

---

# IMPROVED HYBRID LAYERED IMAGE COMPRESSION USING DEEP LEARNING AND TRADITIONAL CODECS

---

A PREPRINT

**Haisheng Fu**

School of Microelectronics  
Xi'an Jiaotong University

fhs4118005070@stu.xjtu.edu.cn

**Feng Liang**

School of Microelectronics  
Xi'an Jiaotong University  
fengliang@xjtu.edu.cn

**Bo Lei**

School of Microelectronics  
Xi'an Jiaotong University  
lei\_bo@stu.xjtu.edu.cn

**Nai Bian**

School of Microelectronics  
Xi'an Jiaotong University  
biannai120@stu.xjtu.edu.cn

**Qian zhang**

School of Microelectronics  
Xi'an Jiaotong University  
zhangqian@stu.xjtu.edu.cn

**Mohammad Akbari**

School of Engineering Science  
Simon Fraser University akbari@sfu.ca

**Jie Liang**

School of Engineering Science  
Simon Fraser University jiel@sfu.ca

**Chengjie Tu**

Tencent Technologies  
chengjietu@tencent.com

July 16, 2019

## ABSTRACT

Recently deep learning-based methods have been applied in image compression and achieved many promising results. In this paper, we propose an improved hybrid layered image compression framework by combining deep learning and the traditional image codecs. At the encoder, we first use a convolutional neural network (CNN) to obtain a compact representation of the input image, which is losslessly encoded by the FLIF codec as the base layer of the bit stream. A coarse reconstruction of the input is obtained by another CNN from the reconstructed compact representation. The residual between the input and the coarse reconstruction is then obtained and encoded by the H.265/HEVC-based BPG codec as the enhancement layer of the bit stream. Experimental results using the Kodak and Tecnick datasets show that the proposed scheme outperforms the state-of-the-art deep learning-based layered coding scheme and traditional codecs including BPG in both PSNR and MS-SSIM metrics across a wide range of bit rates, when the images are coded in the RGB444 domain.

**Keywords** deep learning-based image coding · layered image coding · residual coding · convolutional neural network · autoencoder

## 1 Introduction

Recently with the rapid development of deep learning theory, especially after the successful applications of convolutional neural networks (CNN) in computer vision, deep learning has been applied to many areas, including image compression. Some deep learning-based methods have outperformed traditional image codecs such as JPEG, JPEG2000, and the H.265/HEVC-based BPG image codec [13, 1, 2, 6, 8, 11, 12, 21], demonstrating its great potentials. In [17], Toderici et al. proposed a variable-rate image compression framework for thumbnail images by using the recurrent neural



Figure 1: An example of the decoded images of the BPG, DSSLIC, and the proposed method (bits/pixel/channel, PSNR(dB), MS-SSIM).

networks (RNNs). The scheme was generalized in [12] to compress images of any size and achieve better results than JPEG. The convolutional autoencoder (CAE) framework with a coding network and a decoding network was used for image compression in [7]. In [3], Belle et al. introduced a coding scheme with a generalized divisive normalization (GDN)-based nonlinear analysis transform, a uniform quantizer and a nonlinear synthesis transform.

The presence of quantization, which is non-differentiable, in the pipeline of image coding has posed some challenges to deep learning-based schemes, which use gradient-based backpropagation to train neural networks. To solve the problem, some approximations of the quantization have been proposed. For example, in [9], the quantization of the autoencoder is replaced by a smooth approximation in the backpropagation because the derivative of the rounding function is zero almost everywhere, but the forward pass of the backpropagation still use the quantization to prevent the decoder from learning to invert the smooth approximation. A soft (continuous) relaxation of quantization and entropy was developed in [1]. The designed network is not used to give a specific quantized output, but is used to output a quantization level along with the weight.

The generative adversarial network (GAN) [22] is a powerful and versatile scheme, and has also been adopted in image compression. In [11], a decoder was trained to generate realistic images using a discriminator. The similarity is measured by perceptual loss based on the feature map of a pretrained AlexNet. In [18, 2], the feature maps were extracted from learning-based image compression to complete image segmentation or image classification. Based on the work in [11, 18, 2], the work [2] used segmentation map-based image synthesis to train the discriminator of GAN, which used synthesized images for non-important regions and obtained very good performance at low bite rates (<0.1 bits/pixel).

In [10], a deep semantic segmentation-based layered image compression (DSSLIC) scheme is proposed, which is a hybrid coding approach that uses both deep learning and the traditional codecs such as the BPG and FLIF [4]. It has three layers. First, it obtains the semantic map of the original image and encodes it by the FLIF codec as the base layer, which is used to synthesize an initial reconstruction. To improve the image synthesis quality, a low-dimensional representation of the input is generated and coded by FLIF as the second layer, which is served as a side information to the image synthesis. Moreover, the residual between the input and the synthesized image is encoded by the BPG codec as the third layer. The DSSLIC outperforms the BPG codec (in RGB444 format) by large margin in both PSNR and multi-scale structural similarity (MS-SSIM) metrics across a large range of bit rates.

