Learned Image Compression using Adaptive Block-wise Encoding and Reconstruction Network

Zhenghui Zhao[1,2*], Chuanmin Jia[1,3*], Shanshe Wang[1,3], Siwei Ma[1,3], Jiansheng Yang[2]
[1]*Department of Computer Science, School of EE&CS, Peking University, Beijing 100871, China*
[2]*LMAM, School of Mathematical Sciences, Peking University, Beijing 100871, China*
[3]*Information Technology R&D Innovation Center of Peking University, Shaoxing 312000, China*
{zhzhao, cmjia, sswang, swma, jsyang}@pku.edu.cn

2021 IEEE International Symposium on Circuits and Systems (ISCAS) | 978-1-7281-9201-7/20/$31.00 ©2021 IEEE | DOI: 10.1109/ISCAS51556.2021.9401164

***Abstract*—In this paper, a flexible and adaptive block-wise image compression architecture with restoration network is proposed to improve the rate-distortion (R-D) performance of learned image compression. In contrast to existing learned compression algorithms which perform the image based transform coding, our approach splits the input image into non-overlapped blocks and compresses each block adaptively based on their local content characteristics. To reduce the artifacts caused by the block partitioning, block fusion network is subsequently proposed to overcome the drawbacks of block-wise independent coding. Further, quantization fine-tuning training process is proposed and utilized to reduce the difference of quantization operations between the training process and testing process. Experimental results show that the proposed network architecture has the ability to obtain significant coding gain compared with existing conventional compression standards including Versatile Video Coding (VVC) and achieves comparable R-D performance with state-of-the-art learned image codec using less parameters.**

***Index Terms*—Learned image compression, block-based coding, restoration network**

I. Introduction

Image compression has played an important role in the storage and transmission of multimedia data. In order to reduce the inherent redundancy of natural images, a series of algorithms perform linear transformations on input images to concentrate signal energy for compression. For example, Hadamard transform is used in the image compression task early in 1969 [1]. The most popular image compression standard, JPEG, adopts Discrete Cosine Transform (DCT) to reduce the image storage sizes. Subsequent JPEG2000 standard [2] applies the wavelet transform for better compression. However, considering the intricate correlation among pixels in natural images, limited linear transformation kernels can not achieve the sufficient removal of data redundancy.

Recently, deep learning based algorithms stack multiple nonlinear transformation layers to improve the representation of complex mappings. These algorithms achieve remarkable performance in computer vision tasks such as image classification [3], detection [4] and so on. In view of this, compression methods based on deep learning were considered to overcome the drawbacks of linear transformations. In 2016, Ballé et al. [5] proposed an end-to-end image compression framework which takes the entire image as input and forms the corresponding compact representation. Further work [6] deepens the network architecture and firstly achieves compression performance that exceeds JPEG2000 standard [2]. More works focus on the accurate estimation of bit consuming such as the hyper-prior information model [7] and conditional context model [8]. Although the introduction of deep neural networks improves the performance, existing learned image compression algorithms ignore the difference of local contents at different positions of the entire image and lack the adaptive processing of these contents.

In this paper, we propose a block-based learned image compression architecture using fully convolutional neural network (CNN) framework with block partition and restoration in the end-to-end manner. Further, our work introduces an adaptive normalization module to process the partitioned image blocks. This module makes the compression network focus on the relative changes of input pixels, which reduces the burden of model training. In order to obtain a compressed high quality image, we proposed a block fusion network to achieve the efficient fusion of reconstructed blocks. Quantization fine-tuning process is introduced to reduce the difference of quantization operations in training and testing. Experimental results prove that the proposed block-wise network architecture has the ability to achieve better compression performance than existing coding standards such as HEVC [9] and VVC [10]. The proposed method also obtains comparable performances with the state-of-the-art learned image codec [11] with less learnable parameters.

