Enhanced Invertible Encoding for Learned Image Compression

Yueqi Xie*
HKUST

Ka Leong Cheng*
HKUST

Qifeng Chen
HKUST

ABSTRACT

Although deep learning based image compression methods have achieved promising progress these days, the performance of these methods still cannot match the latest compression standard Versatile Video Coding (VVC). Most of the recent developments focus on designing a more accurate and flexible entropy model that can better parameterize the distributions of the latent features. However, few efforts are devoted to structuring a better transformation between the image space and the latent feature space. In this paper, instead of employing previous autoencoder style networks to build this transformation, we propose an enhanced Invertible Encoding Network with invertible neural networks (INNs) to largely mitigate the information loss problem for better compression. Experimental results on the Kodak, CLIC, and Tecnick datasets show that our method outperforms the existing learned image compression methods and compression standards, including VVC (VTM 12.1), especially for high-resolution images. Our source code is available at https://github.com/xyq7/InvCompress.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; **Artificial intelligence**.

KEYWORDS

Image compression; Invertible neural networks

ACM Reference Format:
Yueqi Xie, Ka Leong Cheng, and Qifeng Chen. 2021. Enhanced Invertible Encoding for Learned Image Compression. In *Proceedings of the 29th ACM Int'l Conference on Multimedia (MM '21), Oct. 20–24, 2021, Virtual Event, China.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3474085.3475213

1 INTRODUCTION

Lossy image compression has been a fundamental and important research topic in media storage and transmission for decades. Classical image compression standards are usually based on handcrafted schemes, such as JPEG [45], JPEG2000 [37], WebP [17], BPG [10], and Versatile Video Coding (VVC) [24]. Some of them are widely used in practice. Recently, there is an increasing interest in learned image compression methods [6, 11, 29, 33, 34] considering their competitive performance. Generally, the recent VAE-based methods [6, 7] follow a process: an encoder transforms the original pixels $\mathbf{x}$ into lower-dimensional latent features $\mathbf{y}$ and then quantizes it to $\hat{\mathbf{y}}$, which can be losslessly compressed using entropy coding methods, like arithmetic coding [39, 48]. A jointly optimized decoder is utilized to transform $\hat{\mathbf{y}}$ back to the reconstructed original image $\hat{\mathbf{x}}$.

Most of the recent developments on this topic focus on the improvement of the entropy model. Ballé et al. [8] propose a variational image compression method with a scale hyperprior. Afterward, Minnen et al. [34] combine the context model [27] and an autoregressive model over the latent features. Cheng et al. [11] further improve the method with attention modules and employ discretized Gaussian mixture likelihoods to better parameterize the latent features. These recent improvements on entropy models greatly contribute to efficient compression. However, they usually model the transformation between the image space and the latent feature space using an autoencoder framework. Although autoencoders have strong capacities to select the important portions of the information for reconstruction, the neglected information during encoding is usually lost and unrecoverable for decoding.

To resolve the information loss problem, a possible idea is to integrate invertible neural networks (INNs) into the encoding-decoding process because INNs have the strictly invertible property to help preserve information. However, it is still nontrivial to leverage INNs for replacing the original encoder-decoder architecture. On the one hand, to ensure strict invertibility, the total pixel number in the output should be the same as the input pixel number. But it is intractable to quantize and compress such high-dimensional features [16] and hard to get desirable low bit rates for compression. The work [20] explores the usage of INNs to break AE limit but only gets good performance in high bpp range. For lower-bpp image compression with INNs, Wang et al. [46] take a subspace of the output to get lower-dimensional features, model lost information with a given distribution, and do sampling for the inverse passing. However, this strategy introduces unwanted errors and leads to unstable training, so they further introduce a knowledge distillation module to guide and stabilize the training of INNs, but it may lead to sub-optimal solutions. On the other hand, although the mathematical network design guarantees the invertibility of INNs, it makes INNs have limited nonlinear transformation capacity [13] compared to other encoder-decoder architectures.

Concerning these aspects, we propose an enhanced Invertible Encoding Network for image compression, which maintains a highly invertible structure based on INNs. Instead of using the unstable sampling mechanism [46], we propose an attentive channel squeeze layer to stabilize the training and to flexibly adjust the feature dimension for a lower bit rate. We also present a feature enhancement module with same-resolution transforms and residual connections to improve the network nonlinear representation capacity for better image information preservation. With our design, we can directly

*Both authors contributed equally.

MM '21, October 20–24, 2021, Virtual Event, China.

ACM ISBN 978-1-4503-8651-7/21/10…$15.00
https://doi.org/10.1145/3474085.3475213

integrate INNs to capture the lost information without relying on any sub-optimal guidance and boost performance.

Experimental results show that our proposed method outperforms the existing learned methods and traditional codecs on three widely-used datasets for image compression, including Kodak [12] and two high-resolution datasets, namely the CLIC Professional Validation dataset [43] and the Tecnick dataset [5]. Visual results demonstrate that our reconstructed images contain more details under similar BPP rates, beneficial from our highly invertible design. Further analysis of our proposed attentive channel squeeze layer and feature enhancement module verifies their functionality. The contributions of this paper can be summarized as follows:

- Unlike widely-used autoencoder style networks for learned image compression, we present an enhanced Invertible Encoding Network with an INN architecture for the transformation between image space and feature space to largely mitigate the information loss problem.
- The proposed method outperforms the existing learned and traditional image compression methods, including the latest VVC (VTM 12.1), especially on high-resolution datasets.
- We propose an attentive channel squeeze layer to resolve the unstable and sub-optimal training of the INN-based networks for image compression. We further leverage a feature enhancement module to improve the nonlinear representation capacity of our network.

2 RELATED WORK

2.1 Lossy Image Compression

Traditional methods. Lossy image compression has long been an important and fundamental topic in image processing. Traditional compression standards includes JPEG [45], JPEG2000 [37], WebP [17], Better Portable Graphics (BPG) [10] and Versatile Video Coding (VVC) [24]. Many of them are widely used in practice. Typically, they follow the pipeline of transformation, quantization, and entropy coding. The transformation is usually based on handcrafted modules with prior knowledge, such as discrete cosine transformation (DCT) [3] and discrete wavelet transform (DWT) [32]. The entropy coders include Huffman coder and some arithmetic coding methods [32, 39, 48]. Some modern standards like BPG and VVC further introduce intra prediction for better compression performance. However, since these traditional methods generally perform compression based on image blocks, their reconstructed images usually contain some limitations of the blocking effects.

Learned methods. Deep learning based methods have raised great interest recently with impressive performance, by employing neural networks instead of handcrafted rules for the nonlinear transformation learning between the image and the feature space.