Although the performance of DSSLIC is quite impressive, we observe that in many cases the semantic information in DSSLIC is not very useful. For example, when the semantic segmentation part fails to assign any label to a region, the image synthesis part is actually not used at all, and the remaining parts of the DSSLIC still works very well. This motivates us to develop a simplified hybrid coding scheme without using semantic segmentation-based image synthesis.

In this paper, we propose an improved hybrid image compression framework. Our scheme only has two layers. Its base layer is essentially an autoencoder, and the result is coded by the FLIF codec. The enhancement layer is the residual between the input and the reconstruction of the autoencoder, which is coded by the BPG codec. A number of

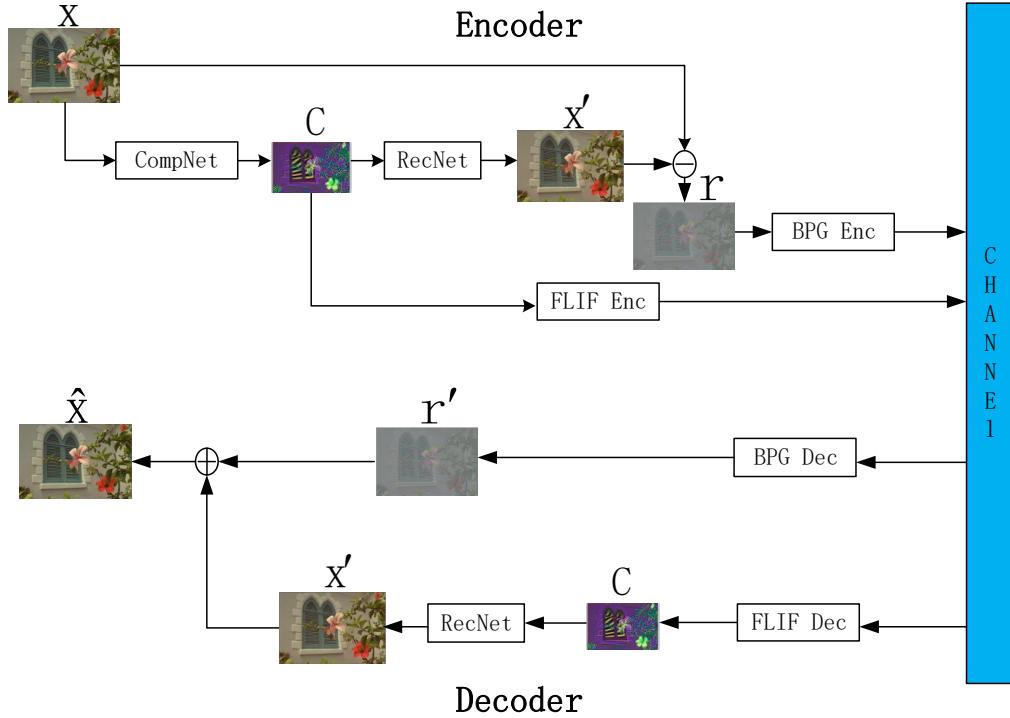


Figure 2: The Framework of the proposed improved hybrid image compression network.

contributions are made in the design of the autoencoder. We change the structure of the original residual block in the ResNet in [14] and add the dropout module between two convolution layers to improve the generalization capability of the network. We also replace the default rectified linear units (ReLU) by the parametric rectified linear units (PReLU). Despite the simplified scheme, our method can outperform the DSSLIC and other traditional codecs in both PSNR and MS-SSIM metrics across large range of bit rates, especially in PSNR. An example is shown in Fig. 1 by comparing with the DSSLIC and BPG. More results are reported in Sec. 4.

The organization of the paper is as follows. The framework and the details of the improved hybrid image compression method are introduced in Sec. 2. In Sec. 3, we evaluate the performance of our method by comparing with DSSLIC, BPG, JPEG, JPEG2000 and WebP codecs using the popular Kodak and Tecnick datasets in the RGB444 domain. Conclusions and discussions are given in Sec. 4.

## 2 The Proposed Hybrid Image Compression Scheme

### 2.1 Framework

The proposed framework is described in Fig. 2. It includes two layers, which is simpler than the DSSLIC. We first use a deep learning network named CompNet to learn the compact image  $c$  of the original image  $x$ , which is losslessly encoded by the FLIF codec. Next, the compact image is used by another deep learning network named RecNet to produce a coarse reconstruction  $x'$  of the input image. CompNet and RecNet thus form a convolutional autoencoder (CAE). After that, the residual  $r$  between the input image and the coarse reconstruction image is obtained and encoded by the lossy BPG codec. In the decoder side, the compact image  $c$  is decoded, which is used by the RecNet to obtain the coarse reconstruction  $x'$ . The residual image is decoded by the BPG decoder, denoted by  $r'$ . Finally, the coarse reconstruction  $x'$  and the decoded residual image  $r'$  are added to get the final reconstruction  $\hat{x}$ .

