Image Compression Architecture with Built-in Lightweight Model

Tien-Ying Kuo*, Yu-Jen Wei† and Jhih-Jhou Lin †
Department of Electrical Engineering, National Taipei University of Technology, Taipei, Taiwan, R.O.C
E-mail: * tykuo@ntut.edu.tw
†{t106319012, t107318045} @ntut.org.tw

***Abstract*— Many studies have applied deep learning techniques to image compression in recent years. However, it is difficult to apply these algorithms to all sorts of images because it is necessary to cover a large number of diverse images for training. To tackle this challenge, we propose a new image compression framework based on customized learning. We only let the model analyze a single image and learn the lost information after traditional compression algorithms. Then, we encode the trained model parameters and the compressed image separately and transmit them together. At the decoder, we can restore the uncompressed image content by the model parameters. In our experiments, we use JPEG and JPEG2000 to validate our algorithm, and from the experimental results we can prove that our framework is feasible.**

I. INTRODUCTION

With the advancement of technology, the number and file sizes of digital images are increasing, which makes storage devices and transmission channels unable to afford. The best solution is to compress the image through an image compression algorithm to represent the same image content with a lower number of bits. Existing image compression algorithms can be divided into traditional and deep learning algorithms based on whether they use a Convolutional Neural Network (CNN) or not.

JPEG, JPEG2000 and HEIF are the traditional algorithms that are widely used today. These algorithms are mainly composed of entropy coding, transform coding and predictive coding techniques. JPEG first partitions the image into blocks, then uses the discrete cosine transform (DCT) to transform the information in the block from the spatial domain to the frequency domain, quantizes the information in different degrees according to the frequency, and then uses entropy coding for compression. In order to improve the blocking artifacts of JPEG, JPEG2000 uses wavelet transform instead of the block-based DCT. HEIF is an encoding technique based on HEVC compression to reduce the amount of storage required for images. While traditional algorithms are effective in removing redundant information from images, they also introduce noise such as blocking artifacts, ringing artifacts and blurring artifact.

With the development of deep learning, many researches use CNN technology to develop compression algorithms. There are two development methods based on CNN: designing a new compression algorithm [1-3], or combining with the traditional compression algorithm [4-7].The first method is to use CNN to design the encoder and decoder of the whole compression algorithm to achieve high compression efficiency. Another method is to use CNN to complement the traditional compression algorithm to improve the quality of the compressed image. The problem of deep learning algorithms is needing a large and diverse set of training images so that the algorithm can maintain high performance across different types of image content. In addition, the existing work often uses complex and deep networks to improve the generalizability of models, but this also brings huge computational complexity and number of parameters, which makes real-time applications very difficult, especially in the lightweight decoder. It is also important to avoid models to fit to specific datasets for the existing CNN approaches.

In this paper, we propose an image compression architecture combining traditional algorithms and CNN to solve the above problems. The input image is first compressed by traditional algorithms, the compressed image is used to train the model, and then the compressed image and the model parameters are stored after the training is completed, where ground truth is the original image, which enables the output of content close to the original image after decoding on the decoding side. Since the number of parameters in the model affects the performance of the compression, we design a lightweight model for the purpose of bit reduction. The contribution of this paper is as follows:

- We propose an innovative image compression algorithm that stores the model parameters carrying the detailed image information together with the compressed image to enhance the viewing quality of the decoded image.
- Unlike current deep learning compression methods, we let the model precisely match a single compressed image, rather than pursuing generalizability of the model.
- We use JPEG and JPEG2000 to test our proposed image compression architecture, and the experimental results prove that our algorithm can achieve better image quality.

