Enhancement of images compression using channel attention and post-filtering based on deep autoencoder

Andri Agustav Wirabudi [a,b,1,2,*], Nurwan Reza Fachrurrozi [b,3], Pietra Dorand [b,4], Muhamad Royhan [b,5]

[a] Department of Intelligence Media Engineering, Hanbat National University, 34158 Daejeon, South Korea
[b] School of Applied Science, Telkom University, 40257 Bandung, West Java, Indonesia
[1] 30221063@edu.hanbat.ac.kr; [2] andriagustaw@telkomuniversity.ac.id; [3] nurwan@telkomuniversity.ac.id; [4] pietra@telkomuniversity.ac.id;
[5] roihani@telkomuniversity.ac.id
* corresponding author

ARTICLE INFO

Article history
Received January 31, 2024
Revised April 06, 2024
Accepted April 07, 2024
Available online August 31, 2024

Keywords
Channel attention
Auto encoder
Post-filtering
Deep learning
Image compression

ABSTRACT

Image compression is a crucial research topic in today's information age, especially to meet the demand for balanced data compression efficiency with the quality of the resulting image reconstruction. Common methods used for image compression nowadays are based on autoencoders with deep learning foundations. However, these methods are limited as they only consider residual values in processed images to achieve compression efficiency with less satisfying reconstruction results. To address this issue, we introduce the Attention Block mechanism to improve coding efficiency further. Additionally, we introduce post-filtering methods to enhance the final reconstruction results of images. Experimental results using two datasets, CLIC for training and KODAK for testing, demonstrate that this method outperforms several previous research methods. With an efficiency coding improvement of -28.16%, an average PSNR improvement of 34%, and an MS-SSIM improvement of 8%, the model in this study significantly enhances the rate-distortion (RD) performance compared to previous approaches.

1. Introduction

Today's advancement of digital media has led to increased data access, particularly concerning high-definition image content. The more images exchanged within a network, the more impact it has on data compression and the required data transmission. The need for this significantly influences the quality of the produced images. Research on data compression currently stands as the best solution to address the issues [1]. Compression techniques fundamentally aim to reduce spatial redundancy in images and optimize the available bandwidth and storage space. In conventional data compression research such as JPEG (Joint Photographic Expert Group) [2] and JPEG 2000 [3], [4], it is insufficient to meet the current data compression needs. This is because conventional methods focus more on capturing the residual values of images to be represented back into their original form. Additionally, conventional methods [5] are considered inadequate for reconstructing images into their original forms upon reception, thereby failing to maintain the visual quality of the reconstructed results. This contradicts the concept of compression itself, which involves reducing data size while retaining the information within to approximate the original.

Modern deep learning-based image compression algorithms currently show improved compression results compared to previous conventional methods, with advancements in rate-distortion (RD) performance and coding efficiency [5]. Research conducted by Tawfik et al. [6] utilizes a real-time generic convolutional autoencoder (CAE) approach capable of achieving high compression ratios, albeit sacrificing image quality for coding efficiency. Additionally, research by Cheng et al. [7]. applies a convolutional autoencoder architecture with a hyperprior trained using distortion rate functions to achieve high coding efficiency by reducing the BD rate. Cheng et al. also employ Non-local Attention Module (NLAM) and Gaussian Mixture Likelihoods (GML) mechanisms to eliminate spatial redundancy in compressed code and achieve a more accurate entropy model, resulting in fewer bits for encoding processes. However, the attention mechanism in their use may reduce some information, especially in testing at low λ. Elsewhere, a study conducted by Google's Nick Johnston [8] showed that MS-SSIM-weighted pixel loss training improved the quality of corresponding reconstructions by several metrics. He modified the recurrent architecture to improve spatial diffusion, allowing the network to capture more effectively and propagate image information through hidden states in the network. Research conducted by Balle combines hyperparameter blocks with variational autoencoder blocks to achieve better loss efficiency and BD rates. The hyperprior method was first introduced by Ballè et al. [9], where Ballè introduced a nonlinear model for image compression using an end-to-end method. This method consists of nonlinear analysis transformation, uniform quantization, and nonlinear synthetic transformation. The stages in Ballè's model are built in three sequential steps, including linear, convolutional, and nonlinear activation functions. The Hyperprior component functions to capture spatial dependencies in latent representation and achieve good image compression performance.

Inspired by the working mechanism of attention mechanisms in [10], [11], which can improve encoding efficiency and significantly enhance compression performance, in this research, we implemented an attention mechanism based on Channel Attention Block (AB) [11]. In addition to the attention mechanism, we introduced a post-filtering block to enhance image reconstruction to approximate the original image. We divided these two blocks to focus on two different pixel values, namely dominant and non-dominant, in an image. The dominant values refer to the object part of the image, while the non-dominant values refer to the background of the image. The use of the AB block in this model aims to focus attention on the image object so that dominant pixel values will be prioritized in the process, allowing the neural network to focus on important features only. Meanwhile, the post-filtering mechanism is widely used in video compression research [12], aiming to improve image performance by enhancing non-dominant pixel values to approximate the original image. The reason for dividing these two blocks to perform different tasks is because previous research [10], [13] on each of these blocks had weaknesses, such as AB focusing only on dominant pixel values and post-filtering having dependencies on the quality of the original image [14], decreasing decompression speed due to requiring additional processes after completion, and other mathematical computation increases [15]. We see these weaknesses as an opportunity to collaborate and innovate by dividing the tasks of each block into different processes. Thus, collaborating these two blocks will greatly help improve coding efficiency and enhance the quality of the final image. Hyperprior autoencoder-based models [9] and Convolutional Neural Network (CNN) [16] are used in this research. The AB block is placed at each final stage of the encoder, decoder, and hyperprior section.

This paper is structured in several stages as follows. In Section II, the method, main references, and other relevant supporting references to this research will be discussed. Some of these references have been addressed in Section I, which we use as the state-of-the-art (SOTA) to test the data performance

conducted and to compare the image quality in Section III. Additionally, this section also contains architecture diagrams and mathematical equations used in this research, including the Hyperprior Model, AB, Postfilter, the dataset used to train and test the model, and the Evaluation Metrics used. Section III contains qualitative results of image reconstruction and qualitative comparisons with relevant SOTA. Finally, in Section V, we conclude the overall results and future research plans.

2. Method

2.1. Related Works

This section explains the differences between the existing current compression methods and the methods used in this research.

2.2. Conventional Methods

In this current field of image compression, the techniques focus more on reducing spatial redundancy, such as changing images from the pixel domain to the frequency domain, to make the compression process easier. For example, JPEG [2] which applies discrete cosine transformation. In contrast, JPEG2000 [3], [17] applies manual discrete wave transformation. To reduce data redundancy, high-frequency information is separated from low-frequency information, and bits are allocated based on signal significance. Next, Entropy such as Huffman [18], [19], arithmetic coding, [20], and the like are also used to increase image compression. Another way of using an intra-prediction approach [5] is similarly used for both image and video compressions. The BPG (Better Portable Graphics) [20] is based on the image compression standard of High-Efficiency Video Coding HEVC/H.265 [21], which gave more optimal image compression results than those of previous methods, such as JPEG [2] and JPEG2000 [3], [17]. The prediction-transformation standard uses the BPG approach, and 35 options of intra-prediction directions are used to create reconstructed images and reduce redundant data. Furthermore, larger computing units, more prediction methods, a greater variety of transformations, and more coding facilities are all supported by VVC (Versatile Video Coding) [15]. However, the conventional method presented was created through manually designed components such as entropy coding and prediction.