### 2.2 Network Architecture

The network structures of the CompNet and RecNet are described in Fig. 3. The two networks constitute a complete CAE. Both of them have 13 network layers. The CompNet contains four downsampling layers and the RecNet contains four upsampling layers. There is a residual network which contains two convolution layers between the two upsampling

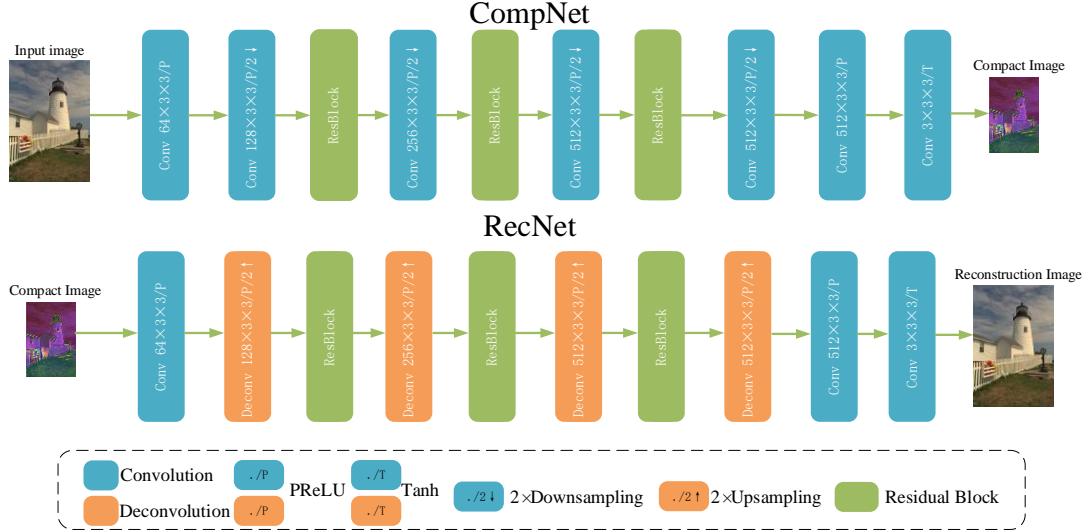


Figure 3: The structures of the CompNet and RecNet.

or downsampling layers. The initial image size is  $W \times H \times 3$ . All the kernel sizes of the convolutional layers and deconvolutional layers are  $3 \times 3$ . The width and the height of the compact image  $c$  are both  $1/16$  of those of the input image.

In order to improve the performance of image compression, we add a residual block between every two upsampling layers and downsampling layers, motivated by the skip connection in the ResNet [14], which can deepen our network and also achieve faster convergence. The work [3, 5] introduce the generalized divisive normalization (GDN) and inverse GDN (IGDN) modules, based on a local divisive normalization transformation. Compared with the residual blocks, the networks that rely on GDN/IGDN modules may have vanishing gradient and exploding gradient problems during training as the increase of the number of network layers. Since our goals are to obtain compact images of smaller size and reconstructed images of higher quality, which call for deeper networks. Therefore we choose to use the residual block as the basic module of our networks.

In the residual block, we use two  $3 \times 3$  convolution layers with  $k$  filters. Inspired by [15], we also make some changes to the structure of the residual block. We remove the nonlinear mapping after the skip connection for each residual unit and add the dropout module between the two  $3 \times 3$  convolution layers. Experimental results show that this can speed up the convergence during training, and can prevent over-fitting as well. The detailed description is shown in Fig. 4.

In addition, inspired by [15, 23], we replace the rectified linear units (ReLU) by the parametric rectified linear units (PReLU), which can achieve a faster convergence speed than ReLU [15]. In the experimental section, we will investigate the effects of these two activation functions on image compression performance.

Moreover, in the last layers of CompNet and RecNet, we use the Tanh activation function to make the output range distributed in  $[-1, 1]$ .

### 2.3 Scaling in the System

In our system, the data at different stages have different ranges. The pixel values of the input image are 8-bit unsigned integers in the range of  $[0, 255]$ . First, the inputs to the CompNet should be normalized to the range of  $[-1, 1]$  as follows.

$$x_f = \frac{x}{255} \times 2 - 1. \quad (1)$$

The output of CompNet is also in  $[-1, 1]$ . In order to encode the output of the CompNet by the FLIF codec, we scale it to 8-bit integers in  $[0, 255]$  as follows, so that it can be accepted by FLIF as a valid image.