II. Related Work

A. Image Compression Standards

A series of image compression standards were published to explore high efficiency compression technologies. Among them, JPEG [12] is the most popular image compression standard which applies DCT on the image blocks with non-overlapped partition to obtain the compact representation. Specified quantization tables were detailedly designed aiming to preserve the low frequency information which was sensitive to human vision system. In 2002, JPEG2000 standard [2] was proposed using 2D wavelet transform for higher performance.

This work was supported in part by the National Natural Science Foundation of China (61632001, 61931014), the China Postdoctoral Science Foundation (2020M680238), PKU-Alibaba Fund, PKU-Baidu Fund (2019BD003) and High-performance Computing Platform of Peking University, which are gratefully acknowledged. * equal contribution.

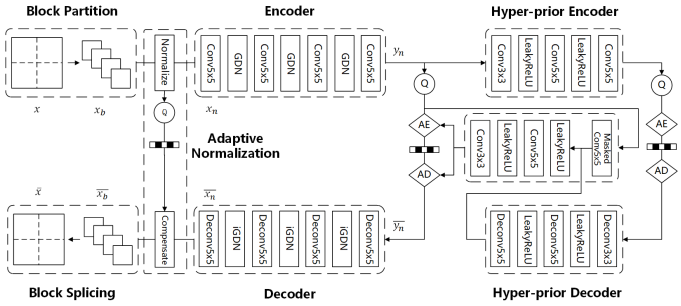
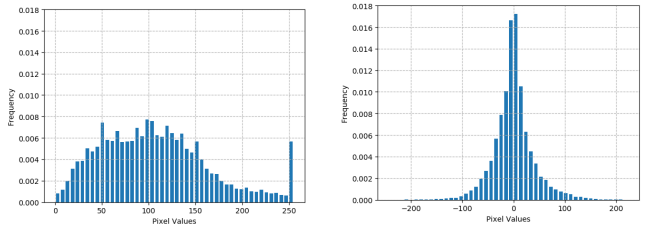


Fig. 1. Detailed architecture of block-wise compression network



(a) (b)

Fig. 2. Distribution of input values in Kodak dataset before and after the adaptive normalization. (a) Pixels before the adaptive normalization; (b) Values after the adaptive normalization.

An efficient arithmetic coding method [13] was adopted to reduce the redundancy in the wavelet coefficients. Intra coding in the video coding standards also achieved superior compression performance for still image coding. AVC [14], HEVC [9] and VVC [10] adopted directional intra prediction based on the block partition is used to explore the spatial redundancy.

B. Neural Network based Image Compression

Learned image compression has been an emerging topic in recent years [6], [7], [11], [15]–[18]. Toderici [15] et al. firstly proposed to use recurrent neural network (RNN) for this task. This work took advantage of the approximation of quantization gradient and progressive reconstruction, which achieved better compression performance than JPEG standards. Ballé et al. [5] adopted variational autoencoder to optimize the compression network. With the help of parameterized probability model, this work built an effective entropy estimation methods. It achieved competitive performance against JPEG2000. In order to describe the dependency between the encoder codes, Mentzer et al. [19] used the conditional probability model to reduce the statistical redundancy. Minnen [20] et al. combined the conditional probability model and hierarchical priors to improve the compression, which obtains better performance than HEVC. Attention module was introduced to activate important regions [21]. Recently, the state-of-the-art learned image codec has realized better coding performance than VVC [11].

III. Proposed Network Architecture

A. Block-wise Compression Network

The proposed scheme is shown in Fig .1. Each input image x undergoes three encoding stages which are called respectively as block partition, adaptive normalization and block encoding. Decoding processing is composed of inverse processes of three encoding stages. Block partition splits the input images into multiple blocks. Concretely, block partition module divides the input image x into multiple 128×128 blocks x_b without overlap. Based on the block partition, the adaptive normalization subsequently implements the adaptive encoding at the block level. Finally, a commonly used compression network in [20] is employed to encode each normalized image block.