2.3. Deep Learning–Based Method

DNN (Deep Neural Network) based methods are modern image compression approaches that use neural networks to compress images. Neural networks inspire these DNN models and can learn from data to identify complex and abstract patterns [14], [21], [22]. In the context of image compression, DNN-based methods combine the principles of Deep Neural Networks with traditional compression techniques to create a more efficient way of representing and storing image data. The general steps used in the DNN Method are Representation Approach, Model Training, Encoding, Compression, and Decompression. In the training process, some methods [23]–[25] use perceptual weighted training loss, hidden state priming, and spatial adaptive bit rate. The combination of these three techniques improves the performance of image compression architectures. Therefore, this research proposes an autoencoder technique by adding an attention and post-filter module to improve image quality after the reconstruction process is complete and the loss efficiency that occurs. Furthermore, the concept utilized in this research is adversarial image compression, introduced by Balle et al. [9]. As explained earlier, the model we employ consists of variational and hyperparameter components, as depicted in Fig. 1. Additionally, the model incorporates the Generalized Divisive Normalization (GDN) layer. This model is chosen because it contains two entropy bounds, enabling the estimation of entropy probability values at each stage. In various research cases [23]–[26] the Balle model can be extended by incorporating

additional blocks or modules to strengthen compression values and improve the quality of the resulting reconstructed images.

2.4. Methodology

This section presents the image compression performance framework in detail. Fig. 1 shows the use of Variational autoencoder architecture by adding Hyperparameter models. This research developed and improved performance compared to previous studies [6], [7], [9], in which we added model parameters such as post-filtering and Channel Attention into the model. CNN architecture has proven to be able to compress images well by capturing spatial relationships while efficiently improving compression performance through entropy models. The use of the attention module can help the compression and reconstruction process become more efficient because the nature of this attention model will focus on the block that contains information. Once the composition process is completed, it is continued with the quantization process, which is used during the formation of the entropy model to increase the probability that it can affect the overall performance to a better compression result of the reconstructed image. Hyperparameters influence how well models can learn patterns in feature blocks, improve model performance, and avoid overfitting or underfitting during training. In the training process, you lowered the image resolution to 256 × 256. The image downscale process to the size of 256 × 256 aims to speed up the required encoding and decoding time. After that, the compression result is performed at a stage (Encoder), producing an l_a output, which contains compression features that will be sent to the decoder stage for the reconstruction process, before passing through the hyperparameter encoder stage used to overfit and streamline the losses obtained.

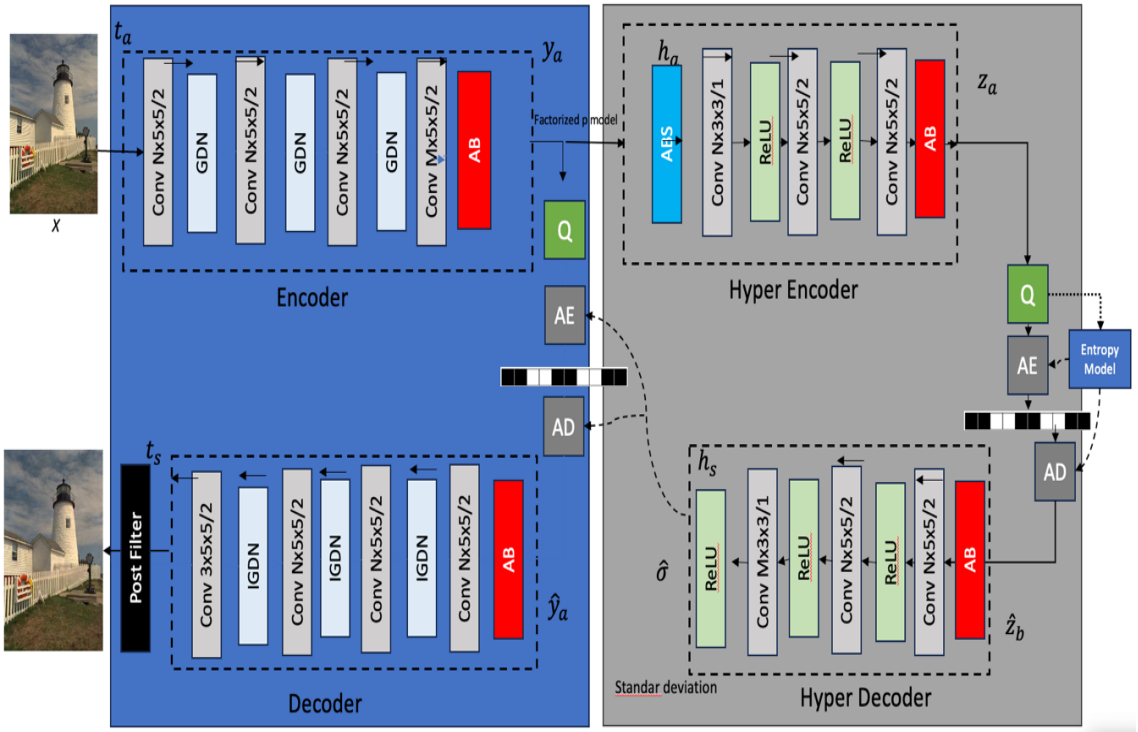


Fig. 1. In the proposed network architecture model, two components are distinguished. The left part illustrates the autoencoder architecture, while the right part represents the hyperprior model.

The z part of the model is the output and input of the hyperparameter h_a for the hyper encoder and h_s for the hyper decoder. The encoding uses the quantified vector $\hat{z}_b$ to predict the spatially distributed

$\hat{\sigma}$ value of the standard deviation and the result of such a prediction is used to compress and transmit representations of quantized images $\hat{y}_a$. At the decoder stage, the image will first be recovered by the upsampling process, i.e., returns the image quality based on the information of the Entropy and quantization performed, but before the reconstruction process is carried out, the décoder must first recover the $\hat{z}_b$ from the compressed signal. Then, to get the value of $\hat{\sigma}$, use the output of h_s, where the result of the recovery is $\hat{\sigma}$, $\hat{\sigma}$ containing information that estimates the correct probability value for performing reconstruction and recovering $\hat{y}_a$.

After the $\hat{y}_a$ value has been known, then perform the reconstitution phase t_s to recover the image $\hat{x}$. Once the image has been reconstructed, the last step is to provide a filter post to improve the quality of the image that is close to the original x image. This compression process aims to generate high-quality reconstruction images at a particular bitrate, and the entropy model is used to predict the target bitrate. This entropic model uses arithmetic encoding to estimate the latent form before performing the quantization process and encode the coding result into the bit stream. Such bit information is important for the decompression process. In addition, a proper entropy model will improve compression efficiency. In the training phase, we used 256 × 256 image resolution with the aim of speeding up the training process, and we used the actual resolution of the codec data set of 768 × 512.