$$x_Q = \text{round}\left[\frac{x_f + 1}{2} \times 255\right]. \quad (2)$$

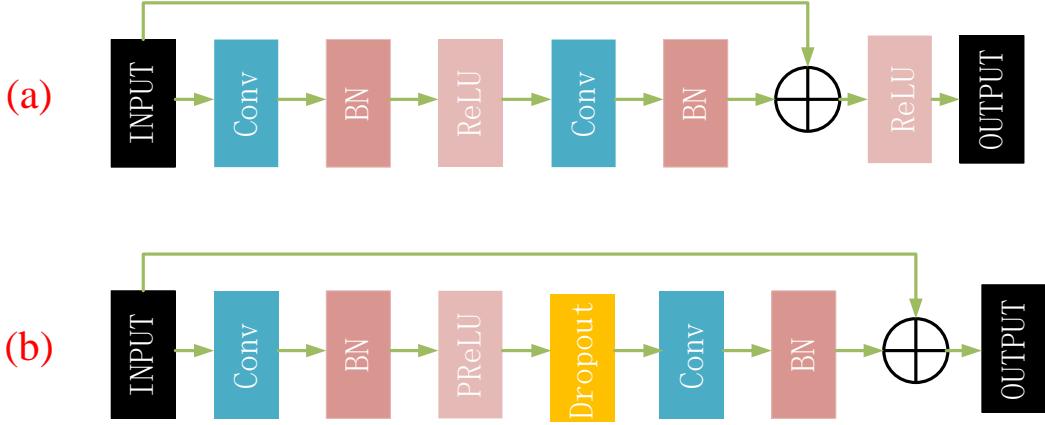


Figure 4: (a) The original residual block in [14]. (b) The proposed residual block.

The scaled result  $x_Q$  is then normalized to  $[-1, 1]$  by Eq. 1, and is sent as the input to the ResNet. The output of ResNet is also in the range of  $[-1, 1]$ , and is converted to 8-bit integers using Eq. 2. This is the coarse reconstruction of the image.

The residual image  $r$  is the difference between the original image and the coarse reconstruction. Therefore its range is  $[-255, 255]$ . Since we use the traditional BPG codec to encode the residual, another scaling function is needed to map the residual image pixels from  $[-255, 255]$  to  $[0, 255]$ , so that the residual image can be accepted by the BPG codec as a valid input image.

In [10] and its early version, two mapping methods are used. The first one is the following simple shift method:

$$x' \leftarrow \text{round}\left[\frac{x' + 255}{2}\right]. \quad (3)$$

A better scaling method is to find the maximum and minimum values of the residual images,  $x'_{\max}$  and  $x'_{\min}$ , and then adaptively map the range between them to  $[0, 255]$ . That is:

$$x' \leftarrow \text{round}\left[\frac{(x' - x'_{\min})}{(x'_{\max} - x'_{\min})} \times 255\right]. \quad (4)$$

In this method, the minimum and maximum values need to be sent to the decoder to do the inverse scaling, but the overhead is negligible. The shift method in Eq. 3 is a special case of this adaptive method when the maximum and minimum values are 255 and -255 respectively.

Our experimental results show that in most cases, the residual image pixel values generally obey a normal distribution with mean around zero and a very small variance. That is, many residuals are very close to 0, and most residuals are within a small range, as shown by the example in Fig. 5, hence the adaptive method can preserve more details of the residual, and yields better performance than the simple shift method.

However, sometimes the maximum and minimum are quite close to 255 and -255, making the adaptive scaling method very similar to the shift method, although most residuals are very small, and there are only a few pixels with extremely large absolute values.

To address this problem, we propose a clipping method. It can be seen from Fig. 5 that most of the values fall in  $[-120, 120]$ . Therefore we set the maximum value to 120 and the minimum value to -120 in Eq. 4. Extreme residual pixels beyond  $[-120, 120]$  are clipped to -120 or 120. This can reduce the range and preserve more details after the scaling. We call this the clipping method. We will show later that the clipping method is slightly better than the adaptive method, and the latter is better than the simple scaling method.

