2023 IEEE International Conference on Visual Communications and Image Processing (VCIP) | 979-8-3503-5985-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/VCIP59821.2023.10402793

MobileViT-GAN: A Generative Model for Low Bitrate Image Coding

Yifei Pei, Ying Liu, Nam Ling
Department of Computer Science and Engineering, Santa Clara University, Santa Clara, CA, USA

***Abstract*— Effective image coding techniques are crucial for digital image storage and transmission. Traditional methods struggle to maintain high visual quality at low bitrates. In this paper, we present MobileViT-GAN, a novel generative adversarial network (GAN) architecture for low bitrate image compression. We propose a lightweight transformer-based discriminator to improve coding performance, compared to convolutional neural network-based discriminators. Additionally, we introduce a smoothness loss function to mitigate artifacts in decoded images, further improving visual quality in low bitrate image coding. We evaluate our proposed method against traditional and state-of-the-art GAN-based image compression techniques, showcasing its superiority in terms of compression ratio and decoded image quality.**

Keywords*— *end-to-end compression, generative adversarial network, image coding, transformer, vision transformer

I. Introduction

Efficient image storage and transmission require effective compression techniques, especially in low bitrate scenarios such as video surveillance or bandwidth-limited applications. Traditional methods like JPEG [1], JPEG 2000 [2], and BPG [3] struggle to maintain visual quality at very low bitrates, whereas deep learning-based methods [4], [5], particularly generative adversarial networks (GANs) [6], exhibit improved performance in image compression tasks. Owing to GAN's ability to generate photorealistic images, employing a generator as encoder-decoder for low bitrate image compression results in better perceptual quality [7]. GAN-based approaches excel in this task as adaptive image representations allow the generator to create realistic images while the discriminator differentiates between original and generated images. Existing GAN-based image compression methods adopt convolutional neural network-based discriminators, which are good at learning local features. In recent years, the vision transformer (ViT) [9] has emerged and shown capability of learning global features for computer vision tasks. Therefore, it has the potential to enhance discriminators in GAN-based image compression frameworks.

In this paper, we propose MobileViT-GAN, a novel GAN-based low bitrate image coding method that incorporates a lightweight transformer MobileViT [8] as the discriminator [6]. Besides, we propose a smoothness loss function to mitigate artifacts in decoded images.

II. Background

A. GAN-based Image Compression Methods

Generative adversarial networks (GANs) [10] consist of a generator and a discriminator, engaging in adversarial training to learn rich image representations for image processing tasks. Several studies have explored GAN-based compression techniques. In [6], a least squares generative adversarial network (LS-GAN) was proposed, minimizing perceptual artifacts while maintaining low bitrates. HiFiC [11] used a conditional GAN for image compression, improving image reconstruction quality. A hinge GAN-GP and a simple entropy estimator were introduced in [12], achieving remarkable visual quality at low bitrates. These methods utilized convolutional neural networks (CNNs) in the discriminator, but CNNs' local receptive field makes it challenging to extract global features from images.

B. Vision Transformer and MobileViT

Transformers [13] demonstrate significant potential in computer vision tasks, with ViT [9] achieving top performance on various benchmarks. However, ViT's computational cost and memory requirements can be prohibitive for resource-constrained devices, particularly for high-resolution images. Reduced-size ViT models may perform worse than lightweight CNNs when tailored for mobile devices.

MobileViT [8] is a lightweight variant of ViT, using inverted bottleneck blocks [14] to minimize computational complexity while maintaining performance. With self-attention for compact global representation, MobileViT outperforms other efficient architectures like MobileNetV3 [15] and EfficientNet [16] in image classification and semantic segmentation. Therefore, we adopt MobileViT as the discriminator in our GAN-based image compression architecture, improving the coding performance for low bitrate image compression, while reducing the computational cost in the training stage.

III. Proposed Approach

Fig. 1 displays our proposed MobileViT-GAN architecture for low bitrate image compression, consisting of a generator for encoding/decoding images and a MobileViT-based discriminator to efficiently discern real (original $\mathbf{X}$) and fake (decoded $\tilde{\mathbf{X}}$) images. The discriminator provides feedback to the generator, refining its encoding and decoding processes for higher-quality image reconstruction. Additionally, we introduce a smoothness loss function when we train the generator to reduce artifacts in decoded images.