Below is the mathematical equation of the loss function (1) in the autoencoder, which is used to optimize the compression model training process, as shown in Fig. 1.

$$L = D + R = d(x, \hat{x}) = ||x - \hat{x}||^2 \quad (1)$$

When,

L is the loss function, D is distortion, R is bitrate, x is the input of the initial image, $\hat{x}$ is the reconstructed output of the autoencoder process, and $||x - \hat{x}||^2$ is the square Euclidean norm for measuring the difference between x and x^.

Entropy estimation method applied during the training process is shown in the form of equation (2) below:

$$P(\hat{y}_a | \hat{z}_b) = \prod_i \mathcal{N}(\varphi^i, \vartheta^{2(i)}) * \mathcal{U}\left(-\frac{1}{2}, \frac{1}{2}\right)(\hat{y}_{a_i}) \quad (2)$$

Each latent representation $\left(\hat{y}_{a_i}\right)$ is expressed as a Gaussian distribution with parameters φ^i and ϑ^i, predicted by the probabilities of hidden elements $\hat{z}_b$. $\hat{z}_b$ is referred to as the hyperprior, $\mathcal{U}$ denotes a uniform distribution and $*$ denotes the convolution process. The hyperprior $\hat{z}_b$ is represented in equation (3) as follows.

$$P_{\hat{z}_b|\psi}(\hat{z}_b|\psi) = \prod_i (P_{z_a|_{\psi^{(i)}}}) * \mathcal{U}\left(-\frac{1}{2}, \frac{1}{2}\right))(\hat{y}_{a_i}) \quad (3)$$

In this context, each distribution is represented by $P_{\hat{z}_b|\psi(i)}$ and its parameters are denoted by $\psi^{(i)}$ Our technique comprises the bit rate for the hidden variable. $\hat{z}_b$ and the latent representation $\hat{y}_a$ However, the bits from equation (2) can be represented as follows:

$$\hat{y}_a = \sum_i - log_2 \left(P_{\hat{y}_a|\hat{z}_b}(\hat{y}_{a_i}|\hat{z}_{b_i})\right) \quad (4)$$

$$\hat{z}_b = \sum_i - log2 (P_{\hat{z}_b|\psi}(\hat{z}_b|\psi^{(i)})) \quad (5)$$

2.4.1. Training Process

In the model training process, we lowered the initial image resolution to 256 × 256 × 3, which aims to speed up the testing model's training process. A set of blocks of images is performed through the feature compression process (Downsampling) up to the feature extraction process to the bottleneck, at the bottleneck stage containing all the information that has been compressed, this section is later the key to the reconstruction of the image at the decoder stage. An illustration of the stages of the process is shown in Fig. 2.

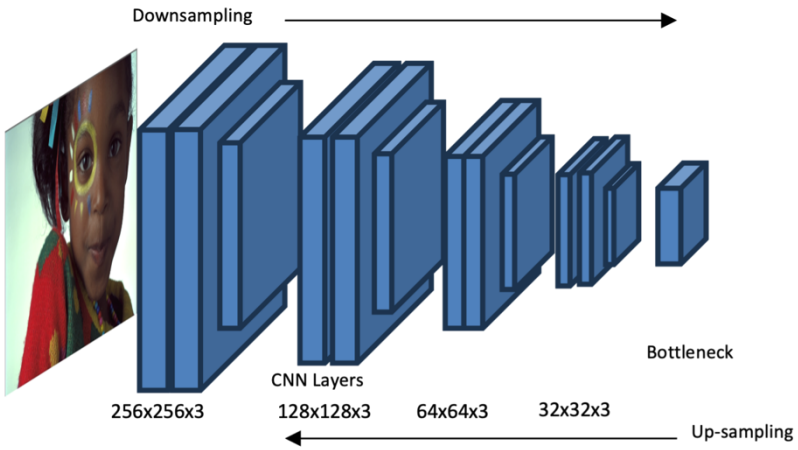


Fig. 2. Illustration of Downsampling in the training process using the CLIC dataset and testing using the Kodak dataset by changing the image into several feature blocks before finally returning it to full size in the reconstruction process from the decoder

2.4.2. Channel Attention Modul

We use the attention block mechanism that is placed after the compression process is obtained on y_a. This channel attention is used to understand and highlight the relevant information in each channel of a feature map of an image. A feature map is a hierarchical representation of the image produced by the convoluted nerve network (CNN) during the extraction process. The type of channel attention model we used was based on Sanghyun Woo's reference [11]. Indirectly, this method can help the model study the compressed image and enhance the network by observing the inter-channel relationship. Fig. 3 below is a model of the channel attention that we use.

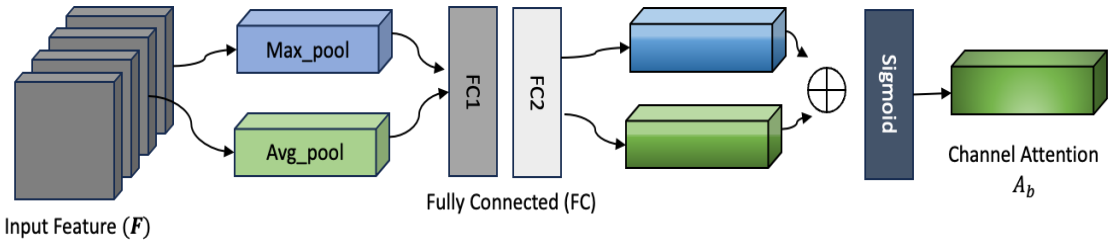


Fig. 3. The attention mechanism works by allocating weights (values) to each input based on its relevance to other elements. By assigning higher weights to more relevant elements. The model can focus on important parts of the images

$$\mathrm{A_b}\,(F) = \sigma(FC(Avgpool(\boldsymbol{F})) + FC\,(MaxPool(\boldsymbol{F}))$$

$$= \sigma\left(W_1\left(W_0\boldsymbol{F}^c_{avg}\right)\right) + W_1(W_0(\boldsymbol{F}^c_{avg})) \quad (6)$$

First, we collect spatial information from the feature map max_{pool} and avg_{poll} operations and generate values $FC(Avgpool(\boldsymbol{F}))and\ FC\ (MaxPool(\boldsymbol{F}))$, each of which shows the values of the average and maximum features. The two descriptors are then forwarded to the joint network to generate attention blocks on the Fully Connected layer $(FC)A_b \in RC \times 1 \times 1$, within the joined network consisting of multi-layer perceptrons (MLP) with one hidden H layer. Where σ indicates the sigmoid function, $W_0 \in \frac{RC}{r} \times\ C\ and\ W_1\ \in\ RC\ \times \frac{C}{r}$. It should be noted that weights $\boldsymbol{MLP}, W_0\ and\ W_1$, used jointly for both inputs and activation functions **ReLU** are followed by $W0$. To measure the load of the parameter in using this attention module, the hidden activation size is set to be $\frac{RC}{1r} \times 1 \times$ where r is the reduction ratio, which plays an important role in setting the value of the attention used; the larger the reduction ratio then, the larger the reductions are made. Note adjusts the reduction value according to the image to be tested. Once all the processes have passed, the attention module can be unplugged $A_b(F)$.