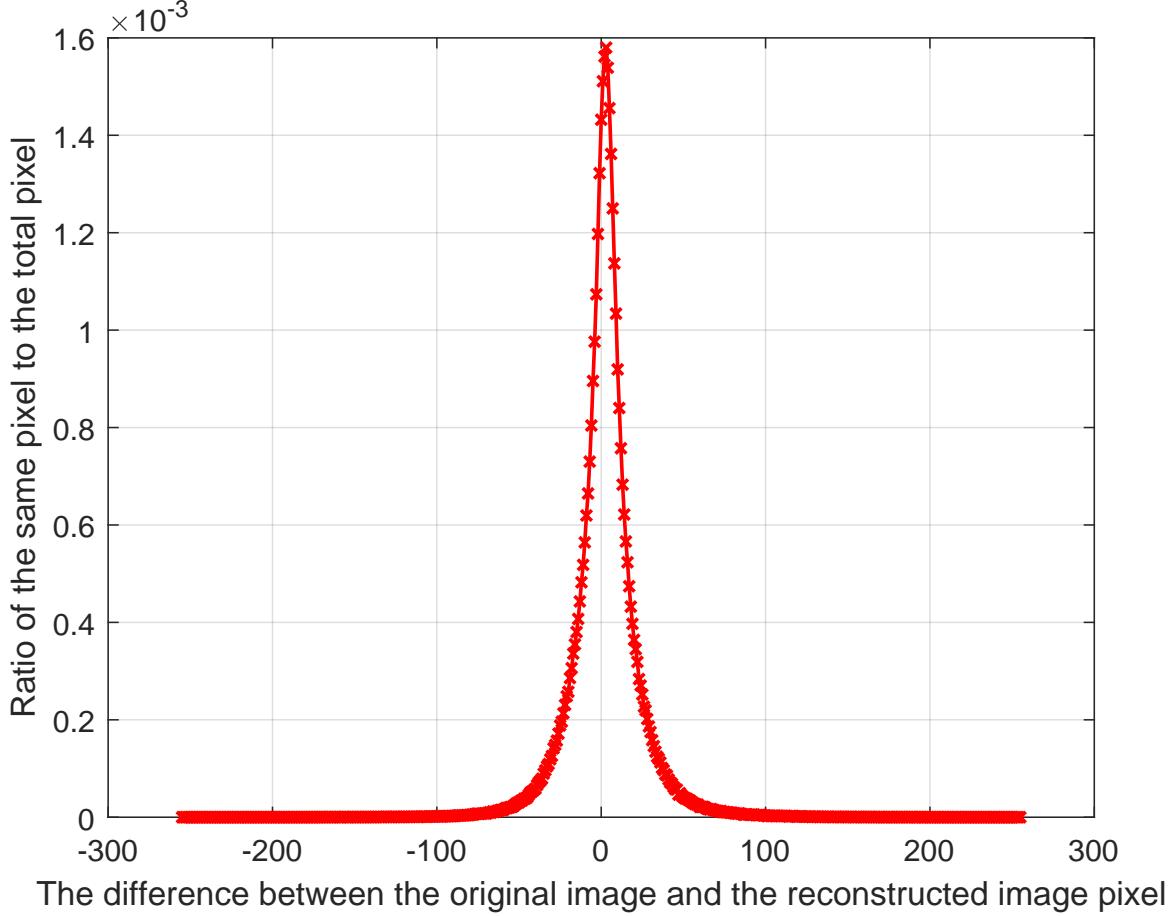


Figure 5: Distribution of the residual image pixels between the original image and the coarse reconstruction.

## 2.4 Network Training

We use the CLIC compression dataset released by the Computer Vision Lab of ETH Zurich to train our network. In order to train our network better, we use several data augmentation methods such as rotation and scaling to randomly extract 81,650 patches from 1633 high-quality images. These images are saved as lossless PNGs to avoid compression artifacts. We only use the mean squared error (MSE) as the loss function to train our network.

$$L = \frac{1}{N} |Y_n - X_n|^2, \quad (5)$$

where  $X_n$  represents the initial input image, and  $Y_n$  represents the coarse reconstruction, which is the output of the RecNet.  $N$  is the batch size. In order to improve the SSIM performance of the reconstructed images, many papers have added some other loss functions such as VGG Loss, Perception Loss, and Adversarial Loss in the training process. Although these loss functions can improve the MS-SSIM metric of the reconstructed images, they will degrade the PSNR of the image. We will show later that although the SSIM is not considered in our cost function, our method still outperforms other methods in SSIM.

The CompNet and RecNet are jointly trained for 40 epochs with mini-batch stochastic gradient descent method (SGD). The batch size is set to 20. The Adam solver with fixed learning rate of 0.0001 is used.

## 3 Experiments

In this section, we evaluate the performance of our method on the Kodak PhotoCD dataset[25] and Tecnick dataset [24], and compare with some other methods, including DSSLIC, BPG, JPEG2000, WebP, and JPEG. All images are coded

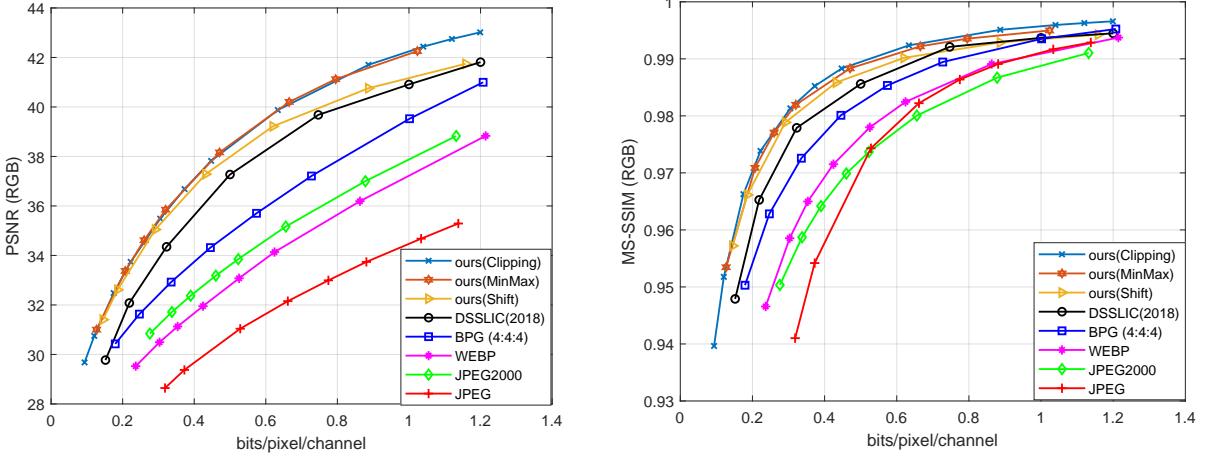


Figure 6: Average comparison results on all 24 Kodak images in terms of PSNR and MS-SSIM.

by the RGB444 format in all codecs. We use both the PSNR and MS-SSIM metrics to compare the performance of different codecs.