B. Adaptive Normalization

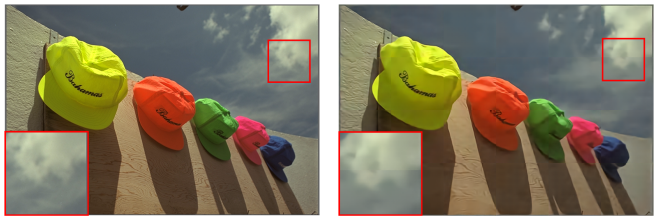
Regarding the adaptive normalization, it removes the average of the partitioned block x_b to make the block encoding network pay more attention to the relative variation in local contents. Different from the existing data pre-processing methods such as uniform normalization, adaptive normalization focuses on the local averages of small blocks rather than the entire image. As Fig. 2 shown, the distribution of input values is more concentrated after the adaptive normalization. At the same time, this operation Gaussianizes data from natural images, which has proved to be effective in the compression task [22]. In order to guarantee the decodability of pixel values, we transmit the averages of each input block channel to the decoder and compensates the missing averages to the reconstructed image blocks $\bar{x_n}$ as the inverse operation of adaptive normalization. Adaptive normalization is formulated as follows:

$$\begin{aligned} Avg^j &= round(mean(x_b^j) * 255) \\ x_n^j &= x_b^j - \frac{Avg^j}{255.0} \\ \bar{x}_n^j &= \bar{x}_b^j + \frac{Avg^j}{255.0} \end{aligned} \tag{1}$$

At the encoder side, x_b^j denotes the j-th channels in the partitioned image blocks which are scaled into the unit interval $[0, 1]$. x_n^j represents the outputs of adaptive normalization. In the decoder side, $\bar{x}_n^j$ and $\bar{x}_b^j$ are the corresponding decoded blocks and outputs of the average compensation module respectively. In particular, we represent the averages of each block in 8 bits, which reduces the transmission cost of these information.

As for block encoding, we adopt the same network architecture as [20] to obtain the compact representation of each normalized block. The detailed network architecture is shown in Fig. 1. Encoder is composed of four convolution layers and three non-linear activation layers alternately. Inside, each convolution layer performs affine transformation with kernel size as 5×5, stride as 2 and channels of 192. Generalized Divisive Normalization (GDN) [22] is used as the activation function, which is shown as following:

$$u_i^j = \frac{v_i^j}{(\beta_i + \sum_j \gamma_{i,j} * v_i^j)^{\frac{1}{2}}} \tag{2}$$



(a) (b)

Fig. 3. Illustration of artifacts caused by block-wise compression network. (a) Original input image; (b) Reconstruction of block-wise compression network.

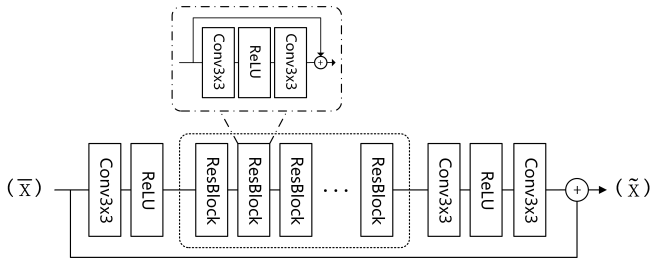


Fig. 4. Illustration of detailed block fusion network architecture

where u_i^j represents the j-th channel of the outputs in the i-th convolution layer. v_i^j denotes the corresponding output channel. β_i and $\gamma_{i,j}$ are the trainable parameters.

As for the decoder, convolution layers are replaced by the transposed convolution layers in encoder with the same setting to upsample the input features. Inverse Generalized Divisive Normalization (iGDN) is utilized to denormalize the features non-linearly, which is presented as:

$$v_i^j = u_i^j * (\beta_i + \sum_j \gamma_{i,j} * u_i^j)^{\frac{1}{2}} \tag{3}$$

Joint autoregressive and hierarchical priors model is adopted for the accurate estimation of bit consumption.

C. Block Fusion Network

Although the above block-wise compression network improves the flexibility of learned image compression, independent processing of partitioned blocks ignores the correlation among different blocks. As shown in Fig. 3, missing pixel information beyond the block boundary leads to nasty artifacts such as block effects, which lowers the coding performance. In order to overcome the drawbacks of independent processing of image blocks, we propose a block fusion network to achieve effective fusion of compressed blocks, and prove that post-processing has the ability to fully overcome the disadvantage of block partition.