2.4.3. Post Filter

In this section, the postfilter block is proposed after the compression and reconstruction process is completed. The aim is to perfect the results or soften the reconstructed image, which may contain artifacts in the form of image fragments resulting from the merging process, as seen in Fig. 4. The artifacts referred to here are parts of the background in the image that do not have perfect pixel values, caused by an imperfect reconstruction process. This happens because the background part of the image is not subject to an attention mechanism, so the attention mechanism only focuses on the foreground area and ignores the background part where dominant information is minimal. In previous research [6], postfilter was not used, which had an impact on the reconstructed images. This postfilter will help increase the probability value of the image pixels so that it can improve the background quality for the better.

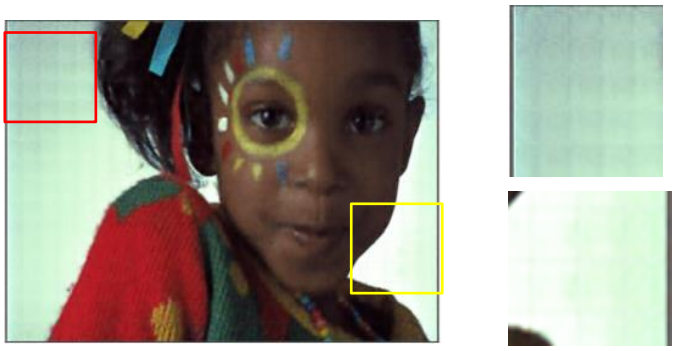


Fig. 4. Images containing artifacts resulting from decreasing resolution at each block during the compression stage are reintegrated into the image reconstruction process

In Fig. 5, a diagram of the post-filter is shown, consisting of convolutional blocks $(conv)$, where N represents the value of the filter used, which is 192. Additionally, the use of $PReLU$ activation is depicted. We adopted the Parametric Rectified Linear Unit $(PReLU)$ [27] over the Rectified Linear Unit (ReLU) [28]. This PReLU activation function is defined as $(f(x) = \max(0, x) + \alpha \min(0, x))$. With the parameter (α) that can be learned during the training process, PReLU has an advantage in addressing the "dead neuron" issue that occurs with ReLU because (α) can be adjusted to overcome saturation. Following this, the process continues with sigmoid activation (σ), then transformed back to the original size and evaluated for image quality $(\hat{I})$, the reconstructed result based on the input image (I).

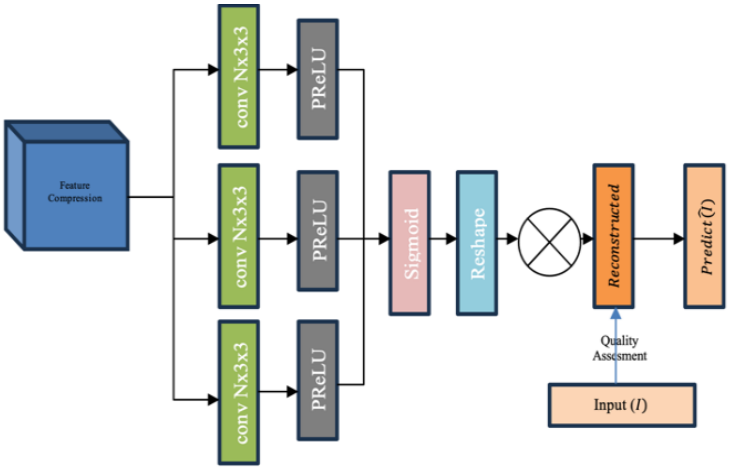


Fig. 5. The post filter layer consists of three convolutional layers with a 3×3 kernel size & N values of 192, 192, &3. We use **PreLU** during the filtering process to ensure that negative values are not eliminated. Afterward, the output from the post-filtering stage generates the Reconstructed output. We then assess the quality by comparing the original input,I, using parameters such as ***PSNR*** & ***MSE***, resulting in the outcome I

2.5. Experiments

2.5.1. Dataset

This research applied two types of datasets: training data and validation data, which use CLIC [29]. For testing, the KODAK dataset [30] was then used. The KODAK dataset consisted of 24 photos with a resolution of $512 \times 758\ pixels$, while the CLIC dataset consisted of 1633 images for training and 102 images for validation. The CLIC dataset has a resolution of each image of $1913 \times 1361\ pixels$ for images taken using a cellphone camera, and around 1803×117 pixels for images taken with a professional camera. Such a dataset was chosen as it performed more optimally for image compression.

2.5.2. Training Details

All experiments were carried out on a workstation under Ubuntu 20.04 with an AMD Ryzen 5 5600X 6-core Processor, 32GB RAM, and one NVIDIA GeForce RTX 3060 12GB GPU running under CUDA 12.2. Python version 3.8.0 in a TensorFlow version 2.12 environment within Docker was then employed to complete such experimental code. The model implementation that was used to optimize learning is Adam [31], which is useful for training all models with a batch size of 256, total parameters of 2.2 million, and Trainable params of 2.3 million for each iteration. The learning rate used starts from $lr = 0.003$ to 0.00003, with the Epoch: 500 and step per epoch: 500, filter sizes used in convolution layers $N = 192$ and $M = 320$, and on each layer the size of the post filter is 512,768,3.

2.5.3. Evaluation Metric

The evaluation metrics we use to measure the accuracy of results and comparisons between models involve measuring the level of distortion in bits per pixel (bpp) [32]. This metric is utilized to observe how many bits are used to represent pixels in an image. Furthermore, it is also employed to ascertain the quality of compression obtained from each λ during testing. Additionally, we employ the Mean Square Error (MSE) [33] metric to gauge the level of distortion between two signals, namely the original signal and the signal generated during training. Here, the value x of the input is subtracted from the value $\hat{x}$ of the reconstructed image, as indicated in Equation 7. After obtaining the MSE value, we

proceed to find the Peak Signal-to-Noise Ratio (PSNR) [33], [34]. This metric is utilized to measure the quality of image compression or the reconstruction of images, as described in Equation 8.

Moreover, we utilize the Multi-Scale Structural Similarity Index (MS-SSIM) [35] to measure the extent to which two images differ in terms of structure, texture, and detail, as shown in Equation 9. Here, L represents the number of scales, ι(i) denotes the intensity difference contribution at scale i, c(i) signifies contrast and s(i) represents structural information. This metric considers the structural differences between two images by considering variations at various scales, making it more sensitive to changes occurring in the image.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(X_{ref}(i) - \hat{X}_{test}(i))^2 \tag{7}$$

$$PSNR = .10.log_{10}(\frac{Max\ Pixel\ Value^2}{MSE} \tag{8}$$

$$MS - SSIM = (x, y) = \frac{l}{L}\sum_{i}^{L} l(i).c(i).s(i) \tag{9}$$

3. Results and Discussion

3.1. Qualitative Result

This section presents some visualization results to clarify the results of the approach taken. Some test–image samples that have been used during testing are shown in Fig. 6 and Fig. 7. These images show a qualitative comparison (final reconstructed images) in the KODAK dataset [30]. Fig. 6 compares the result of the approach to the existing methods [2], [6], [7], [9]. To illustrate the efficiency of the proposed technique, we highlight several specific areas in the reconstructed images for a more in-depth examination of image details. Images with our proposed method have better PSNR and bpp values, namely 35.3 dB (kodim15.png), and dB (kodim14.png). Besides, our proposed method can maintain almost the same bit rate as other methods. Apart from that, the image texture is more similar (especially in the eye area), and this method succeeds in maintaining fine features in the image after compression.