Several recurrent neural network (RNN) based methods [23, 42, 44] progressively encode the residual information from the previous step to compress the image. However, these RNN-based models rely on binary representation at each iteration and cannot directly optimize the rate during training.

Another branch of the methods is based on variational autoencoders (VAEs). Several early works [1, 7, 41] solve the problem of non-differential quantization and estimation of bit rates, which lay the foundation for end-to-end optimization to minimize the estimated bit rates and the reconstructed image distortion. Afterward, the improvement is mainly in two directions. One direction is to build a more effective entropy model for rate estimation. The work [8] proposes a hyperprior entropy model to use additional bits to capture the distribution of the latent features. Some follow-up methods [27, 33, 34] integrate context factors to further reduce the spatial redundancy within the latent features. Also, 3D context entropy model [19], channel-wise models [35] and hierarchical entropy models [21, 34] are used to better extract correlations of latent features. Cheng et al. [11] propose Gaussian mixture likelihood to replace the original single Gaussian model to improve the accuracy. Another direction is to improve the VAE architecture. CNNs with generalized divisive normalization (GDN) layers [6, 7] achieve good performance for image compression. Attention mechanism and residual blocks [11, 29, 51, 52] are incorporated into the VAE architecture. Some other progress includes generative adversarial training [2, 38, 40], spatial RNNs [28] and multi-scale fusion [38].

Generally, the VAE-based methods account for a more significant proportion among all the existing learned methods for image compression, considering the great performance and stability of the encoder and decoder architecture. Still, they cannot explicitly solve the information loss problem during encoding, making the neglected information usually unrecoverable for decoding.

2.2 Invertible Neural Networks

Invertible neural networks (INNs) [13, 14, 26] are popular with three great properties of the design: (i) INNs maintain a bijective mapping between inputs and outputs; (ii) the bijective mapping is efficiently accessible; (iii) there exists a tractable Jacobian of the bijective mapping to compute posterior probabilities explicitly. Many recent works in different areas with INN architecture achieve better performance than those with autoencoder style frameworks, especially for the tasks with inherent invertibility.

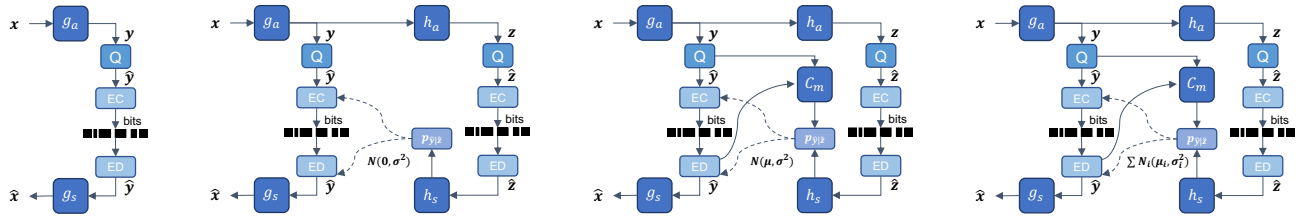
RealNVP [14] uses a multi-scale architecture with coupling layers and convolution operations to first deal with image processing tasks. Ardizzone et al. [4] demonstrate the effectiveness of INNs on both the synthetic data and two real-world applications in the medicine and astrophysics fields. Recently, many works start to use normalizing flow methods with exact likelihoods during training in generative tasks, where the key is to parametrize the distribution using INNs. SRFlow [31] uses a conditional INN architecture to better resolve the ill-posed problem of super-resolution compared to GAN-based methods. Pumarola et al. [36] propose a conditional generative flow model for Image and 3D Point Cloud generation. For the image rescaling task, Xiao et al. [49] use INNs to generate a bijective mapping between high-resolution images and low-resolution images with additional latent variables. Xing et al. [50] propose an invertible image signal processing pipeline based on INNs.

3 METHOD

3.1 Background

Figure 1 provides a high-level overview of general learned image compression models in the transform coding approach [18]. A baseline model (Figure 1a) is formulated as follows:

$$\mathbf{y} = g_a(\mathbf{x}),\ \hat{\mathbf{y}} = Q(\mathbf{y}),\ \hat{\mathbf{x}} = g_s(\hat{\mathbf{y}}). \tag{1}$$



(a) Baseline [6] **(b) Ballé et al. [8]** **(c) Minnen et al. [34]** **(d) Cheng et al. [11]**

Figure 1: The overview of existing learned compression methods. *EC* **and** *ED* **denote the encoder and decoder of the entropy model, respectively, and** C_m **denotes the context model.**

For encoding, a parametric analysis transform g_a encodes the image $\mathbf{x}$ into latent features $\mathbf{y}$, and $\mathbf{y}$ is then quantized to obtain the discrete latent features $\hat{\mathbf{y}}$, which is then losslessly compressed into bitstreams using entropy coding like arithmetic coding [39]. Note that since integer rounding is a fundamentally non-differentiable function, we use the method in [7] to approximately model the quantized latent features by adding a uniform noise $U(-0.5, 0.5)$ to $\mathbf{y}$ during training. For simplicity, we use $\hat{\mathbf{y}}$ to denote both the latent features with uniform noise added during training and the discretely quantized latent features during testing. For decoding, a parametric synthesis transform g_s decodes the quantized $\hat{\mathbf{y}}$ back to the reconstructed image $\hat{\mathbf{x}}$.

The fundamental optimization objective of learned image compression is to minimize a weighted sum of the rate-distortion tradeoff during training:

$$L = R(\hat{\mathbf{y}}) + \lambda D(\mathbf{x}, \hat{\mathbf{x}}). \tag{2}$$

The rate R is the the entropy of $\hat{\mathbf{y}}$, which is estimated by a non-parametric, fully factorized density model (entropy model) $p_{\hat{\mathbf{y}}|\theta}$ during training. So, we have the rate estimation:

$$R = \mathbb{E}[-\log_2 p_{\hat{\mathbf{y}}|\theta}(\hat{\mathbf{y}}|\theta)]. \tag{3}$$

The distortion D is defined as $D = MSE(\mathbf{x}, \hat{\mathbf{x}})$ for MSE optimization and $D = 1 - MS\text{-}SSIM(\mathbf{x}, \hat{\mathbf{x}})$ for MS-SSIM [47] optimization. We use λ to control the rate-distortion tradeoff for different bit rates.