A. The MobileViT-Based Discriminator

Fig. 1 bottom section shows our proposed discriminator: MobileViT. It has two key components: The MobileViT module and the Inverted Residual Block (InvRB).

Fig. 2 depicts the MobileViT module, which consists of the following steps:

1. Input tensor ($\mathbf{Z}$): Represents the input feature map as $\mathbf{Z} \in \mathbb{R}^{H\times W\times C}$ with height H, width W, and channels C.
2. 3×3 convolutional layer: Captures local spatial information from $\mathbf{Z}$.
3. 1×1 convolutional layer: Projects $\mathbf{Z}$ into a higher dimensional space ($d > C$) to create $\mathbf{Z}_L \in \mathbb{R}^{H\times W\times d}$.
4. Unfolding $\mathbf{Z}_L$: Transforms $\mathbf{Z}_L$ into non-overlapping

flattened patches $\mathbf{Z}_U \in \mathbb{R}^{P\times N\times d}$, where $P = wh$ and $N = (HW)/P$ is the number of patches. Here, w and h represent the width and height of the non-overlapping patches, respectively.

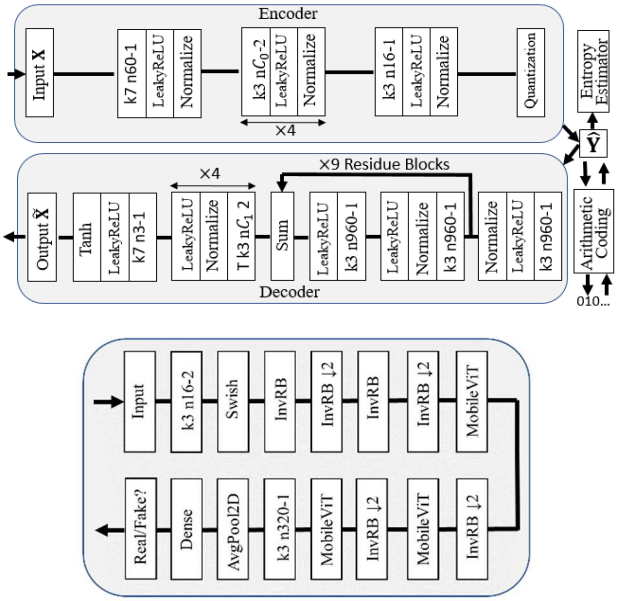

Fig. 1. MobileViT-GAN architecture. The top section depicts the generator (encoder-decoder) and entropy estimator, while the bottom section shows the discriminator: MobileViT. "k3 n$C_0 - 2$" denotes a convolution layer with a 3×3 kernel, C_0 output channels, and a stride of 2; C_0 takes values of 120, 240, 480, and 960 for four convolutional layers. "T k3 n$C_1 - 2$" represents a transposed convolution layer with C_1 output channels, where C_1 is 480, 240, 120, and 60 for the four respective layers. "↓2" in the Inverted Residue Block (InvRB) indicates spatial down-sampling by a factor of 2.

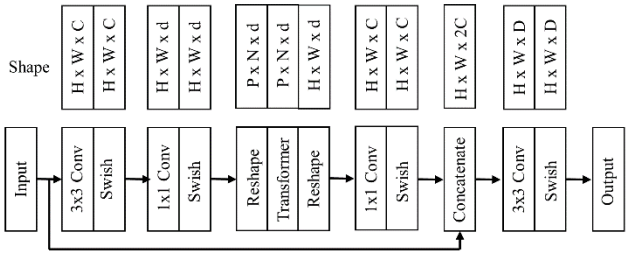

Fig. 2. MobileViT module architecture: output tensor shapes are presented above each respective block.