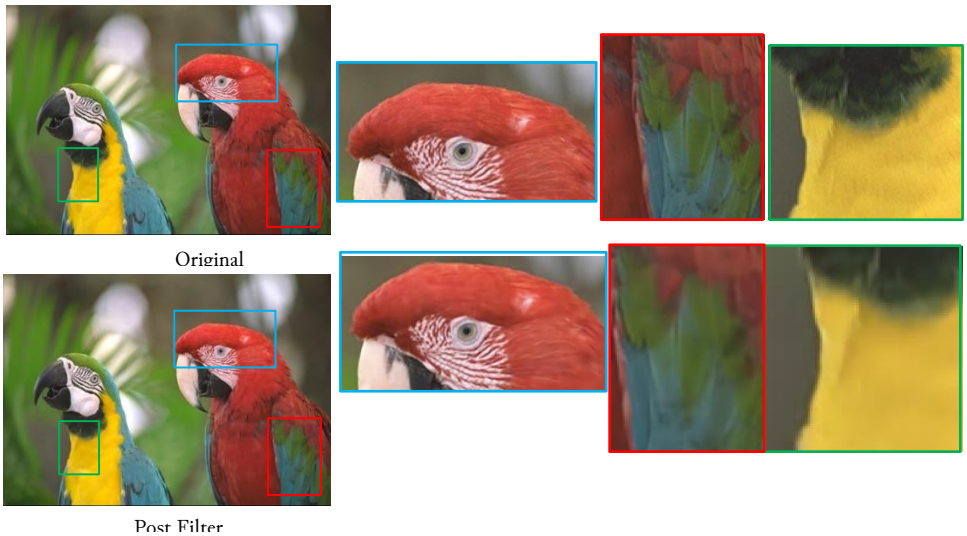


Fig. 6. Qualitative performance comparison dataset Kodim 23 between original and use post filtering after reconstructed

Meanwhile, Fig. 7 shows a comparison of the reconstructed image using a post filter on the original image at various zoom levels for better qualitative visualization.

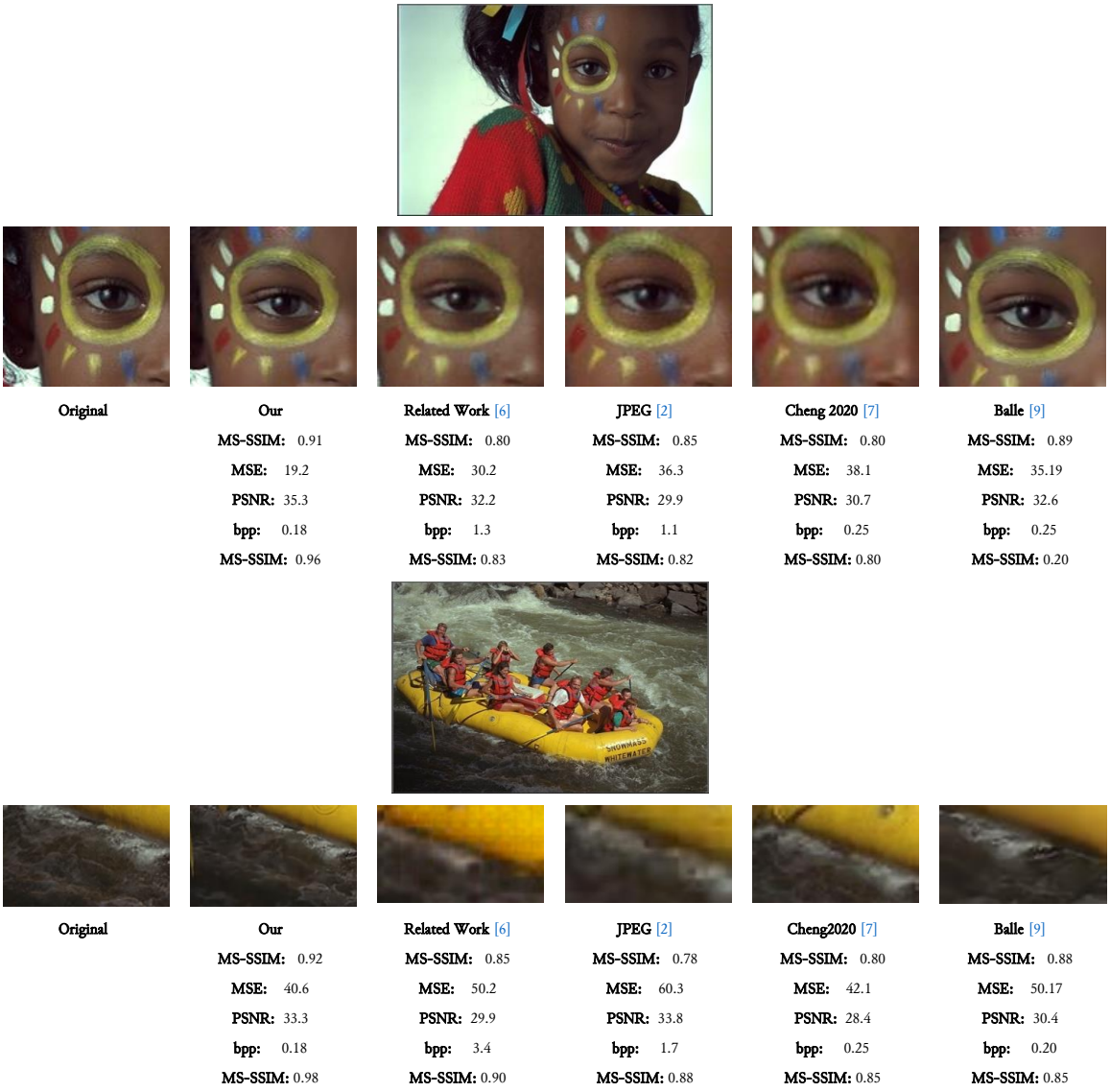


Fig. 7. Qualitative performance comparison Kodim 15, and Kodim 14 of our reconstructed images related work [2], [6], [9], [10] These images are taken from KODAK dataset

Visualization results of Attention Blok (AB) regarding kodim 23.png kodim 19.png from KODAK dataset as show in Fig. 8.

