Image Compression with Deeper Learned Transformer

Licheng Xiao, Hairong Wang and Nam Ling
Santa Clara University, CA, USA
E-mail: lxiao@scu.edu

***Abstract*— Deep learning is known for its flexibility and infinite potential to approximate any function. Is it possible to approximate image compression using deep learning? The answer is yes. This article compares three major deep learning techniques used in image compression now and proposed an approach with deeper learned transformer and improved optimization goal, which achieved improved peak signal-to-noise ratio (PSNR) and multi-scale structural similarity (MS-SSIM) under very low bits per pixel (bpp). Experimental results show that the proposed approach outperformed BPG (RGB 4:4:4) in natural scene images compression, and is capable to handle arbitrary image shapes, which makes it applicable to practical image compression workloads.**

I. INTRODUCTION

Traditional image compression methods have a long history of development and are quite mature and complex now. From widely used JPEG to its successor JPEG2000 [1], and the latest state-of-the-art BPG [2], traditional lossy image compression methods have improved a lot in both reconstructive quality and compression ratio, it seemed hard to exceed BPG in a short time.

Different from traditional image compression methods, deep learning approach seems simpler. Deep learning approaches may use only deep neural networks to approximate any function it needs for image compression. By adjusting network structures, optimization goals, and optimization methods, the performance of such approximation can see improvement towards theoretical optimal status.

Regarding using deep learning to approximate lossy image compression, there have been several milestone research work conducted these years.

There are three major deep learning techniques used in image compression recently. The most important one is deep convolutional autoencoder. The second is generative adversarial networks (GAN); and the third is super resolution using deep neural networks.

Deep convolutional autoencoder, represented by variational autoencoder applied to end-to-end image compression [3], outperformed the other two approaches, because its optimization goal can be set directly to minimize the difference between original image and reconstructed image, together with compressed file size, using metrics like mean square error (MSE), peak-signal-to-noise-ratio (PSNR), multi-scale structural similarity (MS-SSIM), and bits per pixel (bpp). This ensures that the deep neural network is always trained toward a correct direction. To the best of our knowledge, there are two deep convolutional autoencoder based approaches that already outperform BPG using deep learning. The first is by combining hierarchical entropy model with autoregressive priors [4]. The second is by context-adaptive entropy [5].

GAN is good at generating images similar to original images, but it is not designed to minimize the difference in a controlled way. There are no explicit metrics to tell the difference between original images and reconstructed images, and the discriminator has to learn to distinguish arbitrary reconstructed images from all original images, which is much more complex than using autoencoder, which just need to compare one pair of images with explicit metrics. Experimental results have shown that GAN-based image compression often come with some visible difference in details, although it is impressive in compression ratio and image sharpness [6].

Super resolution using deep neural networks, represented by enhanced deep residual networks for single image super resolution (EDSR) [7], is the most limited approach because it is supervised learning, which needs both low-resolution image and corresponding high-resolution image as training data. Since low-resolution images are often achieved by traditional methods, such as bicubic down-sampling, or using some deep neural networks that approximate bicubic down-sampling, the accuracy of compressed data representation is limited by the traditional approach it used as reference, which makes it not as competitive as autoencoder in image quality, and not as competitive as GAN in compression ratio.

Here we propose an improved deep convolutional autoencoder for end-to-end lossy image compression, which is optimized for RMSE (Root Mean Square Error) and bpp (bit per pixel), and also has deeper and doubled network structure. Our approach is different from previous work that set optimization goals only on MSE (Mean Square Error) and bpp [3], or MS-SSIM and bpp [8], or a choice between optimal MSE and optimal MS-SSIM (multi-scale structural similarity) [5]. Experimental results showed that our proposed approach outperformed previous state-of-the-art artificial neural network (ANN)-based image compression approaches represented by Ballé [3], as well as BPG (RGB 4:4:4).