Our proposed block fusion network architecture is shown as Fig. 4. The reconstructed images $\bar{x}$ obtained by block splicing module is the input of this fusion network. First convolution operation is used to extract valid features in the inputs $\bar{x}$. Rectified linear unit (ReLU) $f(x) = max\{x, 0\}$ is utilized as the activation function to provide the nonlinear transformation. Then, ten residual blocks (ResBlock) are stacked to realize more complex feature extraction process. Each residual block is composed of two convolution layers as well as sandwiched ReLU activation function. Finally, two convolution layers along with a ReLU is used to get the approximate reconstruction residual between refined images $\bar{x}$ and input x. In particular, all the convolution layers have a trainable kernel with spatial shape 3×3. Input features are padded with zero to keep the same spatial shape. Except for final convolution, other convolution layers set the number of output channels as 64 for sufficiently rich feature extraction.

D. Quantization Fine-tuning Process

Quantization is one of the most important modules in the image compression task. Continuous outputs of the encoder are processed into discrete symbolic representation, which leads to the limited code length. However, quantization causes the meaningless gradient which is zero in almost everywhere. To realize the end-to-end optimization of compression network, researchers proposed a series of gradient approximation to overcome this problems. One of the gradient approximation methods is proposed in [23] to define the quantization as $Q = round(x)$ and the gradient as $\frac{dQ}{dx} \equiv 1$. Although this approximation implements the gradient descent optimization of the encoder, inconsistency between forward operation and backward propagation leads to biased gradient information, which lowers the compression performance. Ballé [6] adopted a variational auto-encoder architecture to transfer the valid gradient in which quantization is formulated as follows:

$$Training : \tilde{y} = y + e, e \sim U[-0.5, 0.5] \tag{4}$$

$$Testing : \tilde{y} = round(y) \tag{5}$$

where $U[-0.5, 0.5]$ denotes the uniform distribution in the interval $[-0.5, 0.5]$. However, different operations in training and testing hinder the performance improvement.

In this paper, we add the quantization fine-tuning process to reduce the difference of quantization in training and testing. The training process is split into two stages. Firstly, the quantization approximation in (4) is used to provide meaningful gradients to the encoder. Different from the quantization operation in [23], this approximation provides unbiased gradient information during training, which reduce the subsequent loss to performance. Secondly, the quantization fine-tuning process is introduced to reduce the impact of the approximation process on the decoder and block fusion network. Concretely, we fix the weights of trained encoder network and use rounding function in (5) which is same as the operation in testing to update the decoder and block fusion network. By adding the quantization fine-tuning process, the decoder and block fusion network adapt to the quantized codes, which improves the compression performance.

IV. Experiments

A. Experimental Settings

We implemented our proposed network architecture described in Sec.III using PyTorch framework [24]. In order to

improve the diversity of training dataset, Waterloo exploration database [25] and DIV2K training dataset [26] were combined as the training data, which include images with different resolutions. Our training loss function is formulated as follow:

$$loss = \lambda * 255^2 * ||\tilde{x} - x||_2^2 + \mathbb{E}_{\bar{z}\sim P_{\bar{z}}, \bar{y}\sim P_{\bar{y}|\bar{z}}}\{-log\, q_{\bar{y}|\bar{z}}(\bar{y}|\bar{z}) - log\, q_{\bar{z}}(\bar{z})\}. \quad (6)$$