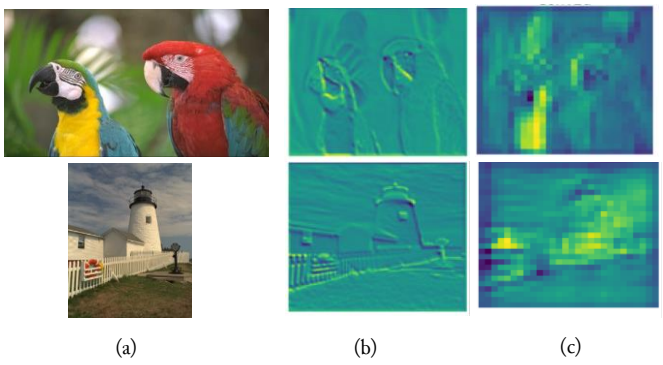


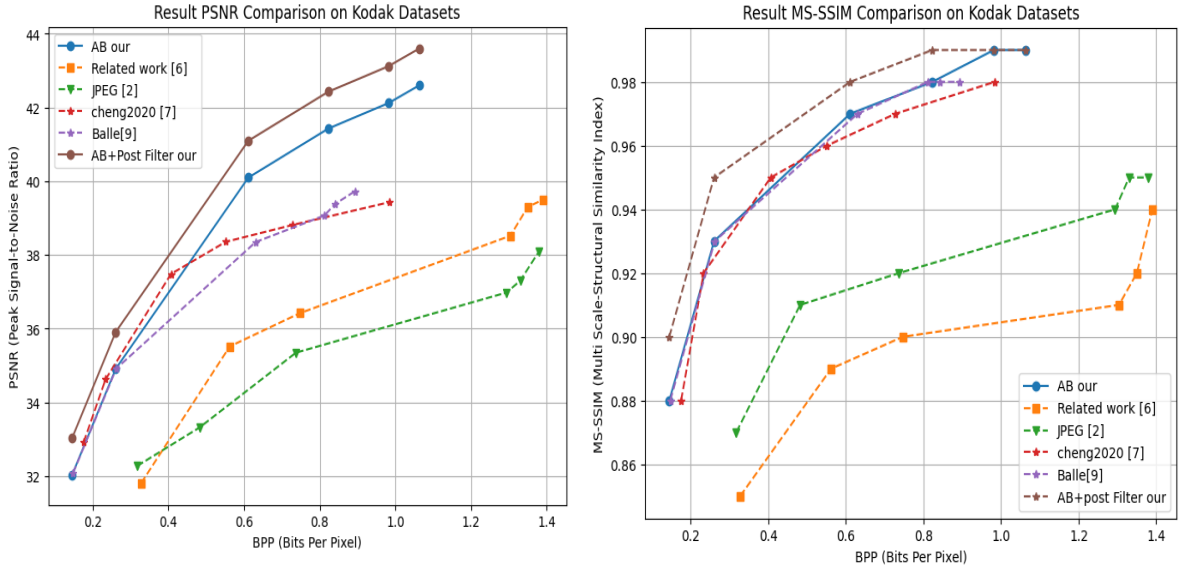
Fig. 8. Visualization results of Attention Blok (AB) regarding kodim 23.png kodim 19.png from KODAK Dataset (a) original image, (b) latent, and (c) Allocated bits in train model

The use of a post filter after the construction process also increases the PSNR value in the image and restores lost pixel values. Typically, textured areas (high contrast) are given more bits than those without texture (low contrast). Therefore, it will produce better visualization quality at the same bit rate. The post filter aims to provide value to areas with low contrast to improve image visualization and sharpness. Table 1 shows a comparison between the reconstructed image and the image after using a post filter.

Table 1. Comparison Performance Reconstructed and Postfilter

Approaches	Comparison Reconstructed and Postfilter		
	PSNR (dB)	*MS-SSIM*	*bpp*
AB	**32.33**	**0.88**	**0.14**
AB+ Post filter	**33.10**	**0.90**	**0.14**
Related Work [6]	31.81	0.85	0.32
Cheng 2020 [7]	32.28	0.88	0.17
JPEG [2]	32.28	0.87	0.31
Balle [9]	32.08	0.88	0.14

Fig. 9. visualizes a feature map that is inserted during the compression process, image (a) is the original image before the process of compression, picture (b) is a feature map of the image after entering the convolutional phase, and the last is a picture (c) is an allocation of bits from the image that has been compressed.



a b

Fig. 9. Visualization results of Attention Blok (AB) regarding kodim 23.png kodim 19.png from KODAK Dataset (a) original image, (b) latent, and (c) Allocated bits in train model.

The method we employed performed much better than the existing related works [2], [6], [7], [9], as can be seen in Table 1 above, which shows the average results of data comparison in the KODAK dataset. An increase in the PSNR value occurred after post-filtering had been applied. Similarly, the resulting compression was also better than others, with a bpp value of 0.14 for the experiment at λ=0.001. Qualitative performance comparison Kodim 15, and Kodim 14 of our reconstructed images as show in Fig. 10.

