GAN-AE (L2, DC)	31.5468
CNN-RNN-AE	31.4092
GAN-AE (L2, W)	**36.9802**

Table 2. Results on high resolutions images.

Method	PSNR	SSIM
JPEG	33.2	0.986
GAN-AE	33.301	0.9871
CNN-RNN-AE	30.2	0.9718

6 CONCLUSION

We implemented an efficient image compression algorithm which succeeds in obtaining a better reconstruction of images, thus reducing the trade-off between image compression and image quality. We have shown that our algorithm can compress images at a lower bit rate and generate better images when compared with standard JPEG algorithm. However, compression is happening in the spatial domain and in classic compression literature, we cannot achieve compression until and unless we have loss in bits per pixel(Bpp). Since our model is trained without using either entropy coding or arithmetic coding, compression in terms of Bpp is not available at the time of submission. Further experiments need to be done to achieve compression in spatial as well as in Bpp stages.

REFERENCES

Vinod Nair Alex Krizhevsky and Geoffrey Hinton. [n. d.]. CIFAR-10. ([n. d.]). https://www.cs.toronto.edu/~kriz/cifar.html

Martin Arjovsky, Soumith Chintala, and LÃĺon Bottou. 2017. Wasserstein GAN. *CoRR* abs/1701.07875 (2017). arXiv:1701.07875 https://arxiv.org/abs/1701.07875

Alain Hore and Djemel Ziou. 2010. Image Quality Metrics: PSNR vs. SSIM. In *Proceedings of the 2010 20th International Conference on Pattern Recognition (ICPR '10)*. IEEE Computer Society, Washington, DC, USA, 2366–2369. https://doi.org/10.1109/ICPR.2010.579

J. Jiang. 1999. Image compression with neural networks - A survey. In *Signal Processing: Image Communication 14*. 737–760.

Nick Johnston, Damien Vincent, David Minnen, Michele Covell, Saurabh Singh, Troy T. Chinen, Sung Jin Hwang, Joel Shor, and George Toderici. 2017. Improved Lossy Image Compression with Priming and Spatially Adaptive Bit Rates for Recurrent Networks. *CoRR* abs/1703.10114 (2017). arXiv:1703.10114 http://arxiv.org/abs/1703.10114

Fig. 8. reconstructed images obtained from JPEG compression are shown on the left and from GAN-AE are shown on the right.

Fig. 9. Performance of GAN-AE on high resolution images with number of epochs, 10 epochs on the left, 100 epochs in the middle and 500 epochs on the right.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'12)*. Curran Associates Inc., USA, 1097–1105. http://dl.acm.org/citation.cfm?id=2999134.2999257

Joan L. Mitchell, William B. Pennebaker, Chad E. Fogg, and Didier J. Legall (Eds.). 1996. *MPEG Video Compression Standard*. Chapman & Hall, Ltd., London, UK, UK.

Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *CoRR* abs/1511.06434 (2015). arXiv:1511.06434 http://arxiv.org/abs/1511.06434

Shibani Santurkar, David M. Budden, and Nir Shavit. 2017. Generative Compression. *CoRR* abs/1703.01467 (2017). arXiv:1703.01467 http://arxiv.org/abs/1703.01467

David S. Taubman and Michael W. Marcellin. 2001. *JPEG 2000: Image Compression Fundamentals, Standards and Practice*. Kluwer Academic Publishers, Norwell, MA, USA.

Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc HuszÃąr. 2017. Lossy Image Compression with Compressive Autoencoders. *CoRR* abs/1703.00395 (2017). arXiv:1703.00395 https://arxiv.org/abs/1703.00395

George Toderici, Sean M. O'Malley, Sung Jin Hwang, Damien Vincent, David Minnen, Shumeet Baluja, Michele Covell, and Rahul Sukthankar. 2015. Variable Rate Image Compression with Recurrent Neural Networks. *CoRR* abs/1511.06085 (2015). arXiv:1511.06085 http://arxiv.org/abs/1511.06085

George Toderici, Damien Vincent, Nick Johnston, Sung Jin Hwang, David Minnen, Joel Shor, and Michele Covell. 2016. Full Resolution Image Compression with Recurrent Neural Networks. *CoRR* abs/1608.05148 (2016). arXiv:1608.05148 http://arxiv.org/abs/1608.05148

Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. 2004. Image Quality Assessment: From Error Visibility to Structural Similarity. *Trans. Img. Proc.* 13, 4 (April 2004), 600–612. https://doi.org/10.1109/TIP.2003.819861

Received December 2017

Fig. 10. Performance of GAN-AE on Kodak dataset with number of epochs, 10 epochs on the left, 100 epochs in the middle and 500 epochs on the right.