5. Transformer layer: Encodes inter-patch relationships, yielding $\mathbf{Z}_G \in \mathbb{R}^{P\times N\times d}$.
6. Folding $\mathbf{Z}_G$: Recovers $\mathbf{Z}_F \in \mathbb{R}^{H\times W\times d}$ from $\mathbf{Z}_G$.
7. 1x1 convolutional layer: Projects $\mathbf{Z}_F$ back into a lower $C-$ dimensional space.
8. Concatenation: Combines $\mathbf{Z}_F$ with the input tensor $\mathbf{Z}$.
9. 3×3 convolutional layer: Fuses concatenated features to generate the output tensor.

The skip connection in the MobileViT module, which combines CNNs and transformers, effectively captures local spatial information with convolutional layers and encodes global inter-patch relationships using the transformer layer.

The Inverted Residual Block is shown in Fig. 3. It employs a depthwise separable convolution layer with a stride of either 1 or 2. This layer extracts essential features separately across input

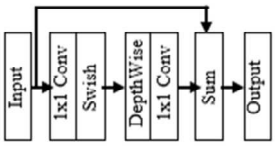

Fig. 3. Inverted Residue Block module architecture.

channels from the local receptive field of the tensor $\mathbf{Z}_L$, producing the output tensor $\mathbf{Z}_F$, which significantly reduces the computational complexity from $\mathcal{O}(H_{out} \times W_{out} \times C_{in} \times C_{out} \times K_h \times K_w)$ to $\mathcal{O}(H_{out} \times W_{out} \times C_{in} \times K_h \times K_w)$, compared to a standard convolutional layer, where H_{out} and W_{out} are height and width of the output feature map, C_{in} and C_{out} are the number of input and output channels, and K_h and K_w are the kernel height and width. A residue connection is used to check if the dimensions of $\mathbf{Z}_F$ and $\mathbf{Z}_L$ match. If not, the residual connection is skipped, and $\mathbf{Z}_L$ becomes the output tensor. Otherwise, the tensors $\mathbf{Z}_L$ and $\mathbf{Z}_F$ are added to form the output tensor.

For the MobileViT discriminator, we use the Swish activation function [17], which can lead to faster convergence during training and is defined as:

$$\text{Swish}(x') = x' \cdot \sigma(\beta x'), \tag{1}$$

where x' is the input feature, β is a trainable parameter, and σ is the standard sigmoid function.

B. Smoothness Loss for Low Bitrate Image Coding

To remove artifacts in decoded images, we introduce a gradient loss in the training loss when we train the generator. This helps to improve spatial smoothness and perceptual quality in decoded images. The gradient loss is defined as:

$$L_{\text{smooth}} = \frac{1}{H_I \times W_I} \sum_{j,k} \left\| \tilde{\mathbf{X}}_{j,k} - \tilde{\mathbf{X}}_{j+\delta,k+\delta} \right\|^2, \tag{2}$$

where H_I and W_I are the height and width of the input image, j and k represent the spatial coordinates of a pixel within a decoded image, and δ represents the horizontal and vertical offsets between the compared pixels. We set δ to 1, comparing each pixel intensity with its immediate neighbor in both horizontal and vertical directions.

C. Quantization and Entropy Coding

We utilize the soft-to-hard quantization from [18] to map each encoder's output feature element, $\mathbf{Y}_{ijk}$, with the i^{th} channel and (j, k) spatial coordinates to its corresponding integer quantization center within the finite set $S = \{-2, -1, 0, 1, 2\}$, obtaining the quantized feature $\bar{\mathbf{Y}}_{ijk}$. Due to the non-differentiability of the nearest neighbor method, we apply differentiable soft quantization to enable backpropagation within the neural network:

$$\overline{\mathbf{Y}}_{ijk} = \sum_{s_l \in S} \frac{\exp\left(-\sigma \left| \mathbf{Y}_{ijk} - s_l \right|\right)}{\sum_{s_h \in S} \exp\left(-\sigma \left| \mathbf{Y}_{ijk} - s_h \right|\right)} s_l. \tag{3}$$

Here, σ is the sigmoid function. Throughout the training process, the quantized feature is obtained using:

$$\widehat{\mathbf{Y}}_{ijk} = \text{stop\_gradient}\left(\widetilde{\mathbf{Y}}_{ijk} - \overline{\mathbf{Y}}_{ijk}\right) + \overline{\mathbf{Y}}_{ijk}$$

The stop gradient function ensures that only the gradient of $\overline{\mathbf{Y}}_{ijk}$ is propagated for updating network parameters during the backward pass, while the gradient of $\left(\widetilde{\mathbf{Y}}_{ijk} - \overline{\mathbf{Y}}_{ijk}\right)$ is not propagated.