II. IMPROVED OPTIMIZATION GOAL

One contribution of our work is the improved optimization goal. Since optimization goal has substantial influence on the performance of a trained ANN, we are especially interested in

modifying the optimization goal, and look for better performance. As proved by previous experiments by Ballé [8], if the autoencoder was optimized for MSE, the result was better than JPEG2000, and just slightly weaker than BPG. If the autoencoder was optimized for MS-SSIM, the result was similar in overall fidelity, but varied for images with different local contrast. Although experimental results exceeded their previous ANN approaches, none of them exceeded BPG. Even in the latest two approaches that outperformed BPG, none of these results were achieved by improving the optimization goal. Here we tried to improve the optimization goal.

Optimization goal, i.e. objective function or loss function of a variational autoencoder [3] can be expressed as (1), with the total loss (denoted by $\mathcal{L}$), of encoder (denoted by g_a), decoder (denoted by g_s), and the discrete probability distribution of the quantized vector (denoted by P_q).

When encoding the input image (denoted by x), the input image x is first transformed by encoder g_a into a continuous-valued vector (denoted by y in Fig. 1, Fig.2 and Fig. 3), and then quantized into a discrete-valued vector (denoted by q) through a uniform scalar quantizer, which round each element to the nearest integer (denoted by Q), and finally use an entropy encoder to encode the quantized vector q to generate the compressed file.

When decoding the compressed file, first use an entropy decoder to decode the compressed file into quantized vector q, then reinterpret the quantized vector q from discrete-valued vector to continuous-valued vector (denoted by $\hat{y}$ in Fig. 1, Fig. 2 and Fig. 3) using interpreter function (denoted by I), then use decoder g_s to synthesis reconstructed image (denoted by $\hat{x}$).

The loss function equals to the entropy of the discrete probability distribution of the quantized vector, plus the distance (denoted by d) between input image x and reconstructed image $\hat{x}$ multiplied by Lagrange multiplier lambda (denoted by λ).

$$\mathcal{L}[g_a, g_s, P_q] = -\mathbb{E}[\log_2(P_q)] + \lambda\mathbb{E}[d(x,\hat{x})]$$
$$where\ q = Q(g_a(x)) \quad (1)$$
$$and\ \hat{x} = g_s(I(q))$$

In our approach, we leave the entropy part unchanged, and modify the distance part. Originally in Ballé's design [3], the distance function is just the MSE of perceptual space, which can be expressed as (2), where x denote input image and $\hat{x}$ denote reconstructed image. The problem of using MSE as the measurement of distance is that if we make a single very bad prediction, the squaring will make the error even worse and it may skew the metric towards overestimating the model's badness. That is a particularly problematic behavior if we have noisy data. On the other hand, if all the errors are small, or rather, smaller than 1, then the opposite effect is felt: we may underestimate the model's badness.

$$d_0(x,\hat{x}) = MSE = \|x,\hat{x}\|_2^2 \quad (2)$$

In our approach, we use RMSE instead of MSE as the measurement of distortion as in (3). The major advantage of RMSE is that it is at the same scale as the target. In case of noisy data, RMSE is much smaller than MSE, which can prevent overestimating the model's badness, and in case of small error less than 1, RMSE is much larger than MSE, which can prevent underestimating the model's badness.

$$d_1(x,\hat{x}) = RMSE = \sqrt{\|x,\hat{x}\|_2^2} \quad (3)$$

III. Deeper Learned Transformer

The original analysis transform network and synthesis transform network designed by Ballé [3] each contains only 3 convolutional layers, with kernel size of 9 x 9, 5 x5 and 5x5, as shown in Fig. 1, where ↓ denotes down-scaling, ↑ denotes up-scaling, GDN stands for generalized divisive normalization, and IGDN stands for inverse generalized divisive normalization. GDN and IGDN are inspired by models of neurons in biological visual systems, and effective in Gaussianizing image densities [10]. Besides, y denotes the transformed continuous-valued vector, it is then quantized by a uniform scalar quantizer (i.e. each element is rounded to the nearest integer) to generate the quantized vector q, and y_tilde denotes a continuous-valued vector reinterpreted from discrete elements of quantized vector q, as mentioned in (1).