In our method, we evaluate three scaling methods of the residual images, namely, the simple shift method in Eq. 3, the MinMax method in Eq. 4, and the Clipping method.

### 3.1 Image Coding Results

The average PSNR performance and MS-SSIM performance over the 24 Kodak images are illustrated in Fig. 6. All three versions of our method achieve better results than other methods in both PSNR and MS-SSIM. The Clipping method and the MinMax method have almost the same performance, and consistently outperform the DSSLIC, with an average gain of over 1 dB. The Shift method has similar performance as the other scaling methods at low rates, but the performance gets worse at high rates. However, it still outperforms the DSSLIC. The DSSLIC outperforms BPG and other codecs by a large margin.

Fig. 7 compares the performance of our method with others on the Tecnick dataset. In PSNR, the Clipping method and the MinMax method still outperform DSSLIC and BPG. The Shift method is similar to DSSLIC, and is worse than BPG at high rates. In the MS-SSIM metric, the clipping and MinMax methods are always better than DSSLIC. They are also better than BPG at low rates. The shift method is slightly worse than DSSLIC.

To compare the visual quality of different methods, we show some examples from the Kodak dataset in Fig. 9 to Fig. 10, with some regions amplified. In the test, the shift scaling method is used in our method. By analyzing all the samples, we can get the following conclusions. Compared to other codecs, the JPEG has the worst performance due to the blocking artifacts. The JPEG2000 shows some ringing artifacts. The WebP codec has better performance than JPEG2000, but it is blurry in certain areas of the image. The BPG result is quite smooth, but it loses some detailed information. DSSLIC is significantly better than BPG, and our method achieves even better result than DSSLIC.

### 3.2 Other Ablation Studies

We also conduct ablation studies to investigate the impacts of the size of the compact representation and the activation functions in our framework. The result is shown in Fig. 8. It can be seen that the best results are given when the width and height of the compact image  $c$  are both  $1/16$  of those of the original image, compared to  $1/8$  and  $1/32$ . In this case, PReLU yields slightly better performance than ReLU. Therefore the configuration of S16+PreLU is selected in our experimental results above. In addition, the PSNR is more susceptible to the compact map size and activation function, compared to the SSIM.

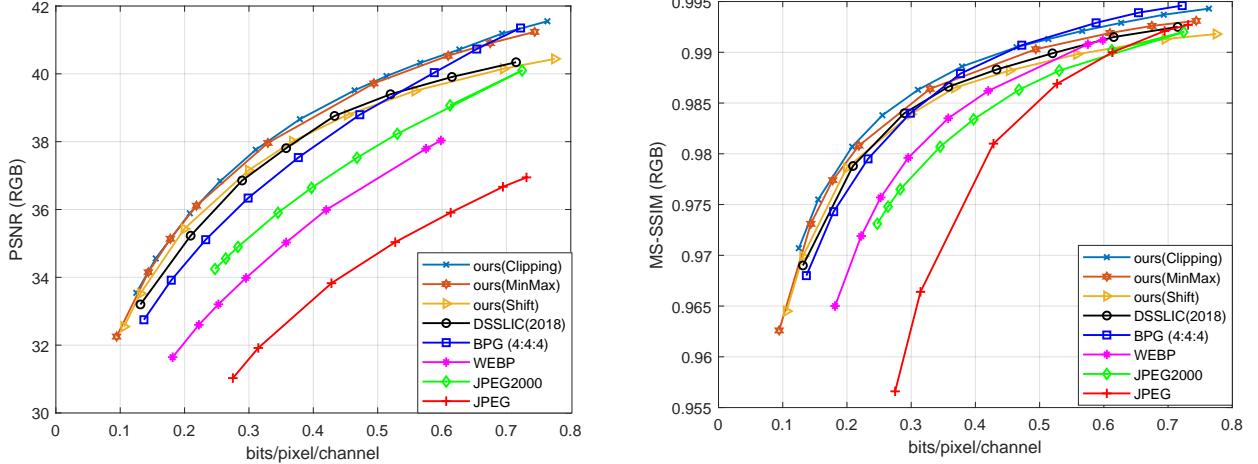


Figure 7: Average comparison results on all 100 Tecnick images in terms of PSNR and MS-SSIM.