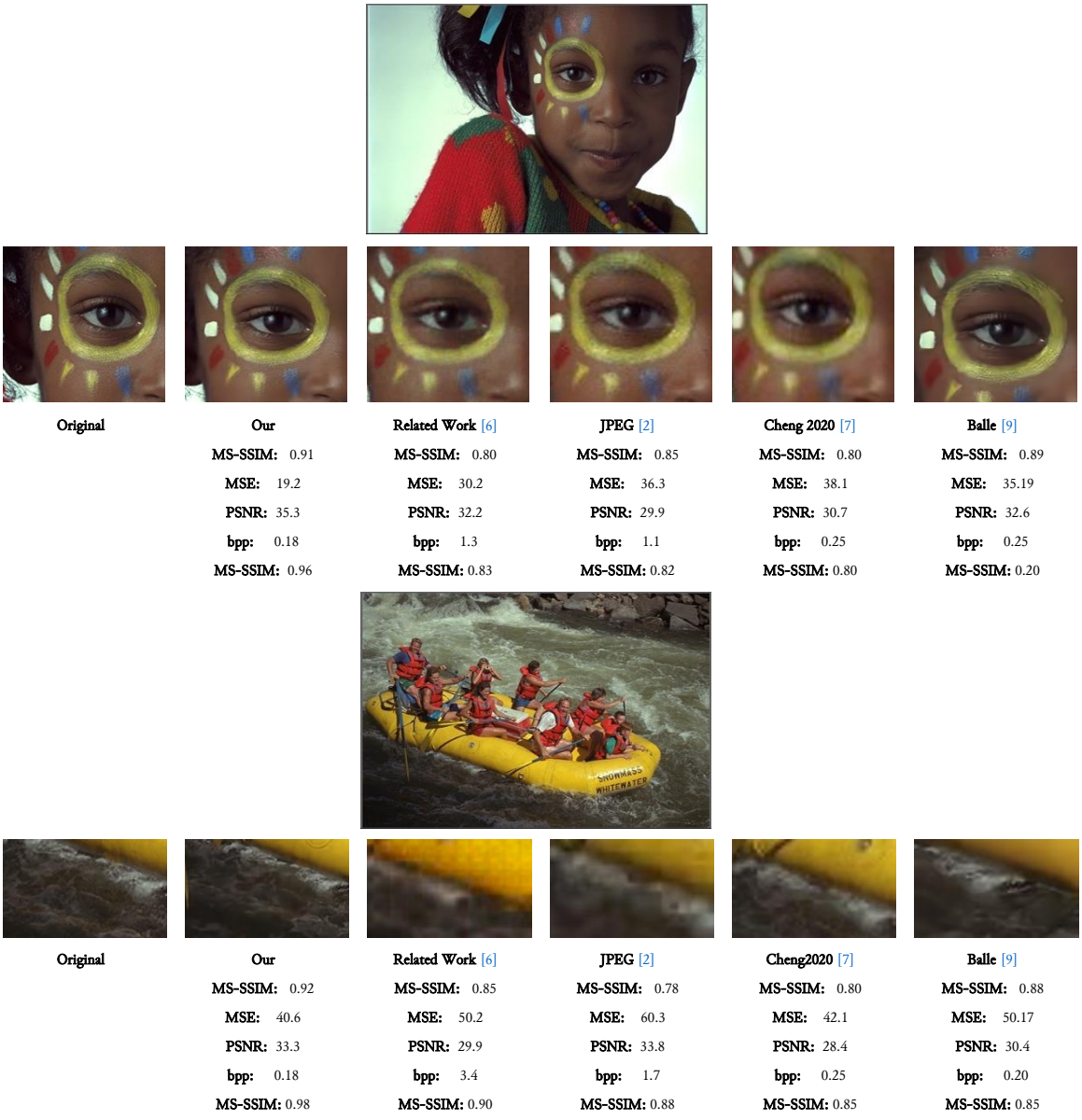


Fig. 10. Qualitative performance comparison Kodim 15, and Kodim 14 of our reconstructed images related work [2], [6], [9], [10] These images are taken from KODAK dataset.

We trained the model with several different lambda tests λ, including (0.01, 0.03, 0.20, 0.40, 0.80, 1.0), to estimate the resulting compression level and the effectiveness of the proposed model. In Table 1, the performance of the obtained model has nearly the same compression density in testing at (λ = 0.01-0.20). This is because, in testing using lambda with low values, the model tends to maintain the values of dominant pixels and ignore the values of non-dominant pixels, causing a loss of information from the model. Using an attention block in our model can help improve the values of dominant pixels, thus outperforming other models at low lambda λ values, minimizing the loss of important information in the image. Based on the results obtained, the PSNR and MS-SSIM values for the baseline balle model [9] were found to be PSNR 32.08 dB and MS-SSIM: 0.88 with bpp: 0.14. Meanwhile, the test results on the model [2], [6] have poor coding efficiency, as seen with the increasing lambda used. Our proposed model was able to outperform other models [2], [6], [7], [9] with the coding efficiency obtained in the test using a low lambda value of bpp 0.14 and PSNR 32.22 as well as MS-SSIM 0.8. Additionally, the

use of post-filtering can improve the quality of the obtained reconstructed image, resulting in a better PSNR of 33.10 dB and MS-SSIM of 0.9, outperforming other models. In Table 2, in the section of the results that we tabulated, the coding efficiency level obtained in our model was able to surpass almost all the SOTA, with an average difference of -33.34% for those using the Attention Module and -45.62% for the post-filtering module combined with AB. Therefore, it can be concluded that the use of attention modules in the compression process can improve coding efficiency while retaining important features in the image, and the effect of post-filtering can help improve the quality of the resulting image to approach the original image.

Table 2. Comparison BD-rate with SOTA

Dataset Kodak						BD-rate %	
	Anchor	*Balle [9]*	*Related work [6]*	*JPEG [2]*	*Cheng 2020 [7]*	*AB our*	*AB+Post Filter our*
PSNR	Balle [10]	0.00%	9.63%	163.28%	-8.06%	12.67%	**-28.16%**
	Related work [9]	-48.99%	0.00%	-32.09%	-52.71%	**-55.02%**	**-62.57%**
	JPEG [2]	-62.02%	-24.29%	0.00%	-66.15%	**-64.29%**	**-71.09%**
	Cheng 2020 [6]	8.77%	111.46%	111.46%	0.00%	**-5.38%**	**-20.66%**

4. Conclusion

This research paper uses hyperparameters to introduce a deep learning-based variational autoencoder image compression model. In our study, we introduce two new blocks to enhance encoding efficiency and improve reconstruction results: Attention Block (AB) and Post-Filtering. Several other supporting components such as Generalized Divisive Normalization (GDN), are also utilized. In our experiments, the model is tested with various lambda values to estimate the proposed model's compression level and effectiveness. The test results indicate that low lambda tends to preserve dominant pixel values but may sacrifice important information from the image. To address this, our model's attention blocks have been shown to enhance dominant pixel values, thereby minimizing the loss of crucial information in the image. The proposed model also demonstrates improved encoding efficiency and reconstruction image quality compared to the baseline model. Furthermore, post-filtering has proven to enhance image quality after the reconstruction process. Our model outperforms state-of-the-art (SOTA) models in terms of encoding efficiency, with a significant average difference. Therefore, it can be concluded that adding attention blocks in the compression process and using post-filtering can enhance encoding efficiency while preserving important features in the image and improving the quality of the reconstructed image. Nevertheless, this research still has its limitations. While the attention mechanism enhances quality only in dominant pixels, the coding efficiency achieved is significantly better. Moreover, the addition of blocks affects the computational process. In our subsequent research, we aim to optimize the attention blocks' shortcomings and develop a model that can reduce the overall complexity level.

Acknowledgment

We would like to thank to Hanbat National University Korea and the board management of Telkom University Jakarta who have facilitated us to accomplish our great work collaboration along this year without many obstacles. We expect that everyone finds it beneficial to reference this work.

Declarations

Author contribution. A.A.W., N.R.F.R; methodology, N.R.F.R; validation, A.A.W., P.D. and M.R; formal analysis, A.A.W, P.D and M.R; investigation, N.R.F.R; writing— original draft preparation,

N.R.F.R; writing—review and editing, A.A.W., P.D and M.R; supervision, P.D and M.R; funding acquisition, A.A.W., P.D and M.R. All authors have read and agreed to the published version of the manuscript.

Funding statement. None of the authors have received funding or grants from any institution or funding body for the research.

Conflict of interest. The authors declare no conflict of interest.

Additional information. No additional information is available for this paper.

References

[1] M. A. Rahman and M. Hamada, "Lossless Image Compression Techniques: A State-of-the-Art Survey," *Symmetry (Basel).*, vol. 11, no. 10, p. 1274, Oct. 2019, doi: 10.3390/sym11101274.

[2] M. Al-Ani, M. Shaban AL-Ani, and F. Hammadi Awad, "The Jpeg Image Compression Algorithm," *Int. J. Adv. Eng. Technol.*, vol. 6, no. December, pp. 1055–1062, 2013, [Online]. Available at: https://www.researchgate.net/publication/268523100.

[3] D. S. Taubman and M. W. Marcellin, "JPEG2000: standard for interactive imaging," *Proc. IEEE*, vol. 90, no. 8, pp. 1336–1357, Aug. 2002, doi: 10.1109/JPROC.2002.800725.

[4] Z. Jin, M. Z. Iqbal, W. Zou, X. Li, and E. Steinbach, "Dual-Stream Multi-Path Recursive Residual Network for JPEG Image Compression Artifacts Reduction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 2, pp. 467–479, Feb. 2021, doi: 10.1109/TCSVT.2020.2982174.

[5] W. Cui *et al.*, "Convolutional Neural Networks Based Intra Prediction for HEVC," in *2017 Data Compression Conference (DCC)*, Apr. 2017, vol. Part F1277, pp. 436–436, doi: 10.1109/DCC.2017.53.

[6] A. Tawfik *et al.*, "A Generic Real Time Autoencoder-Based Lossy Image Compression," in *2022 5th International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*, Dec. 2022, pp. 1–6, doi: 10.1109/ICCSPA55860.2022.10019047.