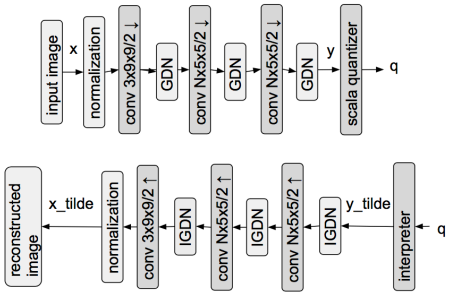

Fig. 1 Ballé's network structure for analysis transform (encoder, on the top) and synthesis transform (decoder, at the bottom) [3].

Inspired by progressive growing generative adversarial network (PGGAN) [9], we modify the network structure of autoencoder-like transformer, to make it deeper. As is shown in Fig. 2, we reduce the kernel size to 3x3, and add three more layers in the encoder, i.e. analysis transform. The benefit of doing this is that the encoder is capable to capture larger spatial features as the network become deeper. However, there is also minor disadvantage for doing so, which is the information loss during down-scaling, which may cause it harder to preserve enough information before entropy coding. The decoder part, i.e. synthesis transform, has a similar structure as the encoder, but with opposite directions.

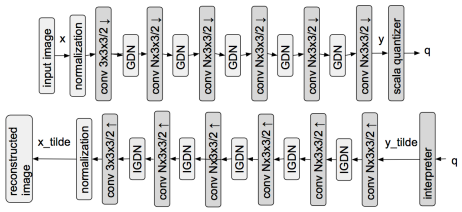

Fig. 2 Our deeper network structure for analysis transform (encoder, on the top) and synthesis transform (decoder, at the bottom).

However, single layer of convolution does not provide enough trainable parameters and structures to learn complex feature representations before down-scaling. That is why PGGAN uses two fully connected convolutional layers before each down-scaling step. Therefore, we doubled the convolutional layers before each GDN and IGDN in our structure, as in Fig. 3, which allows the network to learn more features before losing information during down-scaling.

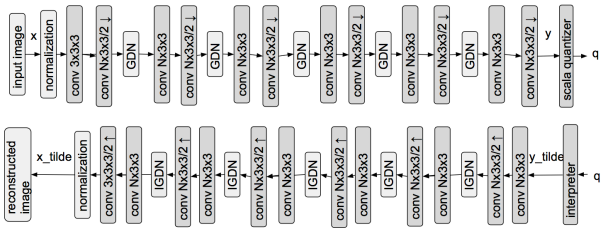

Fig. 3 Our deeper and doubled network structure for analysis transform (encoder, on the top) and synthesis transform (decoder, at the bottom).

As PGGAN outperforms previous GAN approaches in generative tasks, we expect such deeper network structure would also boost the performance of autoencoder, which was supported by experimental results.

Although not directly compared in this paper, the experimental results of the deeper network design in Fig. 2 and Fig. 3 were better than that of the previous shallower network structure in Fig. 1, which can be concluded from the improvements of our approach over BPG (RGB 4:4:4), which was not achieved by Ballé's approach [3].

IV. Experiments

Our modifications and experiments are based on a Python implementation [11] of Ballé's work in 2017 [3]. The performance of the original implementation was slightly better than JPEG 2000, but significantly worse than BPG (RGB 4:4:4), which can also be inferred from the original paper [3], so we didn't include it in our experimental results in this paper.

We trained the autoencoder with 139 high resolution sRGB images of natural scenes from CLIC [12] professional training dataset. Each image was cropped to 1024x1024. For each iteration the model was trained with a batch of 8 randomly cropped 256x256 patches truncated to RGB format. For quick comparison between different model configurations, the early stage models were trained for only 10K iterations, and there had been hundreds of such quick and small experiments. Then several outstanding models were selected, and continued to be trained to 50K iterations, and came to second round of selection. The third round was 250K iterations, and the final stage models were trained to 1M iterations.