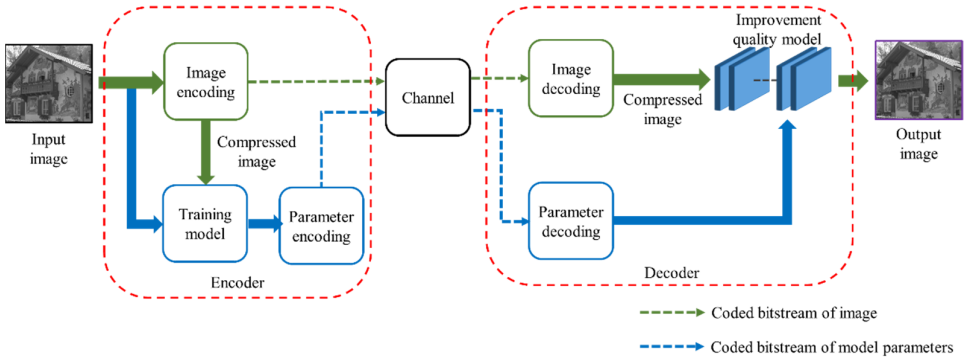


Fig. 1 Flowchart of the proposed architecture

II. Related Work

A. Compression algorithm based on CNN

Toderici [1] designed the model architecture based on the concept of auto-encoder, adopted the binary neural network to quantize the representation code, and added the architecture of recurrent neural network in the decoder to improve the image quality after decoding. Johnston [2] improved the architecture of Toderici [1] by adding rate distortion optimization to improve the compression efficiency. Agussson [3] combined the architecture of auto-encoder with GAN for training, which can produce high visual quality images even in the case of low bit rate. A bit rate allocation architecture was also proposed to select the parts that need to be reserved according to semantic segmentation.

B. Improving tradition compression algorithm using CNN

Jiang [4] proposed ComCNN and RecCNN at the encoder and decoder to reduce the burden of image transmission and storage. ComCNN is used to extract the representative components of the image, then output the compact representation image, and then encode the image. RecCNN is responsible for restoring the decoded image to the original image. BlockCNN [5] combines the concept of PixelCNN with JPEG image compression standard, uses the surrounding coded blocks to predict the content of the current block, then calculates the residual between the predicted result and the original content, and encodes and stores the residual with the compressed content. DnCNN [6] uses a 20-layer network to repair the same type of distorted image but with different degrees of disortion, and repairs the distorted image by predicting the residual information. [7] used recursion to simulate the effect of deep network with fewer convolution layers, and used the dilated convolution to increase the effective receptive field of the model.

III. Proposed Method

Our proposed framework for image compression is shown in Fig. 1. Since the human eye is more sensitive to the changes in luminance, our compression framework is constructed for the luminance component of the image. At the encoding side, the input image is first compressed and converted to a bit-stream, and then the bit-stream is decoded to obtain the content of the compressed image. Then we partition the input image and the compressed image into block pairs and use each block pairs to train the model. Finally, we convert all the trained parameters to bit-stream and transmit them to the decoder together with the bit-stream of the compressed image. In the decoder, the bitstream is decoded and converted to the compressed image and model parameters, and then the corresponding model parameters are applied to each area of the compressed image, resulting in a high quality output image.

A. Design Model

Since our compression framework has to store the model parameters together with compressed image, the number of model parameters could affect the compression performance. Thus, it is necessary to design a restoration model with low number of parameters. The model designed in this work is shown in Fig. 2. Throughout the network, we use a residual architecture to predict the difference between the input image and the ground truth. Compared to directly predicting the whole image, the predicted residual component only needs to generate the difference from the original image, so using only a shallow network can give a closer result to the ground truth.