Later, Ballé et al. [8] propose a scale hyperprior on top of the general learn image compression model, as shown in Figure 1b. Specifically, they stack another parametric analysis transform h_a on top of $\mathbf{y}$ to capture the significant spatial dependencies in the quantized latent features $\hat{\mathbf{y}}$ and obtain an additional set of encoded variables $\mathbf{z}$. In this way, they model $\hat{\mathbf{y}}$ as a zero-mean Gaussian distribution with standard deviations σ for all the elements. The standard deviations are estimated by another parametric synthesis transform h_s, which takes the quantized $\hat{\mathbf{z}}$ as input and outputs the estimated standard deviations $\hat{\sigma}$. So, we have the conditional probability distribution $p_{\hat{\mathbf{y}}|\hat{\mathbf{z}}} = \mathcal{N}(0, \sigma^2)$. Similarly, an entropy model $p_{\hat{\mathbf{z}}|\theta}$ is applied for entropy estimation of $\hat{\mathbf{z}}$. In this case, the rate estimation contains two terms:

$$R = \mathbb{E}[-\log_2 p_{\hat{\mathbf{y}}|\hat{\mathbf{z}}}(\hat{\mathbf{y}}|\hat{\mathbf{z}})] + \mathbb{E}[-\log_2 p_{\hat{\mathbf{z}}|\theta}(\hat{\mathbf{z}}|\theta)]. \tag{4}$$

Later works further improve the hyperprior to better parameterize the distributions of the quantized latent features with a more accurate and flexible entropy model. Minnen [34] propose an autoregressive context model with a mean and scale hyperprior, as shown in Figure 1c. Cheng et al. [11] utilize discretized Gaussian mixture likelihoods with attention enhancement to model the distributions of $\hat{\mathbf{y}}$, as shown in Figure 1d.

3.2 Proposed Method

Instead of optimizing the parameterization h_a, h_s of the latent feature distribution, our proposed method focuses on enhancing the analysis g_a and synthesis g_s transforms between the image space $\mathcal{X}$ and the latent feature space $\mathcal{Y}$. Considering the natural invertibility in image compression, we use an invertible network design with a feature enhancement module and an attentive channel squeeze layer to play the role of the analysis g_a and synthesis g_s transforms. Figure 2 shows an overview of our proposed approach.

INN architecture. We formulate an INN architecture design to serve as the analysis g_a and synthesis g_s transforms. It consists of two essential invertible layers: the downscaling layer and the coupling layer. Existing methods [8, 11, 34] usually employ 4 downscaling and 4 corresponding upscaling modules in the analysis and synthesis transforms, respectively. Similarly, we also stack 4 invertible blocks in our INN architecture to downscale the input resolution by a factor of 2^4, where each block sequentially contains 1 downscaling layer and 3 coupling layers. The kernel sizes of the coupling layers for the 4 blocks are empirically set as $k = 5, 5, 3, 3$.

The downscaling layer is composed of a pixel shuffling layer [14] and an invertible 1×1 convolution [26], and each downscaling layer reduces the resolution of the input tensor by 2 and quadruples the channel dimension.

We use the affine coupling layer design introduced in RealNVP [14]. For the i^{th} coupling layer that takes an input $\mathbf{u}^{(i)}_{1:C}$ with dimensional size of C, it splits the input at position $c < C$ into two parts and gives a C dimensional output $\mathbf{u}^{(i+1)}_{1:C}$:

$$\mathbf{u}^{(i+1)}_{1:c} = \mathbf{u}^{(i)}_{1:c} \odot \exp(\sigma_c(g_2(\mathbf{u}^{(i)}_{c+1:C}))) + h_2(\mathbf{u}^{(i)}_{c+1:C}), \tag{5}$$

$$\mathbf{u}^{(i+1)}_{c+1:C} = \mathbf{u}^{(i)}_{c+1:C} \odot \exp(\sigma_c(g_1(\mathbf{u}^{(i+1)}_{1:c}))) + h_1(\mathbf{u}^{(i+1)}_{1:c}), \tag{6}$$

where $\odot$ denotes the Hadamard product, $\exp(\cdot)$ denotes the exponential function, and $\sigma_c(\cdot)$ denotes the center sigmoid function. Symmetrically, the i^{th} coupling layer inversely takes $\mathbf{u}^{(i+1)}_{1:C}$ as input with splitting position c. The affine coupling layer gives a perfect inverse:

$$\mathbf{u}^{(i)}_{c+1:C} = (\mathbf{u}^{(i+1)}_{c+1:C} - h_1(\mathbf{u}^{(i+1)}_{1:c})) \odot \exp(-\sigma_c(g_1(\mathbf{u}^{(i+1)}_{1:c}))), \tag{7}$$

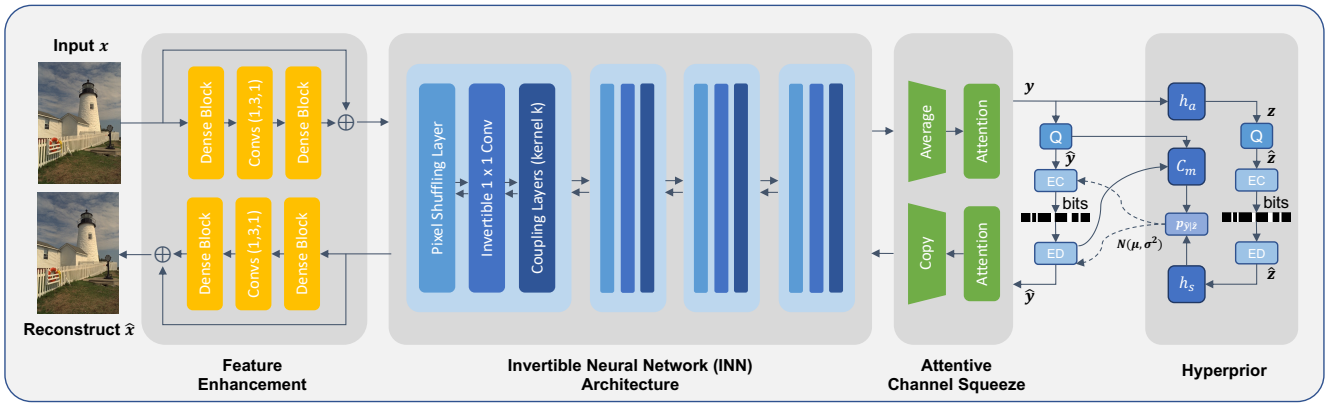


Figure 2: Overview and workflow of our proposed enhanced Invertible Encoding Network.

$$\mathbf{u}_{1:c}^{(i)} = (\mathbf{u}_{1:c}^{(i+1)} - h_2(\mathbf{u}_{c+1:C}^{(i)})) \odot \exp(-\sigma_c(g_2(\mathbf{u}_{c+1:C}^{(i)}))). \quad (8)$$

Please note that such invertibility is inherently guaranteed by the mathematical design. Thus, the invertibility holds for any arbitrary feedforward functions g_1, g_2, h_1, h_2, meaning that these functions need not be invertible. In our implementations, all these four functions use the following bottleneck design: Conv-LeakyReLU-Conv-LeakyReLU-Conv, where the kernel size of the first and the last convolution is set as k; the middle one is a convolutional layer with filter size 1.