Our approaches were evaluated using Kodak True Color Image Suite [13], for the average of PSNR, MS-SSIM, and bpp on all 24 images, in comparison with BPG (RGB 4:4:4). The results showed that our proposed approach with improved optimization goal as described in (3) and deeper network structures as described in Fig. 3 has achieved better performance than BPG (RGB 4:4:4) in terms of subjective fidelity in sharpness and details.

Fig. 4 shows one of the original lossless pictures from Kodak True Color Image Suite, and we can see clear details of the trees on the top right corner and the wall tiles on the bottom right corner, especially the transition of shadows between trees (marked with a blue box), and the sharp edges of each wall tile (marked with a blue box), which was compared to the compressed pictures later.

Fig. 4 Original picture. Blue box areas are used to compare compression results of BPG (RGB 4:4:4) and that of our approach.

Fig. 5 shows the picture compressed by BGP (RGB 4:4:4), and we can see significant blurred effect on the trees and wall tiles when comparing with the original picture. The shadows of the trees become flat blocks (marked with a red box), and

Fig. 5 Picture compressed with BPG (RGB 4:4:4), QP = 42, compression level = 9, bpp = 0.835. Red box areas look blurred.

the edges of the wall tiles almost disappear (marked with a red box). However, the bpp of this compressed picture is 0.835, relatively higher than that of our approach which is shown next.

Fig. 6 shows the picture compressed by our approach, where the details of the trees and wall tiles (marked with green boxes) are well preserved when comparing with the original picture, and is much sharper and clearer than that of the picture compressed by BPG (RGB 4:4:4), even though the bpp is at only 0.561, significantly lower than that of BPG (RGB 4:4:4).

Fig. 6 Picture compressed with our approach, bpp = 0.561. Green box areas are still very sharp.

The major reason for such significant improvement in sharpness and detail fidelity is that our approach provides a non-linear transformer implemented as deep neural network which is better than the linear integer transformer used in BPG in terms of transforming from intensity domain to frequency domain.

In plain areas, such as pure color walls, we did not see much differences regarding color or luminance between the original picture, the picture compressed by BPG (RGB 4:4:4) and that by our approach. Both BPG (RGB 4:4:4) and our approach performed very well on these plain areas.

Regarding objective quality, we have compared our approach with BPG (RGB 4:4:4) on PSNR and MS-SSIM at comparable bpp, and our approach is superior in both metrics. From Fig. 7 we can see that the PSNR of our approach (red line) is consistently higher than that of BPG (RGB 4:4:4) by around 1%, when bpp raised from 0.1 to 0.4. Although PSNR under 30 is still not considered good quality, it is acceptable for the compression ratio between 60 and 240.

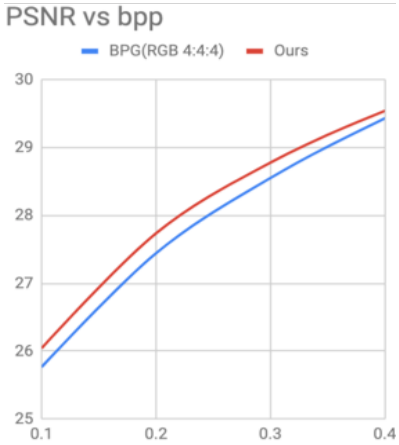

Fig. 7 PSNR (dB) vs bpp on Kodak dataset. The vertical axis represents average PSNR over 24 Kodak images, the horizontal axis represents average bpp (bit per pixel) over 24 Kodak images.