To optimize the entropy of quantized features, we use the entropy estimator from [12], assuming the encoder's final layer output, $\mathbf{Y}_{ijk}$, follows a normal distribution $\mathcal{N}(\beta_i, \alpha_i^2)$, where α_i and β_i are learnable per-channel offsets in the channel normalization. The estimated entropy of each quantized feature element $\widetilde{\mathbf{Y}}_{ijk}$ is obtained by calculating the expected value of the negative logarithm of the probability density function:

$$L_{\text{entropy}} = \mathbb{E}\left[-\log_2 p\left(\widetilde{\mathbf{Y}}_{ijk}\right)\right] = \begin{cases} \mathbb{E}\left\{-\log_2 \Phi\left(\widetilde{\mathbf{Y}}_{ijk} + 0.5\right)\right\}, & \widetilde{\mathbf{Y}}_{ijk} = -2, \\ \mathbb{E}\left\{-\log_2\left[1 - \Phi\left(\widetilde{\mathbf{Y}}_{ijk} - 0.5\right)\right]\right\}, & \widetilde{\mathbf{Y}}_{ijk} = 2, \\ \mathbb{E}\left\{-\log_2\left[\Phi\left(\widetilde{\mathbf{Y}}_{ijk} + 0.5\right) - \Phi\left(\widetilde{\mathbf{Y}}_{ijk} - 0.5\right)\right]\right\}, & \text{otherwise.} \end{cases} \quad (4)$$

In this equation, $\Phi(\cdot)$ is the cumulative distribution function (CDF) of the standard normal distribution.

D. Our Loss Functions

We train the discriminator and generator by adopting the Hinge GAN-GP [12] strategy for efficient training.

For the discriminator, we use equation (5) proposed by [12]:

$$\min_D \mathbb{E}\left[\max\left(0, 1 - f_D(\mathbf{X})\right)\right] + \mathbb{E}\left[\max\left(0, 1 + f_D\left(\widetilde{\mathbf{X}}\right)\right)\right] + \lambda_1 \mathbb{E}\left[\left(\left\|\nabla_{\widehat{\mathbf{X}}} f_D\left(\widehat{\mathbf{X}}\right)\right\|_2 - 1\right)^2\right], \quad (5)$$

where $\mathbf{X}$ is the original image, $\widetilde{\mathbf{X}}$ is the decoded image, $\widehat{\mathbf{X}}$ is the interpolation between the real image $\mathbf{X}$ and the decoded image $\widetilde{\mathbf{X}}$, λ_1 is the gradient penalty weight. In (5), $f_D(\cdot)$ is the discriminator. While $f_D(\mathbf{X})$ is anticipated to approach 1, $f_D\left(\widetilde{\mathbf{X}}\right)$ is expected to approach -1. We only penalize errors when $f_D(\mathbf{X}) < 1$ or $f_D\left(\widetilde{\mathbf{X}}\right) > -1$, avoiding penalties for correct classifications, which makes the discriminator training more effective.

For the generator, we use:

$$\min_G L_g^{\text{total}} = \min_G\left(L_g + \lambda_2 L_{\text{content}} + \lambda_3 L_{\text{entropy}} + \lambda_4 L_{\text{smooth}}\right), \quad (6)$$

where $L_g = \mathbb{E}[-f_D\left(\widetilde{\mathbf{X}}\right)]$ is the adversarial loss, $L_{\text{content}} = \mathbb{E}[(1-\alpha)\text{MAE}\left(\widetilde{\mathbf{X}}, \mathbf{X}\right) - \alpha\text{MS-SSIM}\left(\widetilde{\mathbf{X}}, \mathbf{X}\right)]$ is the content loss, combining the mean absolution error (MAE) and the multi-scale structural similarity (MS-SSIM) between the original image $\mathbf{X}$ and the decoded image $\widetilde{\mathbf{X}}$ [12]. Besides, L_{entropy} is the entropy loss provided in (4), and L_{smooth} is defined in (2).