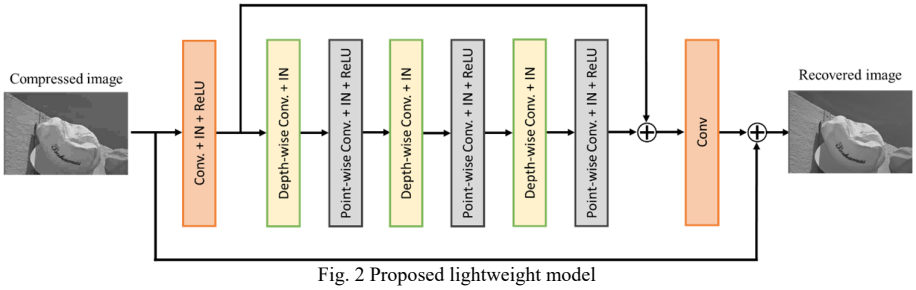
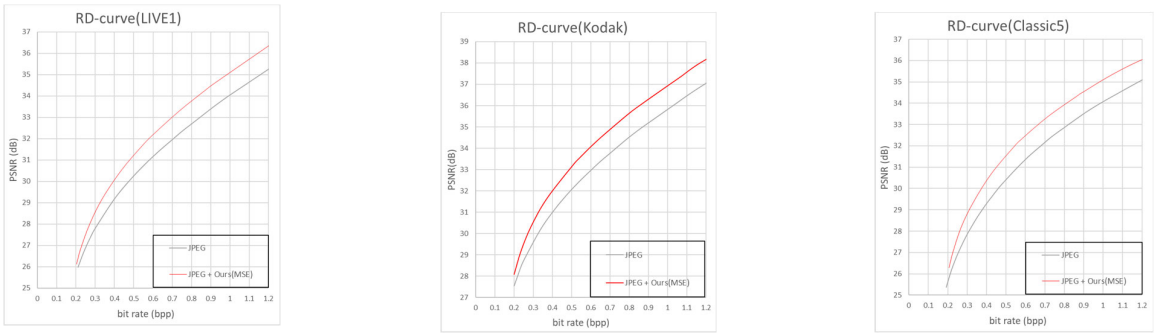
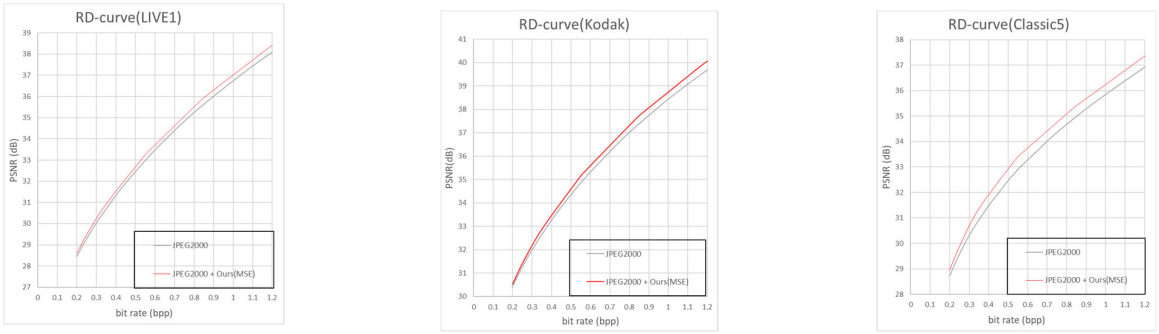


Fig. 2 Proposed lightweight model



(a) Our method combined with JPEG



(b) Our method combined with JPEG2000

Fig. 3 Results of RD-curve on different databases

We use a standard convolutional layer for the first and last layer of the model structure and a depthwise separable convolution for the rest of the model structure. The size of the convolution kernel for all layers is 3 x 3 pixels. Compared to using standard convolution layers, the depthwise separable convolution can effectively reduce the number of model parameters while maintaining similar model performance. We add instance normalization to all the convolution layers except the last one to improve the training speed and stability of the model.

B. Training

In order to allow the model to be quickly converged for practical applications, we pre-trained the model. In the model pre-training process, we used BSDS500 and DIV2K as the pre-training dataset and generated compressed images with different compression levels using JPEG and JPEG2000, and cropped the images to 256×256 size during training. The model training results are susceptible to the initialization of the weights, so we choose the MSRA initialization proposed by He [8]. We use SGDM as the optimizer and set the initial learning rate to 0.01. The loss functions are based on MSE as shown in (1).