Feature enhancement module. INNs are powerful when modeling invertible transformation. However, since the property of invertibility is guaranteed by the strictly invertible network design, INNs usually have a limited capacity of nonlinear representation [13]. Hence, we add a feature enhancement module in a residual manner before the INN architecture to improve the nonlinear representativeness of our network. Specifically, this module is based on the popular Dense Block [22], and "Convs (1,3,1)" means three cascade convolutions with kernel size 1, 3, 1.

Attentive channel squeeze. All the operations in INNs cannot change the total number of pixels in the input tensor, many of which are simply redundant pixels to be compressed. To resolve this problem, we introduce an attentive channel squeeze layer to reduce the channel dimension of the output tensor of INNs. The idea is as follows: Given a compression ratio α and an input tensor $\mathbf{v}$ with size (C, H, W), the attentive channel squeeze layer first forwardly reshape the tensor into a shape of $(\alpha, \frac{C}{\alpha}, H, W)$. Then it performs average operation along the first dimension to obtain the latent features $\mathbf{y}$ with size $(\frac{C}{\alpha}, H, W)$, followed by an attention module proposed in [11]. For the inverse process, the attentive channel squeeze layer makes α copies of the quantized $\hat{\mathbf{y}}$ first after the attention module and reshapes it into a size of (C, H, W).

3.3 Overall Workflow

For the encoding process, given an input image $\mathbf{x} \in \mathbb{R}^{3\times H\times W}$ to be compressed with height H and width W, the feature enhancement module first extracts and adds some nonlinear representation to obtain $\mathbf{u} \in \mathbb{R}^{3\times H\times W}$. Then the forward pass of our INN architecture transforms $\mathbf{u}$ into $\mathbf{v} \in \mathbb{R}^{d\times h\times w}$, where $d = 3 \times 4^4$, $h = \frac{H}{2^4}$, and $w = \frac{W}{2^4}$ under our architecture design. Given a hyper-parameter compression ratio α, the attentive channel squeeze layer further compresses $\mathbf{v}$ into latent features $\mathbf{y} \in \mathbb{R}^{\frac{d}{\alpha}\times h\times w}$.

For the decoding process, the attentive channel squeeze layer copies the quantized latent features $\hat{\mathbf{y}}$ after attention enhancement for α times and then reshapes it as $\hat{\mathbf{v}}$. The inverse pass of the INN architecture decodes $\hat{\mathbf{v}}$ to obtain $\hat{\mathbf{u}}$, which finally undergoes the feature enhancement module for the reconstructed image $\hat{\mathbf{x}}$.

We employ the same hyperprior proposed by Minnen et al. [34], which uses a mean and scale Gaussian distribution to parameterize the quantized latent features $\hat{\mathbf{y}}$ with a pair of analysis h_a and synthesis h_s transforms. Specifically, h_a obtains the side information $\mathbf{z} = h_a(\hat{\mathbf{y}})$, and h_s takes the quantized side information $\hat{\mathbf{z}}$ as input. The output $h_s(\hat{\mathbf{z}})$ works with the causal context $C_m(\hat{\mathbf{y}})$ for the mean and scale Gaussian model $p_{\hat{\mathbf{y}}|\hat{\mathbf{z}}} \leftarrow h_s(\hat{\mathbf{z}}), C_m(\hat{\mathbf{y}})$. Since there is no prior for $\hat{\mathbf{z}}$, a factorized-prior entropy model θ introduced in [7] is used to parametize the distribution of $\hat{\mathbf{z}}$ as $p_{\hat{\mathbf{z}}|\theta}$. For entropy coding, we use the asymmetric numeral systems (ANS) [15] to losslessly compress $\hat{\mathbf{y}}$ and $\hat{\mathbf{z}}$ into bitstreams.

4 EXPERIMENTS

We conduct experiments on three commonly used image compression datasets to validate our method. In the remainder of this section, we first introduce the detailed experimental setup, followed by quantitative and qualitative comparisons with existing state-of-the-art methods. We further conduct some analysis and ablation studies to examine the effectiveness of our proposed feature enhancement module and attentive channel squeeze layer.

4.1 Experimental Setup

Training details. We use the Flicker 2W dataset used in [30], consisting of 20,745 high-quality general images for the image compression task. We randomly select around 200 images for our validation set, and the remaining images are used for training. Our network is trained on 256×256 randomly cropped patches, using the recently well-developed CompressAI PyTorch library [9]. Note that we drop a few images with either height or width smaller than 256 px for convenience.

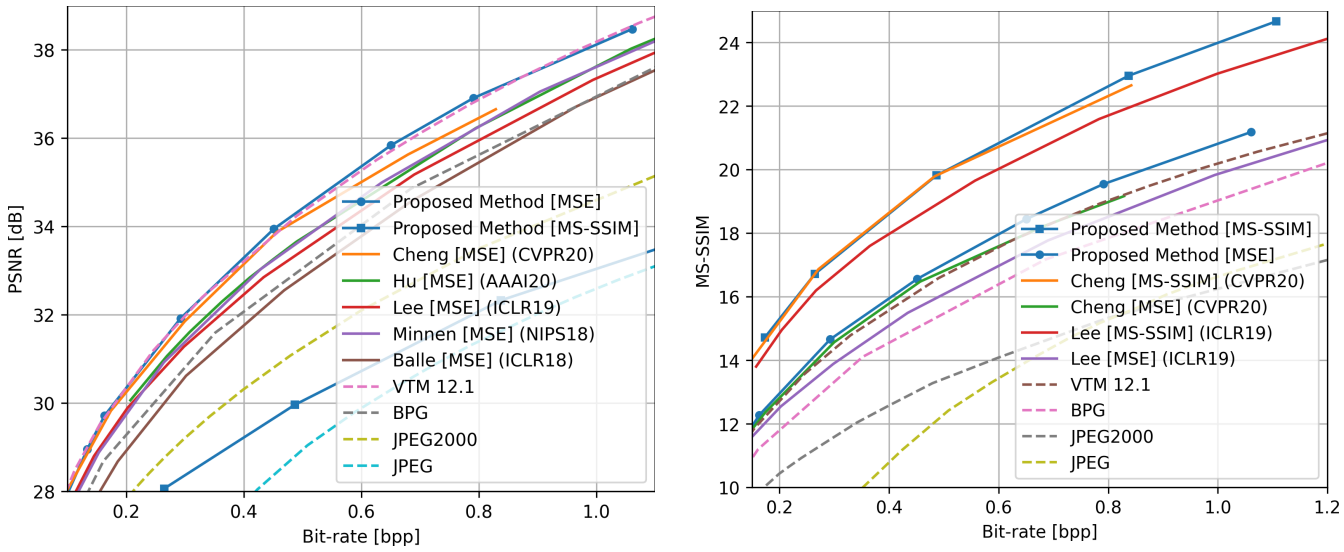


Figure 3: Performance evaluation on the Kodak dataset.