IV. Experimental Results

We trained GAN models on 235,679 distinct 256×256 image patches, obtained from 118,287 images in the COCO dataset [19], using the Adam optimizer [20] with a 0.0001 learning rate, a batch size of 24, and 40 epochs. To evaluate these models' performance, we tested them against all 24 images of 512×768 resolution from the Kodak dataset [21], which is commonly used to evaluate BPG [3] and learned image compression models [4, 5, 6, 11, 12].

Since our focus was on perceptual quality, we employed the Fréchet Inception Distance (FID) score [22] and the MS-SSIM as evaluation metrics for the decoded images. The FID score is determined by computing the mean and covariance matrix of the features extracted from a pretrained Inception network for both the original and decoded images. A lower FID score indicates that the distribution of the decoded images is more similar to the distribution of the original images.

For the GAN models, we configured the compressed feature dimensionality to be $16 \times 16 \times 16$ for height, width, and channel number, respectively. In our proposed model, we set the hyper-parameters in the discriminator loss function (5) and the generator loss function (6) as $\lambda_1 = 10, \lambda_2 = 100, \lambda_3 = 1$, $\lambda_4 = 5$, and α=0.84. These settings resulted in good visual quality for decoded images, while maintaining a low bitrate.

Fig. 5 displays visual qualities of decoded test images for BPG, LS-GAN, HiFiC, Hinge GAN-GP, and our proposed MobileViT-GAN. Our model provides the best visual quality, preserving more texture details, compared to other models. LS-GAN poorly reconstructs textures of hats and struggles with coastal areas, and its decoded images exhibit unfaithful colors. BPG's decoded images show block artifacts in cloud and coastal regions, affecting visual quality. HiFiC's decoded images are less sharp in cap and coastal areas compared to our model. Hinge GAN-GP fails to accurately reconstruct the railing area in the second decoded image.

Table 1 shows the average bits per pixel (bpp), FID, and MS-SSIM scores for all methods in Fig. 5, using 24 Kodak test images. Our proposed MobileViT-GAN attains the best bpp, MS-SSIM, and FID scores. Furthermore, our model saves 6.2%, 30.54%, 40.01%, 2.09% in bitrates compared to BPG, LS-GAN, HiFiC, and Hinge GAN-GP, respectively, and reducing bitrates for these models would result in FID and MS-SSIM scores that are even worse compared to our proposed MobileViT-GAN.

We conducted an ablation study to demonstrate the effectiveness of our proposed components: the MobileViT discriminator and the smoothness loss function. Fig. 6 displays the result, which is an enlarged patch of a Kodak image decoded by our generator under three scenarios at approximately 0.090 bpp: without a discriminator, without the smoothness loss, and with both the discriminator and smoothness loss. The decoded image generated by the model that includes both the discriminator and smoothness loss exhibits clearer texture and appears more realistic compared to the other two scenarios, which indicates the effectiveness of our proposed discriminator and smoothness loss function.

We also compare the number of parameters and the floating-point operations (FLOPs) for a single image patch of the proposed MobileViT discriminator with the CNN-based discriminators in other GAN-based image compression models. The MobileViT discriminator has 1,301,249 parameters and 0.63 giga FLOPs, while the discriminator of LS-GAN has 2,766,529 parameters and 4.37 giga FLOPs. Similarly, the discriminator of HiFiC has 2,777,985 parameters and 3.73 giga