[7] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Deep Convolutional AutoEncoder-based Lossy Image Compression," in *2018 Picture Coding Symposium (PCS)*, Jun. 2018, pp. 253–257, doi: 10.1109/PCS.2018.8456308.

[8] N. Johnston *et al.*, "Improved Lossy Image Compression with Priming and Spatially Adaptive Bit Rates for Recurrent Networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2018, pp. 4385–4393, doi: 10.1109/CVPR.2018.00461.

[9] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," *6th Int. Conf. Learn. Represent. ICLR 2018 - Conf. Track Proc.*, pp. 1–23, Feb. 2018. [Online]. Available at: https://arxiv.org/abs/1802.01436v2.

[10] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Learned Image Compression With Discretized Gaussian Mixture Likelihoods and Attention Modules," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2020, pp. 7936–7945, doi: 10.1109/CVPR42600.2020.00796.

[11] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional Block Attention Module," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11211 LNCS, Springer Verlag, 2018, pp. 3–19, doi: 10.1007/978-3-030-01234-2_1.

[12] M. Wang, S. Wan, H. Gong, Y. Yu, and Y. Liu, "An Integrated CNN-based Post Processing Filter For Intra Frame in Versatile Video Coding," in *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, Nov. 2019, pp. 1573–1577, doi: 10.1109/APSIPAASC47483.2019.9023240.

[13] A. A. Jeny, M. B. Islam, M. S. Junayed, and D. Das, "Improving Image Compression With Adjacent Attention and Refinement Block," *IEEE Access*, vol. 11, pp. 17613–17625, 2023, doi: 10.1109/ACCESS.2022.3195295.

[14] W. Li, W. Sun, Y. Zhao, Z. Yuan, and Y. Liu, "Deep Image Compression with Residual Learning," *Appl. Sci.*, vol. 10, no. 11, p. 4023, Jun. 2020, doi: 10.3390/app10114023.

[15] B. Bross *et al.*, "Overview of the Versatile Video Coding (VVC) Standard and its Applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 3736–3764, Oct. 2021, doi: 10.1109/TCSVT.2021.3101953.

[16] J. Wu, J. Ma, F. Liang, W. Dong, G. Shi, and W. Lin, "End-to-End Blind Image Quality Prediction With Cascaded Deep Neural Network," *IEEE Trans. Image Process.*, vol. 29, pp. 7414–7426, 2020, doi: 10.1109/TIP.2020.3002478.

[17] X. Zhang and X. Wu, "Ultra High Fidelity Deep Image Decompression With l ∞ -Constrained Compression," *IEEE Trans. Image Process.*, vol. 30, pp. 963–975, 2021, doi: 10.1109/TIP.2020.3040074.

[18] A. Said, *Introduction to Arithmetic Coding - Theory and Practice*. p. 1-63, 2023. [Online]. Available at: https://arxiv.org/abs/2302.00819v1.

[19] G. G. Langdon, "An Introduction to Arithmetic Coding," *IBM J. Res. Dev.*, vol. 28, no. 2, pp. 135–149, Mar. 1984, doi: 10.1147/rd.282.0135.

[20] M. U. Hassan, M. H. Rehmani, and J. Chen, "Huff-DP: Huffman Coding based Differential Privacy Mechanism for Real-Time Data," *arXiv*, pp. 1–12, Jan. 2023. [Online]. Available at: https://arxiv.org/abs/2301.10395v1.

[21] L. Liu, T. Chen, H. Liu, S. Pu, L. Wang, and Q. Shen, "2C-Net: integrate image compression and classification via deep neural network," *Multimed. Syst.*, vol. 29, no. 3, pp. 945–959, Jun. 2023, doi: 10.1007/s00530-022-01026-1.

[22] Y. Wu, Z. Qi, H. Zheng, L. Tao, and W. Gao, "Deep Image Compression with Latent Optimization and Piece-wise Quantization Approximation," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Jun. 2021, pp. 1926–1930, doi: 10.1109/CVPRW53098.2021.00219.

[23] M. Song, J. Choi, and B. Han, "Variable-Rate Deep Image Compression through Spatially-Adaptive Feature Transform," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2021, pp. 2360–2369, doi: 10.1109/ICCV48922.2021.00238.

[24] L. Zhao, H. Bai, A. Wang, and Y. Zhao, "Multiple Description Convolutional Neural Networks for Image Compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 8, pp. 2494–2508, Aug. 2019, doi: 10.1109/TCSVT.2018.2867067.

[25] I. Schiopu and A. Munteanu, "Deep-Learning based Lossless Image Coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 7, pp. 1–1, Jul. 2020, doi: 10.1109/TCSVT.2019.2909821.

[26] M. Wang *et al.*, "End-to-end Image Compression with Swin-Transformer," in *2022 IEEE International Conference on Visual Communications and Image Processing (VCIP)*, Dec. 2022, pp. 1–5, doi: 10.1109/VCIP56404.2022.10008895.

[27] B. Xu, N. Wang, H. Kong, T. Chen, and M. Li, "Empirical Evaluation of Rectified Activations in Convolutional Network," *arXiv*, pp. 1–5, May 2015. [Online]. Available at: https://arxiv.org/abs/1505.00853v2.

[28] Y. Bai, "RELU-Function and Derived Function Review," *SHS Web Conf.*, vol. 144, p. 02006, Aug. 2022, doi: 10.1051/shsconf/202214402006.

[29] "CLIC · Challenge on Learned Image Compression," 2024. [Online]. Available at: https://compression.cc/.

[30] V. N. V. Satya Prakash, K. Satya Prasad, and T. Jaya Chandra Prasad, "Color image demosaicing using sparse based radial basis function network," *Alexandria Eng. J.*, vol. 56, no. 4, pp. 477–483, Dec. 2017, doi: 10.1016/j.aej.2016.08.032.

[31] D. P. Kingma and J. L. Ba, "Adam: A Method for Stochastic Optimization," *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR, pp. 1–15, Dec. 22, 2014. [Online]. Available at: https://arxiv.org/abs/1412.6980v9.

[32] G. Zhai and X. Min, "Perceptual image quality assessment: a survey," *Sci. China Inf. Sci.*, vol. 63, no. 11, p. 211301, Nov. 2020, doi: 10.1007/s11432-019-2757-1.

[33] D. Image and P. Laboratories, *Digital Image Processing Laboratory : Image Restoration Minimum Mean Square Error (MMSE) Linear Fil- ters*, no. 765. pp. 1-4, 2011. [Online]. Available at: https://engineering.purdue.edu/~bouman/grad-labs/Image-Restoration/pdf/lab.pdf.

[34] S. Fraihat and M. A. Al-Betar, "A novel lossy image compression algorithm using multi-models stacked AutoEncoders," *Array*, vol. 19, p. 100314, Sep. 2023, doi: 10.1016/j.array.2023.100314.

[35] A. Hore and D. Ziou, "Image Quality Metrics: PSNR vs. SSIM," in *2010 20th International Conference on Pattern Recognition*, Aug. 2010, pp. 2366–2369, doi: 10.1109/ICPR.2010.579.