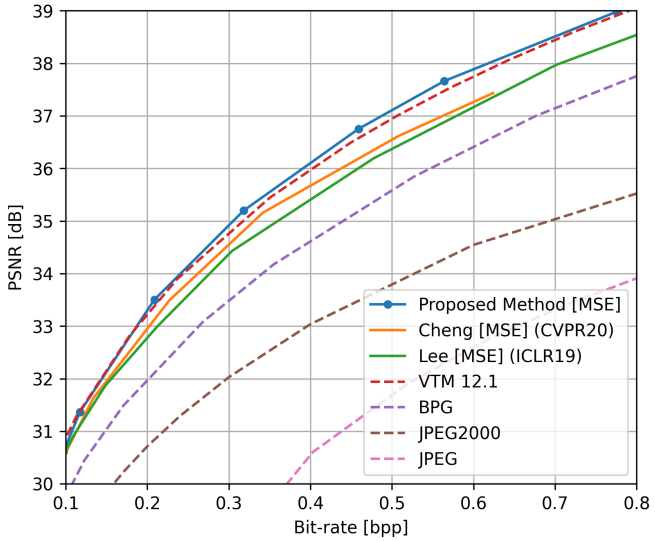


Figure 4: Performance evaluation on the CLIC Professional Validation dataset.

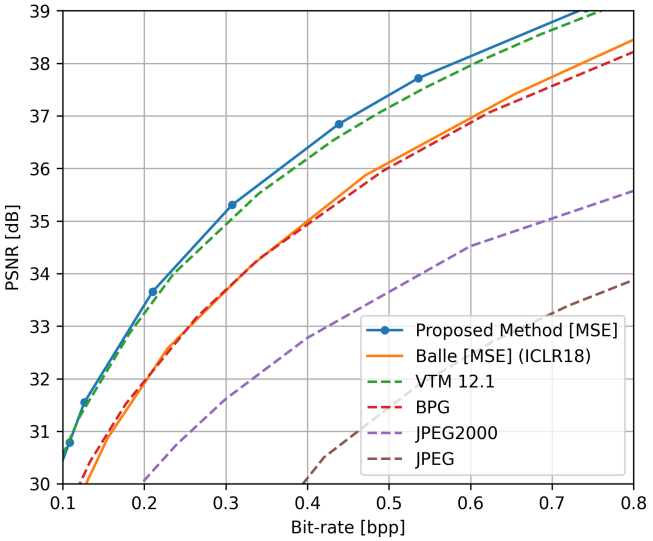


Figure 5: Performance evaluation on the Tecnick dataset.

Table 1: Area under curve (AUC) of VVC (VTM 12.1) and our MSE method on different datasets, with the bpp range determined by our quality 1 and quality 8 models.

Dataset	VVC (VTM 12.1)	Ours [MSE]
Kodak	33.309	**33.373**
CLIC	25.153	**25.238**
Tecnick	23.563	**23.693**

All the experiments are conducted on a single RTX 2080 Ti GPU and trained for 600 epochs with a batch size of 8 using Adam [25] optimizer. Generally, it takes around 10 days to train a model. Our network is first optimized for 450 epochs with an initial learning rate of 10^{-4}; then the learning rate is reduced to 10^{-5} at epoch 450 and further down to 10^{-6} at epoch 550.

We use the channel number $N = \frac{C}{\alpha}$ and the weight factor λ as our quality parameters. We in-total train 8 models optimized with MSE (mean squared error) quality metric and 5 models with MS-SSIM (multiscale structural similarity) quality metric [47] under different quality levels. For the MSE models, λ is chosen from the set $\{0.0016, 0.0024, 0.0032, 0.0075, 0.015, 0.03, 0.045, 0.09\}$, in which the first four λ values are paired with channel number $N = 128$ for lower-rate models, and N is set as 192 to pair with the remaining four λ values for higher-rate models. For the MS-SSIM models, λ

Figure 6: Visualization of sample reconstructed images from the Kodak dataset.

belongs to the set of $\{6, 12, 40, 120, 220\}$, and we use $N = 128$ for the first two λ values to train lower-rate models and $N = 192$ for the remaining three λ values to train higher-rate models.

Evaluation. We evaluate our methods on three commonly used datasets for image compression, which are the Kodak PhotoCD image dataset (Kodak) [12], the CLIC Professional Validation dataset (CLIC) [43], and the old Tecnick dataset [5]. The Kodak dataset contains 24 uncompressed images with resolutions of 768×512; the CLIC dataset comprises 41 high-quality images with much higher resolutions; the Tecnick dataset contains 100 images with high resolutions of 1200×1200.

We use the peak signal-to-noise ratio (PSNR) and the multiscale structural similarity index (MS-SSIM) [47] to quantify the image distortion level; we use the bits per pixel (bpp) to evaluate the rate

Table 2: Deviation and scaled deviation of the averaging operation of our attentive channel squeeze layer on the Kodak dataset.

Quality	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8
Deviation (ϵ)	0.1344	0.1448	0.1662	0.2344	0.2047	0.2520	0.3541	0.4568
Scaled Deviation ($\tilde{\epsilon}$)	0.6640	0.5884	0.5912	0.5015	0.4421	0.3637	0.4106	0.3611