In (6), the first item measures the reconstruction quality using $L2$ norm. x denotes the input images which are normalized to the unit interval $[0, 1]$. $\tilde{x}$ denotes the reconstructed images after the block fusion network. The second item computes the entropy of the compressed representation. $\bar{y}$ and $\bar{z}$ represent the quantized codes and hyper-prior information shown in Fig. 1 respectively. $P_{\bar{z}}$ is the real distribution of hyper-prior information and $P_{\bar{y}|\bar{z}}$ is the real conditional distribution of codes based on the hyper-prior information. Neural network based non-parametric density model and conditional Gaussian model [20] were utilized to approximate the real distribution $P_{\bar{z}}$ and $P_{\bar{y}|\bar{z}}$. The proposed network was optimized by Adam optimizer [27]. We adopted 5×10^{-5} as the learning rate to optimize the trainable parameters with a mini-batch of 8. To accelerate training, the block-wise compression net and the block fusion net were firstly trained respectively for 5 million steps. Then these two models were then fine-tuned in the end-to-end manner for 1 million iterations. Finally, the quantization fine-tuning process was used to reduce the gap of quantization operations. Parameters of the decoder and block fusion network were further updated for 1 million iterations.

B. Test Conditions

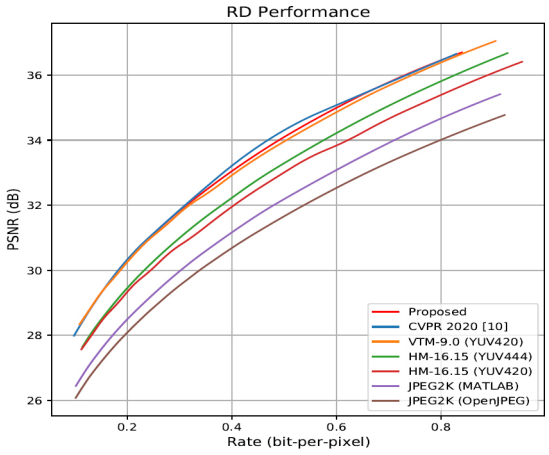
Our proposed image compression architecture is tested on Kodak dataset[1] and CLIC test dataset ("Professional")[2]. Kodak dataset consists of 24 lossless true color images with the spatial resolution 512×768 or 768×512. In order to fully present the effectiveness of the proposed block-wise architecture on high-resolution images, we also test our architecture on CLIC test dataset. We utilized the center-cropping to preprocess CLIC dataset to make resolutions to be multiple of 128. To realize different bit-rate points, the λ is selected from the set $\{0.003, 0.005, 0.01, 0.03, 0.05\}$ independently.

C. Performance Comparison

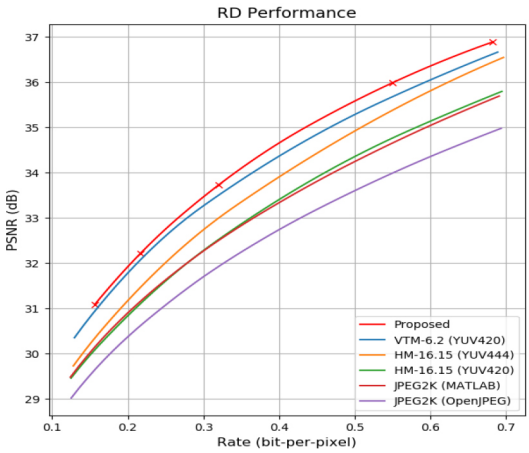
The rate-distortion curve is shown in Fig. 5. From this figure, our proposed block-wise compression architecture has better compression performance compared with other algorithms. In particular, our proposed network exceeds the different implementations of JPEG2000 and HEVC standards significantly. In Kodak test dataset, our reconstruction quality is 0.12 dB higher than VTM-6.2 intra coding at 0.3741 bpp. These experimental results proves that our block-wise compression network provides a more flexible architecture and obtains better compression performance at the same time. And our approach with a single guassian entropy hyper prior obtains similar performance with the scheme in [11] who utilizes a mixed of three gaussian entropy model. Less parameter but comparable performance is another contribution of this paper. We could additionally learn that the proposed scheme achieves significant improvement on the high-resolution images from Fig. 5 (b).