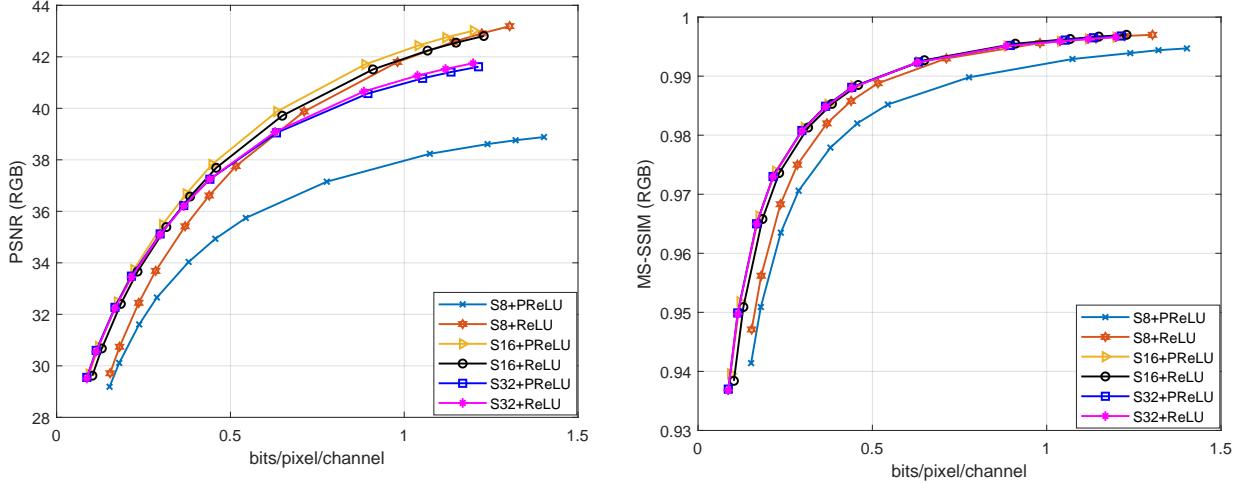


Figure 8: The effect of different layers and activation functions on the performance of the framework on all 24 Kodak images in terms of PSNR and MS-SSIM. **Sn:** the width and the height of the compact image  $c$  are both  $1/n$  of those of the input image. **PReLU or ReLU:** the activation functions of the network.

## 4 Conclusion

In this paper, we propose an improved hybrid image compression framework, which is a simplified version of the state-of-the-art DSSLIC scheme in [10] without using the semantic segmentation layer. Various modifications to the autoencoder network are proposed, and a new clipping method is proposed to scale the residual image. The performance of our method in both PSNR and MS-SSIM metrics is better than BPG codec and DSSLIC across a large range of bit rates. In the future work, we will further improve the performance of the image on the MS-SSIM metric, and we will investigate how to encode the residual image more efficiently.

## Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grant No. 61474093 and by Tencent.

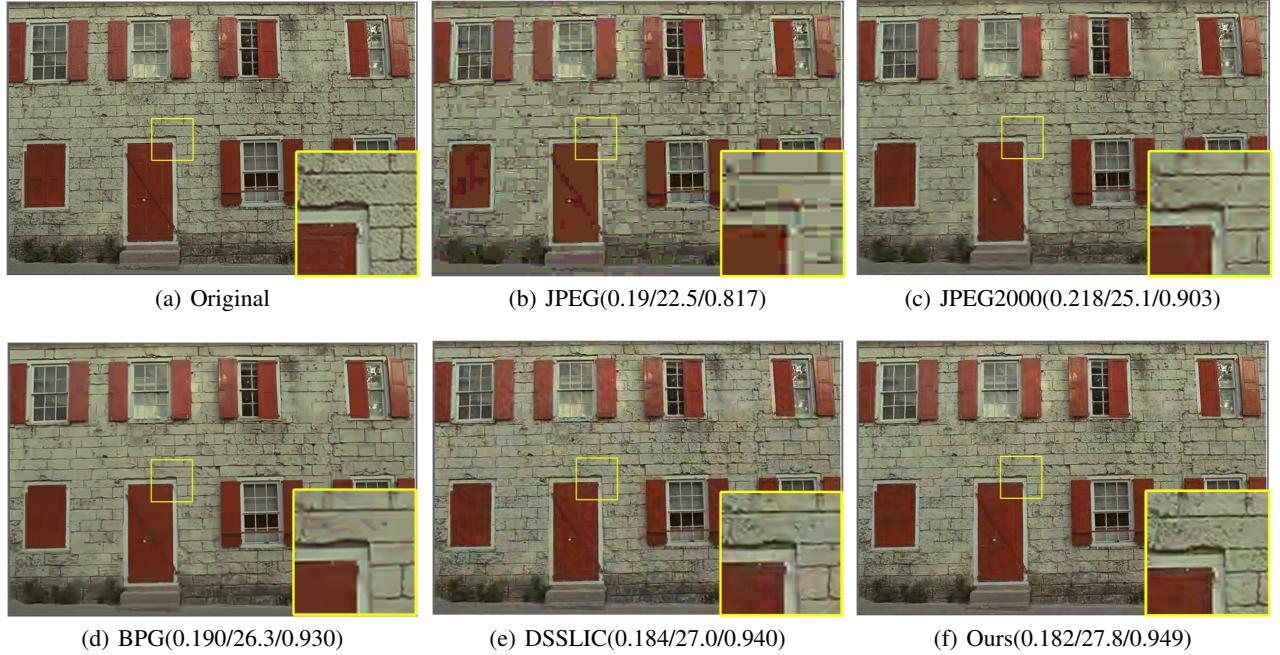


Figure 9: Example 1 in the Kodak dataset (bits/pixel/channel, PSNR, MS-SSIM).