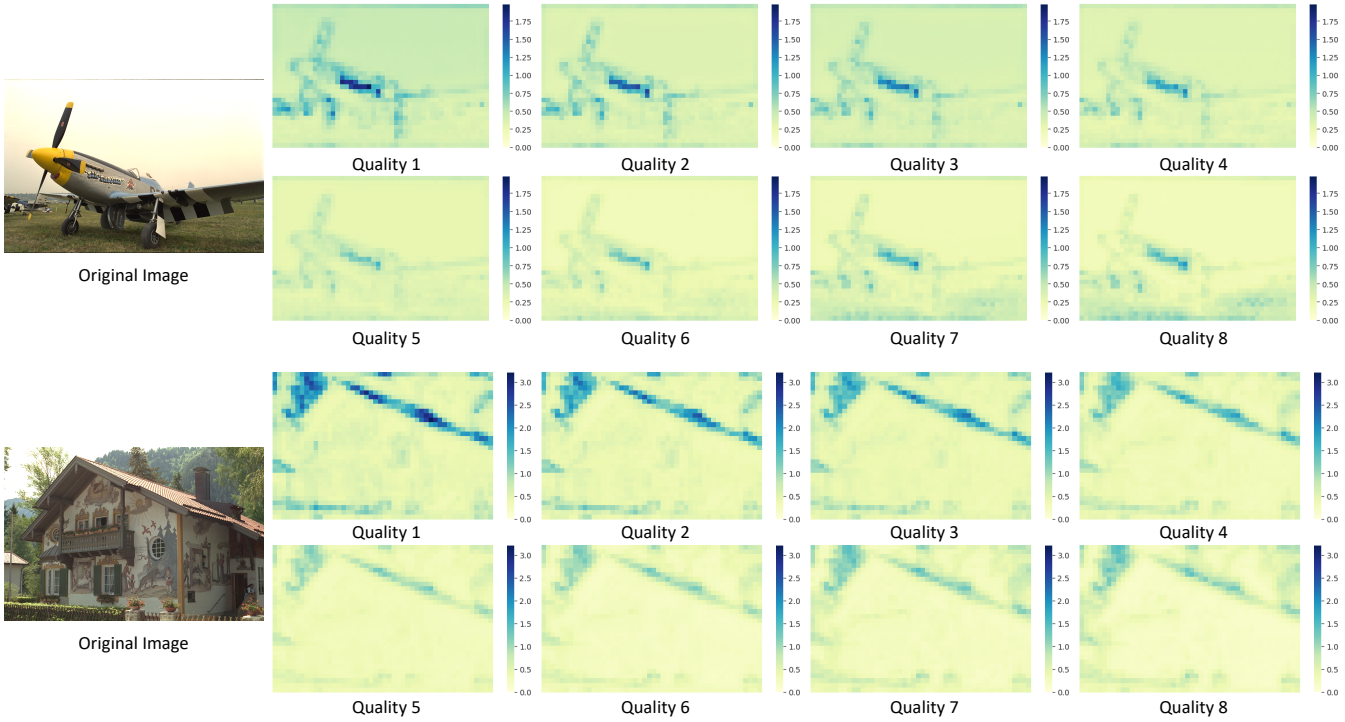


Figure 7: Scaled deviation map of image *kodim20* and *kodim24* from the Kodak dataset.

performance. We draw the rate-distortion (RD) curves according to their rate-distortion performance to compare the coding efficiency of different methods. We also report the area under the rate-distortion curve (AUC) as an aggregate measurement to better compare methods with similar performance.

4.2 Rate-distortion Performance

We compare our model with state-of-the-art learned image compression models, including the methods proposed by Ballé el al. [8], Minnen et al. [34], Lee et al. [27], Hu et al. [21], and Cheng et al. [11]. The corresponding data points on their RD curves are collected from their paper or their official GitHub pages. We also compare with some widely-used image compression codecs, including JPEG [45], JPEG2000 [37], WebP [17], BPG [10], and VVC [24]. We evaluate their performance using the CompressAI evaluation platform. For VVC, we use the most up-to-date VVC Official Test Model VTM 12.1 (accessed on April 2021) with an intra-profile configuration from the official GitHub page to test on images. To evaluate the BPG performance, we use the BPG software with subsampling mode of YUV444, HEVC implementation of x265, and bit depth of 8 to test on images.

Figure 3 shows the RD curve comparison on the Kodak dataset. Similar to existing work [11], we convert $MS{-}SSIM$ to $-10\log_{10}(1-MS\text{-}SSIM)$ for clearer comparison. It can be observed that our method slightly outperforms VVC (VTM 12.1) and yields much better performance when comparing with both the existing learned methods and other traditional image compression standards.

For the CLIC dataset and the Tecnick dataset, we compare our MSE optimized results with traditional compression standards and the learned methods with official testing results available in their paper or their official GitHub pages. We show the RD curves on the CLIC dataset in Figure 4 and the Tecnick dataset in Figure 5. We can see that our MSE optimized method outperforms all other approaches. Note that most images in the CLIC dataset and the Tecknick dataset are of high resolutions, implying that our method is more robust and promising to compress high-resolution images.

From the RD curves, we can see that VVC (VTM 12.1) and our approach achieve similar performance in terms of PSNR. Hence, we further report their corresponding AUC values for better comparison and ranking. Statistics in Table 1 indicates that our method outperforms the latest VVC (VTM 12.1) codec in terms of the aggregated AUC metric.

4.3 Qualitative Results

We show some qualitative comparison of some sample reconstructed images on the Kodak dataset in Figure 6. The first sample is image *kodim01* with an approximate bpp of 0.145; the second sample is image *kodim07* with approximately 0.125 bpp; the last sample is image *kodim22* with around 0.130 bpp. For JPEG and JPEG2000, we use the lowest quality since they cannot reach the mentioned bpp levels. We can see that our MSE optimized method achieves a good performance compared with the latest VVC (VTM 12.1) codec, much better than the performance of other codecs. Also, our MS-SSIM optimized method can reconstruct images with much more structural details than all the traditional codecs. We present more qualitative results in our supplementary materials.

4.4 Analysis of Attentive Channel Squeeze

In this section, we demonstrate that our proposed attentive channel squeeze layer introduces only minor deviation to enable a stable and tractable dimension adjustment. We also analyze the distribution of the deviation maps on images and the deviation levels of different quality models.

Let $y \in \mathbb{R}^{d \times h \times w}$ and $\hat{y} \in \mathbb{R}^{\frac{d}{\alpha} \times h \times w}$ be the input and output latent feature tensors of the averaging operation, respectively. For simplicity of notation, we reshape the y as (α, l) and $\hat{y}$ as $(1, l)$, where $l = \frac{d}{\alpha} \times h \times w$. The averaging operation in the attentive channel squeeze layer compresses y by compression ratio α along the channel dimension. Specifically, each pixel in $\hat{y}$ is the mean value of the corresponding α pixels in y. The corresponding inverse operation for dimension matching is copying. It is obvious that such operation can lead to some deviation.

To quantify such deviation, we do some analysis using the Kodak dataset to calculate the mean absolute pixel deviation as follows:

$$\epsilon = \frac{1}{l} \sum_{j=1}^{l} \sum_{i=1}^{\alpha} |y_{i,j} - \hat{y}_{1,j}|, \tag{9}$$

where $y_{i,j}$ and $\hat{y}_{1,j}$ denote the corresponding pixel in y and $\hat{y}$, respectively. As shown in Table 2, the value of the deviation is minor (less than or comparable with the quantization error in most cases), which is beneficial for stable training.