Next, we describe the setup of the compression algorithm in the practical application. We use the pre-trained parameters for the initialization of the parameters. The compressed image is used as the input to the model, and the original image is used as the ground truth. Since Adam is better than SGDM in making the model to converge quickly, we adopt Adam as the optimizer and set the initial learning rate to 0.1. To avoid unstable network training due to gradient exploding, we use clipping gradient to avoid excessive parameter updates. The loss function used here is the same as in pre-training.

$$Loss_{MSE} = \frac{1}{m}\sum_{i=1}^{m}\left\|I_{gt_i} - I_{rec_i}\right\|_2^2 \quad (1)$$

IV. Experiment Result

We use a PC with an Intel i7-7700K CPU running at 3.0GHz and an NVIDIA 2080Ti GPU as our test environment. Table I shows the number of parameters in our model, the computational complexity of our model, and the time required to decode an image with an image size of 256 × 256. It is worth noting that the number of parameters and complexity of our model are relatively low compared to existing models, and it takes less than 0.5s to process an image using the CPU, which proves that this architecture has a low hardware requirement.

Table I Details of our model

# of Parameters	378
GFLOPs	0.14
Run time on CPU (s)	0.43
Run time on GPU (s)	0.015

We use LIVE1, Kodak and Classic5 as test sets as well as BD-rate alike and BD-PSNR alike as evaluation criteria, which is modified for images from videos to evaluate the averaging RD-curve performance for compression algorithms. We choose JPEG and JPEG2000, two common lossy compression algorithms, as the image encoders in our compression framework.

Fig. 4 Results of different compression methods

Table II show the results of our method. We compare the results of combining JPEG and JPEG2000 with our method respectively. Fig. 3 is the RD-curve on all test datasets. Our method reduces average rate by 16.18% and 4.77%, and improves the average PSNR by 0.952dB and 0.254dB, respectively, compared to the original algorithm. It can be seen from Fig. 4 that our method is not only better in numerical data analysis, but also outperforms in terms of image visual quality.

Table II The results of our method on different datasets

	JPEG		**JPEG2000**	
Dataset	Rate (Δ%)	PSNR (ΔdB)	Rate (Δ%)	PSNR (ΔdB)
LIVE1	-15.27	0.906	-4.12	0.224
Kodak	-17.05	0.998	-4.76	0.256
Classic5	-17.31	1.000	-8.61	0.419
Average	-16.18	0.952	-4.77	0.254

V. Conclusions

We combine the traditional image compression algorithm and deep learning technology to propose a novel image compression architecture. Compared to other work using deep learning, we let the model learn the information lost when a single image is compressed, and encode and store the model parameters with the compressed image to aid in image decoding to restore high quality images. We tested three image databases, LIVE1, Classic5 and Kodak. The experimental results show that our architecture can provide better compression efficiency.

Acknowledgment

This work was supported by Ministry of Science and Technology (grant # MOST 109-2221-E-027- 088-)

References

[1] G. Toderici et al., "Full resolution image compression with recurrent neural networks," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5306-5314.

[2] N. Johnston et al., "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, 2018, pp. 4385-4393.

[3] E. Agustsson, M. Tschannen, F. Mentzer, R. Timofte, and L. V. Gool, "Generative adversarial networks for extreme learned image compression," in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 221-231.

[4] F. Jiang, W. Tao, S. Liu, J. Ren, X. Guo, and D. Zhao, "An end-to-end compression framework based on convolutional neural networks," IEEE Transactions on Circuits and Systems for Video Technology, vol. 28, no. 10, pp. 3007-3018, 2017.

[5] D. Maleki, S. Nadalian, M. Mahdi Derakhshani, and M. Amin Sadeghi, "Blockcnn: A deep network for artifact removal and image compression," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2018, pp. 2555-2558.

[6] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," IEEE transactions on image processing, vol. 26, no. 7, pp. 3142-3155, 2017.

[7] T.-Y. Kuo, Y.-J. Wei, and C.-H. Chao, "Restoration of Compressed Picture Based on Lightweight Convolutional Neural Network," in 2019 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), 2019, pp. 1-2: IEEE.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in Proceedings of the IEEE international conference on computer vision, 2015, pp. 1026-1034.