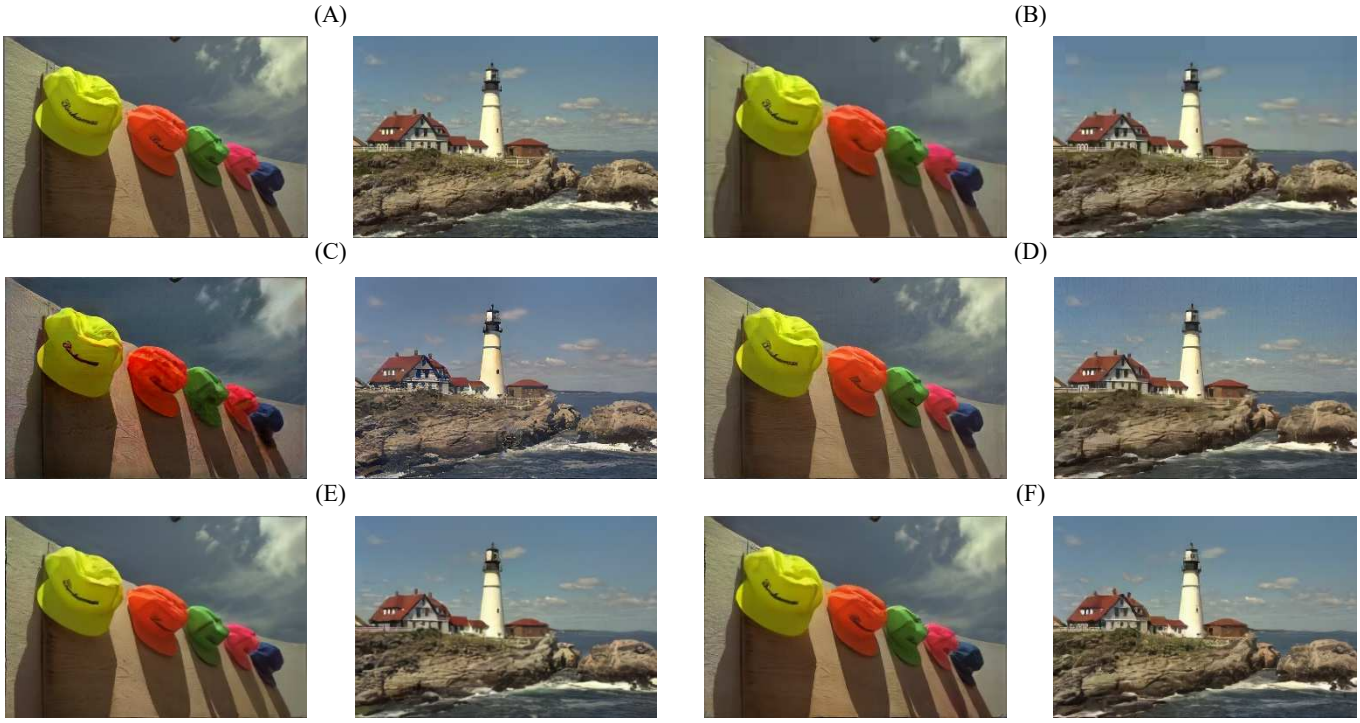

Fig. 5. Sample images of Kodak dataset [21]. (A) Original; (B) BPG [3]; (C) LS-GAN [6]; (D) HiFiC [11]; (E) Hinge GAN-GP [12]; (F) **The Proposed MobileViT-GAN.**

Table 1. Comparison of given approaches on 24 Kodak images [20]. The best value of each metric is marked in red.

	BPG	LS-GAN	HiFiC	Hinge GAN-GP	MobileViT-GAN
Average bpp	0.0948	0.1280	0.1501	0.0908	**0.0889**
Average MS-SSIM	0.8932	0.8401	0.8802	0.8998	**0.9012**
FID	107.40	98.07	55.27	67.44	**53.14**

Fig. 6. An enlarged patch of the kodim21 image from the Kodak dataset [21], reconstructed by our MobileViT-GAN's generator. Left to right: original; **w/o** discriminator; **w/o** smoothness loss; **with** discriminator and smoothness loss.