To compare the deviation among models under different quality levels, it is unfair to directly compare the values because y and $\hat{y}$ have a larger value range for the model with higher quality. Accordingly, the absolute value of deviation is amplified. As a result, calculating the "relative" deviation concerning the value range is a more reasonable practice. A naive way is to divide the mean absolute pixel deviation ϵ by the value range for fair comparison among different quality models. However, we find that the distribution of the pixel values has long tails, so the value range is very likely to be affected by the outlier pixel values. Hence, to better scale the mean absolute pixel deviation, we introduce a scaling factor μ:

$$\mu = \frac{1}{l} \sum_{j=1}^{l} \sum_{i=1}^{\alpha} |y_{i,j}|. \tag{10}$$

The scaled mean absolute pixel deviation $\tilde{\epsilon} = \frac{\epsilon}{\mu}$. We report the ϵ and $\tilde{\epsilon}$ values of our models under different quality levels in Table 2.

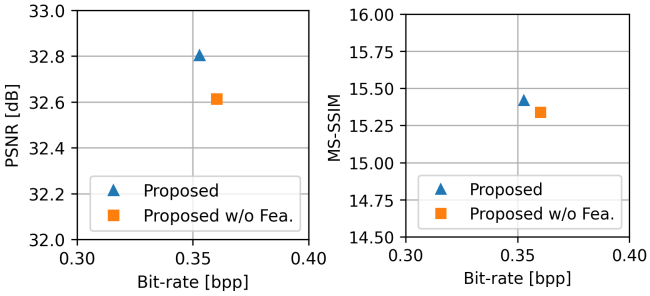


Figure 8: Ablation study on feature enhancement module.

We further visualize the scaled deviation map of the image *kodim20* and image *kodim24* between y and $\hat{y}$ of our models under different quality levels in Figure 7. Note that the deviation map is in shape (h, w), and each pixel is the mean of the absolute deviation along the channel dimension after scaling with μ. We can see that the "relative" deviation generally decreases as the quality increases, indicating less information loss of the averaging operation, which is in line with our intuition: higher-quality models usually lead to less information loss in the process of compression.

4.5 Ablation Study

To verify the contribution of the proposed nonlinear feature enhancement module, we conduct a corresponding ablation study on this module. Specifically, we train two models with and without using the nonlinear feature enhancement module on the Flicker 2W dataset for 600 epochs, using the same weight factor $\lambda = 0.01$ and channel number $N = 192$.

Figure 8 shows the rate-distortion points of two models evaluated on the Kodak dataset. We can observe that the proposed nonlinear feature enhancement module improves the modeling capacity of the model to compress images with lower bit rates and higher PSNR and MS-SSIM values.

5 CONCLUSION

Unlike existing autoencoder style networks, our proposed enhanced Invertible Encoding Network based on invertible neural networks (INNs) can better model image compression as an invertible process. The critical issues in integrating INNs for image compression include unstable training and limited nonlinear transformation capacity. Our proposed attentive channel squeeze layer offers stable and tractable feature dimension adjustment; the incorporated feature enhancement module increases the network nonlinear transformation capacity. Overall, our network maintains a highly invertible architecture to largely mitigate information loss when compressing images.

Extensive experiments on three widely-used datasets show that our approach outperforms state-of-the-art learned image compression methods and existing compression standards, including VVC (VTM 12.1), especially on the two high-resolution image datasets. Furthermore, the visual results demonstrate that our compression methods preserve more detailed information than current compression standards.

REFERENCES

[1] Eirikur Agustsson, Fabian Mentzer, Michael Tschannen, Lukas Cavigelli, Radu Timofte, Luca Benini, and Luc Van Gool. 2017. Soft-to-Hard Vector Quantization for End-to-End Learning Compressible Representations. In *Advances in Neural Information Processing Systems*.

[2] Eirikur Agustsson, Michael Tschannen, Fabian Mentzer, Radu Timofte, and Luc Van Gool. 2019. Generative Adversarial Networks for Extreme Learned Image Compression. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 221–231.

[3] Nasir U. Ahmed, T. Natarajan, and Kamisetty R. Rao. 1974. Discrete Cosine Transform. *IEEE Trans. Comput.* C-23, 1 (1974), 90–93.

[4] Lynton Ardizzone, Jakob Kruse, Sebastian Wirkert, Daniel Rahner, Eric W. Pellegrini, Ralf S. Klessen, Lena Maier-Hein, Carsten Rother, and Ullrich Köthe. 2019. Analyzing Inverse Problems with Invertible Neural Networks. In *Proceedings of the International Conference on Learning Representations*.

[5] Nicola Asuni and Andrea Giachetti. 2014. TESTIMAGES: a Large-scale Archive for Testing Visual Devices and Basic Image Processing Algorithms. In *Proceedings of the Conference on Smart Tools and Applications in Computer Graphics*. 63–70.

[6] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. 2016. End-to-end Optimization of Nonlinear Transform Codes for Perceptual Quality. In *Proceedings of Picture Coding Symposium*. 1–5.

[7] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. 2017. End-to-end Optimized Image Compression. In *Proceedings of the International Conference on Learning Representations*.

[8] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. 2018. Variational Image Compression with a Scale Hyperprior. In *Proceedings of the International Conference on Learning Representations*.

[9] Jean Bégaint, Fabien Racapé, Simon Feltman, and Akshay Pushparaja. 2020. CompressAI: a PyTorch Library and Evaluation Platform for End-to-end Compression Research. *arXiv preprint arXiv:2011.03029* (2020).

[10] Fabrice Bellard. 2015. BPG Image Format. https://bellard.org/bpg/

[11] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. 2020. Learned Image Compression with Discretized Gaussian Mixture Likelihoods and Attention Modules. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7939–7948.

[12] Eastman Kodak Company. 1999. Kodak Lossless True Color Image Suite. http://r0k.us/graphics/kodak/

[13] Laurent Dinh, David Krueger, and Yoshua Bengio. 2015. NICE: Non-linear Independent Components Estimation. In *Proceedings of the International Conference on Learning Representations Workshops*.

[14] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. 2017. Density Estimation using Real NVP. In *Proceedings of the International Conference on Learning Representations*.

[15] Jarek Duda. 2009. Asymmetric numeral systems. *arXiv preprint arXiv:0902.0271* (2009).

[16] Allen Gersho and Robert M. Gray. 2012. *Vector Quantization and Signal Compression*. Vol. 159.

[17] Google. 2010. Web Picture Format. https://chromium.googlesource.com/webm/libwebp

[18] Vivek K. Goyal. 2001. Theoretical Foundations of Transform Coding. *IEEE Signal Processing Magazine* 18, 5 (2001), 9–21.

[19] Zongyu Guo, Yaojun Wu, Runsen Feng, Zhizheng Zhang, and Zhibo Chen. 2020. 3-D Context Entropy Model for Improved Practical Image Compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 116–117.