## References

- [1] Eirikur Agustsson, Fabian Mentzer, Michael Tschannen, Lukas Cavigelli, Radu Timofte, Luca Benini, and Luc V Gool. Soft-to-hard vector quantization for end-to-end learning compressible representations. In *Advances in Neural Information Processing Systems*, pages 1141–1151, 2017.
- [2] Eirikur Agustsson, Michael Tschannen, Fabian Mentzer, Radu Timofte, and Luc Van Gool. Generative adversarial networks for extreme learned image compression. *arXiv preprint arXiv:1804.02958*, 2018.
- [3] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. End-to-end Optimized Image Compression. In *International Conference on Learning Representations*, 2017.
- [4] Jon Sneyers and Pieter Wuille. FLIF: Free lossless image format based on MANIAC compression. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 66–70, 2016.
- [5] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang and Nick Johnston. Variational image compression with a scale hyperprior. In *International Conference on Learning Representations*, 2018.
- [6] Mu Li, Wangmeng Zuo, Shuhang Gu, Debin Zhao, and David Zhang. Learning Convolutional Networks for Content-Weighted Image Compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3214–3223, 2018.
- [7] Oren Rippel, and Lubomir Bourdev. Real-time adaptive image compression. In *International Conference on Machine Learning*, pages 2922–2930, 2017.
- [8] Nick Johnston, Damien Vincent, David Minnen, Michele Covell, Saurabh Singh, Troy T. Chinen, Sung Jin Hwang, Joel Shor, George Toderici. Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4385–4393, 2018.
- [9] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. In *International Conference on Learning Representations*, 2017.
- [10] Mohammad Akbari, Jie Liang, and Jingning Han. DSSLIC: Deep Semantic Segmentation-based Layered Image Compression. In *The 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2042–2046, 2019.
- [11] Shibani Santurkar, David Budden, and Nir Shavit. Generative Compression. In *2018 Picture Coding Symposium (PCS)*, pages 258–262, 2018.



Figure 10: Example 2 in the tecnick dataset (bits/pixel/channel, PSNR, MS-SSIM).

- [12] George Toderici, Damien Vincent, Nick Johnston, Sung Jin Hwang, David Minnen, Joel Shor, and Michele Covell. Full resolution image compression with recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5306–5314, 2017.
- [13] Fabrice Bellard. Bpg image format (2017). <http://bellard.org/bpg>, 2016.
- [14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [15] Haojie Liu, Chen Tong, Shen Qiu, Yue Tao, and Ma Zhan. Deep Image Compression via End-to-End Learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition(CVPR)*, 2018.
- [16] Zhou Wang, Eero P. Simoncelli, and Alan C. Bovik. Multiscale structural similarity for image quality assessment. In *The Thirty-Seventh Asilomar Conference on Signals, Systems Computers*, 1398–1402, 2003.
- [17] George Toderici, Sean M. O’Malley, Sung Jin Hwang, Damien Vincent, David Minnen, Shumeet Baluja, Michele Covell, and Rahul Sukthankar. Variable rate image compression with recurrent neural networks. *arXiv preprint arXiv:1511.06085*, 2015.
- [18] Sihui Luo, Yezhou Yang, and Mingli Song. DeepSIC: Deep Semantic Image Compression. *arXiv preprint arXiv:1801.09468*, 2018.
- [19] Robert Torfason, Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, and Luc Van Gool. Towards Image Understanding from Deep Compression without Decoding. *arXiv preprint arXiv:1803.06131*, 2018.

- [20] Eirikur Agustsson, Michael Tschannen, Fabian Mentzer, Radu Timofte, and Luc Van Gool. Extreme learned image compression with gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition(CVPR)*, pages 2587–2590, 2018.
- [21] Eirikur Agustsson, Michael Tschannen, Fabian Mentzer, Radu Timofte, and Luc Van Gool. Extreme learned image compression with gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition(CVPR)*, pages 2587–2590, 2018.
- [22] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville and Yoshua Bengio. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.
- [23] David Minnen, Johannes Ballé, and George Toderici. Joint Autoregressive and Hierarchical Priors for Learned Image Compression. In *Advances in Neural Information Processing Systems*, pages 10771–10780, 2018.
- [24] Tecnick. <https://bellard.org/bpg/>.
- [25] Kodak. <https://r0k.us/graphics/kodak/>.