[1]http://http://r0k.us/graphics/kodak

[2]https://data.vision.ee.ethz.ch/cvl/clic/test/professional_test.zip



(a)



(b)

Fig. 5. Performance comparison of different algorithms. (a) RD Performance on Kodak dataset; (b) RD Performance on CLIC test dataset;

V. Conclusion

In this paper, a block-wise compression architecture is proposed to achieve the adaptive process of local contents at different positions in the entire image. Concretely, the entire image are split into non-overlapped blocks by the block partition mechanism. Adaptive normalization makes the compression network focus on the relative changes in pixel values, which reduce the difficulty of training. Subsequent block fusion network has the ability to overcome the drawbacks caused by the block partition. Experimental results confirm that the proposed architecture provides the encouraging coding gain as well as a more flexible compression framework using less parameters.

REFERENCES

[1] W. K. Pratt, J. Kane, and H. C. Andrews, "Hadamard transform image coding," *Proceedings of the IEEE*, vol. 57, no. 1, pp. 58–68, 1969.

[2] M. Rabbani, "JPEG2000: Image compression fundamentals, standards and practice," *Journal of Electronic Imaging*, vol. 11, no. 2, p. 286, 2002.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[5] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimization of nonlinear transform codes for perceptual quality," in *2016 Picture Coding Symposium (PCS)*. IEEE, 2016, pp. 1–5.

[6] ——, "End-to-end optimized image compression," *International Conference on Learning Representations (ICLR)*, 2017.

[7] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," *arXiv preprint arXiv:1802.01436*, 2018.

[8] J. Lee, S. Cho, and S.-K. Beack, "Context-adaptive entropy model for end-to-end optimized image compression," *arXiv preprint arXiv:1809.10452*, 2018.

[9] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (hevc) standard," *IEEE Transactions on circuits and systems for video technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[10] ITU-T and ISO/IEC, "Versatile video coding," 2020, ITU-I Rec. H.266 and ISO/IEC 23090-3, to be published.

[11] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Learned image compression with discretized gaussian mixture likelihoods and attention modules," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[12] G. K. Wallace, "The JPEG still picture compression standard," *IEEE transactions on consumer electronics*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

[13] D. Taubman, "High performance scalable image compression with ebcot," *IEEE Transactions on image processing*, vol. 9, no. 7, pp. 1158–1170, 2000.

[14] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h. 264/avc video coding standard," *IEEE Transactions on circuits and systems for video technology*, vol. 13, no. 7, pp. 560–576, 2003.

[15] G. Toderici, D. Vincent, N. Johnston, S. Jin Hwang, D. Minnen, J. Shor, and M. Covell, "Full resolution image compression with recurrent neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5306–5314.

[16] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Energy compaction-based image compression using convolutional autoencoder," *IEEE Transactions on Multimedia*, vol. 22, no. 4, pp. 860–873, 2019.

[17] C. Jia, Z. Liu, Y. Wang, S. Ma, and W. Gao, "Layered image compression using scalable auto-encoder," in *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*. IEEE, 2019, pp. 431–436.

[18] S. Ma, X. Zhang, C. Jia, Z. Zhao, S. Wang, and S. Wang, "Image and video compression with neural networks: A review," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 6, pp. 1683–1698, 2019.

[19] F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, and L. Van Gool, "Conditional probability models for deep image compression," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4394–4402.

[20] D. Minnen, J. Ballé, and G. D. Toderici, "Joint autoregressive and hierarchical priors for learned image compression," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 771–10 780.

[21] H. Liu, T. Chen, Q. Shen, and Z. Ma, "Practical stacked non-local attention modules for image compression," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[22] J. Ballé, V. Laparra, and E. P. Simoncelli, "Density modeling of images using a generalized normalization transformation," *arXiv preprint arXiv:1511.06281*, 2015.

[23] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," *arXiv preprint arXiv:1703.00395*, 2017.

[24] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," *In NeurIPS Workshop*, 2017.

[25] K. Ma, Z. Duanmu, Q. Wu, Z. Wang, H. Yong, H. Li, and L. Zhang, "Waterloo exploration database: New challenges for image quality assessment models," *IEEE Transactions on Image Processing*, vol. 26, no. 2, pp. 1004–1016, 2016.

[26] E. Agustsson and R. Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.