[20] Leonhard Helminger, Abdelaziz Djelouah, Markus Gross, and Christopher Schroers. 2021. Lossy Image Compression with Normalizing Flows. In *Proceedings of the International Conference on Learning Representations Workshop Neural Compression*.

[21] Yueyu Hu, Wenhan Yang, and Jiaying Liu. 2020. Coarse-to-Fine Hyper-Prior Modeling for Learned Image Compression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 11013–11020.

[22] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2017. Densely Connected Convolutional Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2261–2269.

[23] Nick Johnston, Damien Vincent, David Minnen, Michele Covell, Saurabh Singh, Troy Chinen, Sung Jin Hwang, Joel Shor, and George Toderici. 2018. Improved Lossy Image Compression with Priming and Spatially Adaptive Bit Rates for Recurrent Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4385–4393.

[24] Joint Video Experts Team (JVET). 2021. VVC Official Test Model VTM. https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM/-/tree/VTM-12.1 accessed on April 5, 2021.

[25] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceddings of the International Conference on Learning Representations*.

[26] Durk P. Kingma and Prafulla Dhariwal. 2018. Glow: Generative Flow with Invertible 1x1 Convolutions. In *Advances in Neural Information Processing Systems*, Vol. 31.

[27] Jooyoung Lee, Seunghyun Cho, and Seung-Kwon Beack. 2019. Context-adaptive Entropy Model for End-to-end Optimized Image Compression. In *Proceedings of the International Conference on Learning Representations*.

[28] Chaoyi Lin, Jiabao Yao, Fangdong Chen, and Li Wang. 2020. A Spatial RNN Codec for End-to-End Image Compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13266–13274.

[29] Haojie Liu, Tong Chen, Peiyao Guo, Qiu Shen, Xun Cao, Yao Wang, and Zhan Ma. 2019. Non-local Attention Optimized Deep Image Compression. *arXiv preprint arXiv:1904.09757* (2019).

[30] Jiaheng Liu, Guo Lu, Zhihao Hu, and Dong Xu. 2020. A Unified End-to-End Framework for Efficient Deep Image Compression. *arXiv preprint arXiv:2002.03370* (2020).

[31] Andreas Lugmayr, Martin Danelljan, Luc Van Gool, and Radu Timofte. 2020. SR-Flow: Learning the Super-Resolution Space with Normalizing Flow. In *Proceedings of the European Conference on Computer Vision*.

[32] Detlev Marpe, Heiko Schwarz, and Thomas Wiegand. 2003. Context-based Adaptive Binary Arithmetic Coding in the H. 264/AVC Video Compression Standard. *IEEE Transactions on Circuits and Systems for Video Technology* 13, 7 (2003), 620–636.

[33] Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, and Luc Van Gool. 2018. Conditional Probability Models for Deep Image Compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4394–4402.

[34] David Minnen, Johannes Ballé, and George Toderici. 2018. Joint Autoregressive and Hierarchical Priors for Learned Image Compression. In *Advances in Neural Information Processing Systems*. 10794–10803.

[35] David Minnen and Saurabh Singh. 2020. Channel-Wise Autoregressive Entropy Models for Learned Image Compression. In *Proceedings of the IEEE International Conference on Image Processing*. 3339–3343.

[36] Albert Pumarola, Stefan Popov, Francesc Moreno-Noguer, and Vittorio Ferrari. 2020. C-Flow: Conditional Generative Flow Models for Images and 3D Point Clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

[37] Majid Rabbani. 2002. JPEG2000: Image Compression Fundamentals, Standards and Practice. *Journal of Electronic Imaging* 11, 2 (2002), 286.

[38] Oren Rippel and Lubomir Bourdev. 2017. Real-time Adaptive Image Compression. In *Proceedings of the International Conference on Machine Learning*. 2922–2930.

[39] Jorma Rissanen and Glen G. Langdon. 1981. Universal Modeling and Coding. *IEEE Transactions on Information Theory* 27, 1 (1981), 12–23.

[40] Shibani Santurkar, David M. Budden, and Nir Shavit. 2018. Generative Compression. In *Proceedings of Picture Coding Symposium*. 258–262.

[41] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. 2017. Lossy Image Compression with Compressive Autoencoders. In *Proceedings of the International Conference on Learning Representations*.

[42] George Toderici, Sean M. O'Malley, Sung Jin Hwang, Damien Vincent, David Minnen, Shumeet Baluja, Michele Covell, and Rahul Sukthankar. 2015. Variable Rate Image Compression with Recurrent Neural Networks. In *Proceedings of the International Conference on Learning Representations*.

[43] George Toderici, Wenzhe Shi, Radu Timofte, Johannes Balle Lucas Theis, Eirikur Agustsson, Nick Johnston, and Fabian Mentzer. 2021. Workshop and Challenge on Learned Image Compression. http://www.compression.cc

[44] George Toderici, Damien Vincent, Nick Johnston, Sung Jin Hwang, David Minnen, Joel Shor, and Michele Covell. 2017. Full Resolution Image Compression with Recurrent Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 5306–5314.

[45] Gervais Knox Wallace. 1992. The JPEG Still Picture Compression Standard. *IEEE Transactions on Consumer Electronics* 38, 1 (1992), xviii–xxxiv.

[46] Yaolong Wang, Mingqing Xiao, Chang Liu, Shuxin Zheng, and Tie-Yan Liu. 2020. Modeling Lost Information in Lossy Image Compression. *arXiv preprint arXiv:2006.11999* (2020).

[47] Zhou Wang, Eero P. Simoncelli1, and Alan C. Bovik. 2003. Multiscale Structural Similarity for Image Quality Assessment. In *Proceedings of the Asilomar Conference on Signals, Systems and Computers*, Vol. 2. 1398–1402.

[48] Ian H. Witten, Radford M. Neal, and John G. Cleary. 1987. Arithmetic Coding for Data Compression. *Commun. ACM* 30, 6 (1987), 520–540.

[49] Mingqing Xiao, Shuxin Zheng, Chang Liu, Yaolong Wang, Di He, Guolin Ke, Jiang Bian, Zhouchen Lin, and Tie-Yan Liu. 2020. Invertible Image Rescaling. In *Proceedings of the European Conference on Computer Vision*. 126–144.

[50] Yazhou Xing, Zian Qian, and Qifeng Chen. 2021. Invertible Image Signal Processing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

[51] Yulun Zhang, Kunpeng Li, Kai Li, Bineng Zhong, and Yun Fu. 2019. Residual Non-local Attention Networks for Image Restoration. In *Proceedings of the International Conference on Learning Representations*.

[52] Lei Zhou, Zhenhong Sun, Xiangji Wu, and Junmin Wu. 2019. End-to-end Optimized Image Compression with Attention Mechanism.. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*.