FLOPs, and the discriminator of Hinge GAN-GP has 3,049,857 parameters and 6.67 giga FLOPs. We observe that MobileViT is the most lightweight discriminator, but it still helps our GAN-based image compression model achieve the best quality of decoded images compared to the other three GAN-based methods. This demonstrates that transformer structures help to build a more effective discriminator compared to CNN structures. Additionally, the lightweight discriminator also helps to save training costs. It is important to mention that all GAN models have very close parameter numbers and FLOPs for their generators: MobileViT-GAN has 160,672,403 parameters and 96.08 giga FLOPs for the generator; LS-GAN has 160,672,371 parameters and 95.97 giga FLOPs for the generator; HiFiC has 160,603,239 parameters and 96.08 giga FLOPs for the generator; Hinge GAN-GP has 160,672,403 parameters and 96.08 giga FLOPs for the generator. The closely matched parameters and FLOPs for the generators of these GAN models indicate the fairness of comparison, and also demonstrate that our proposed MobileViT discriminator is effective in improving the performance of our generator.

V. Conclusion

We propose MobileViT-GAN, a generative adversarial network for low bitrate image coding, with a lightweight transformer-based discriminator and a smoothness loss function. Our model delivers superior visual quality while maintaining a highly efficient discriminator with fewer parameters and lower computational complexity in terms of FLOPs, compared to existing GAN-based image compression models. Future work will focus on developing efficient transformer structures for the generator. We also consider applying MobileViT-GAN to image compressed sensing [23].

REFERENCES

[1] W. B. Pennebaker and J. L. Mitchell, "JPEG: Still image data compression standard," in Springer Science & Business Media, 1992.

[2] M. Rabbani and R. Joshi, "An overview of the jpeg 2000 still image compression standard," in Signal processing: Image Communication, vol. 17, no. 1, pp. 3-48, 2002.

[3] F. Bellard, "BPG image format," in https://bellard.org/bpg, 2015.

[4] J. Ballé, V.Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in arXiv preprint arXiv:1611.01704, 2016.

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

[6] 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.

[7] N. Ling, C.-C. J. Kuo, G. J. Sullivan, D. Xu, S. Lin, H.-H. Huang, W.-H. Peng, J. Liu et al, "The future of video coding," in APSIPA Transactions on Signal and Information Processing, vol. 11, no. 1, 2022.

[8] S. Mehta and M. Rastegari, "Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer," in arXiv preprint arXiv:2110.02178, 2021.

[9] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in arXiv preprint arXiv:2010.11929, 2020.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in Advances in Neural Information Processing Systems, vol. 27, 2014.

[11] F. Mentzer, G. D. Toderici, M. Tschannen, and E. Agustsson, "High-fidelity generative image compression," in Advances in Neural Information Processing Systems, vol. 33, pp. 11 913-11 924, 2020.

[12] Y. Pei, Y. Liu, N. Ling, Y. X. Ren, and L. Z. Liu, "An end-to-end deep generative network for low bitrate image coding," in 2023 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, 2023, pp. 1-5.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in Advances in Neural Information Processing Systems, vol. 30, 2017.

[14] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4510–4520.

[15] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan et al., "Searching for mobilenetv3," in Proceedings of the IEEE/CVF International Conference on Computer vision, 2019, pp. 1314–1324.

[16] Tan. M. Tan and V. Q. Le, "Rethinking model scaling for convolutional neural networks," in Proceedings of the 36th International Conference on Machine Learning. New York: IEEE, vol. 97, 2019, pp. 6105–6114.

[17] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," in arXiv preprint arXiv:1710.05941, 2017.

[18] E. Agustsson, F. Mentzer, M. Tschannen, L. Cavigelli, R. Timofte, L. Benini, and L. V. Gool, "Soft-to-hard vector quantization for end-to-end learning compressible representations," in Advances in neural information processing systems, vol. 30, 2017.

[19] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick, "Microsoft coco: Common objects in context," in European conference on computer vision. Springer, 2014, pp. 740-755.

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

[21] R. Franzen, "Kodak lossless true color image suite," source: https://r0k.us/graphics/kodak, vol. 4, no. 2, 1999.

[22] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in Advances in Neural Information Processing Systems, vol.30, 2017.

[23] Y. Pei, Y. Liu, and N. Ling, "Deep learning for block-level compressive video sensing," 2020 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, 2020, pp. 1-5.