The most impressive improvement is on MS-SSIM, which we can see from Fig. 8 that the MS-SSIM of our approach (red line) is consistently higher than that of BPG (RGB 4:4:4) by around 23% at the same bpp. This means our approach can preserve significantly more structural information than BPG (RGB 4:4:4) with the same compression ratio.

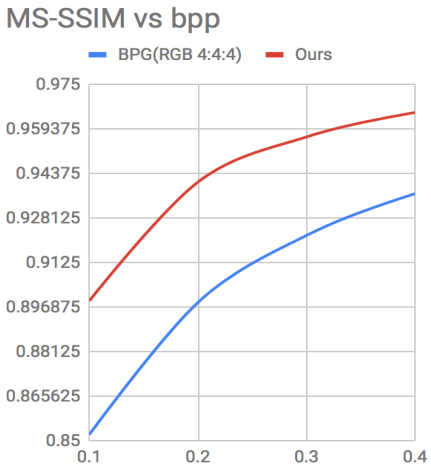

Fig. 8 MS-SSIM vs bpp on Kodak dataset. The vertical axis represents average rescaled MS-SSIM over 24 Kodak images, and the horizontal axis represents average bpp (bit per pixel) over 24 Kodak images.

V. Conclusions

We proposed an improved variational autoencoder with better optimization goal and deeper learned transformer, that outperforms BPG (RGB 4:4:4) in both PSNR and MS-SSIM at comparable bpp. With the same compression ratio, our approach achieved around 1% improvement on PSNR and 23% improvement on MS-SSIM over BPG (RGB 4:4:4), which was not achieved by Ballé [3].

Future work includes improvement with network structure, optimization on encoding and decoding time, and learned compression in the YCbCr color domain. This end-to-end image compression approach may also be applied to intra frame coding in video compression.

Acknowledgments

We acknowledge the support and facilitation from the Dept. of Computer Science and Engineering, Santa Clara University, and the support from Ping An Artificial Intelligence Institute.

References

[1] Descampe Antonin. OpenJPEG 2.3.0. https://www.openjpeg.org/.
[2] Fabrice Bellard. BPG Image format. https://bellard.org/bpg/.
[3] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. End-to-end optimized image compression. In *International Conference on Learning Representations (ICLR),* April 2017. https://arxiv.org/abs/1611.01704.
[4] David Minnen, Johannes Ballé, and George D. Toderici. Joint autoregressive and hierarchical priors for learned image compression. *Advances in Neural Information Processing Systems*, pages 10794–10803, 2018.
[5] Jooyoung Lee, Seunghyun Cho, and Seung-Kwon Beack. Context-adaptive Entropy Model for End-to-end Optimized Image Compression. In *International Conference on Learning Representations (ICLR),* 2019. https://openreview.net/forum?id=HyxKIiAqYQ.
[6] Eirikur Agustsson, Michael Tschannen, Fabian Mentzer, Radu Timofte, and Luc Van Gool. Genrative adversarial networks for learned image compression, 2018. arXiv:1804.02958v2.
[7] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) workshops,* volume 1, page 4, July 2017.
[8] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *The International Conference on Learning Representations (ICLR),* April 2018. https://openreview.net/forum?id=rkcQFMZRb.
[9] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *International Conference on Learning Representations (ICLR),* May 2018. https://openreview.net/forum?id=Hk99zCeAb.
[10] Balle´, Johannes, Valero Laparra, and Eero P. Simoncelli (2015). "Density Modeling of Images Using a Generalized Normalization Transformation". In: *arXiv e-prints*. Presented at the 4th Int. Conf. for Learning Representations, 2016. arXiv: 1511.06281.
[11] Johannes Ballé, Sung Jin Hwang, and Nick Johnston. Python implementation of End-to-end optimized image compression, May 2018. Accessed 2018-11-20. https://github.com/tensorflow/compression.
[12] CLIC "Workshop and challenge on learned image compression". https://www.compression.cc.
[13] Kodak lossless true color image suite. Accessed 2018-12-1, http://r0k.us/graphics/kodak.