Lossy Image Compression

Review of Theory and State-of-the-Art Techniques

Agenda

1. The Problem of Lossy Compression
2. Transform Coding Lossy Compression
    a. Traditional Techniques
    b. Basic Learning-Driven Techniques
3. Current Research and SOTA Techniques
    a. Directions of Research
    b. Notable Works
4. Discussion
5. Summary

The Problem of Lossy Compression

What is Lossy Compression?

$$\overset{\text{data}}{x} \xrightarrow{\text{encoder}} \overset{P}{\hat{z}} \xrightarrow{\text{decoder}} \hat{x} \qquad \hat{x} \neq x$$

data   latent representation   reconstruction

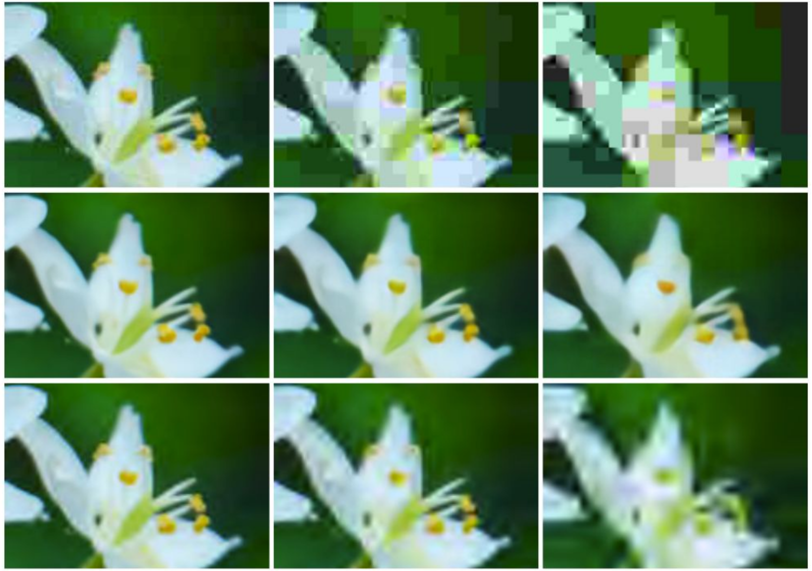
Quantizes x: some information lost

Lossless

- Entropy model P(z) to write prefix code
- Traditional compression: deterministic
- Neural compression: stochastic

Key Problems of Lossy Compression

- Which information can be discarded?
- How do we evaluate lossy compression?

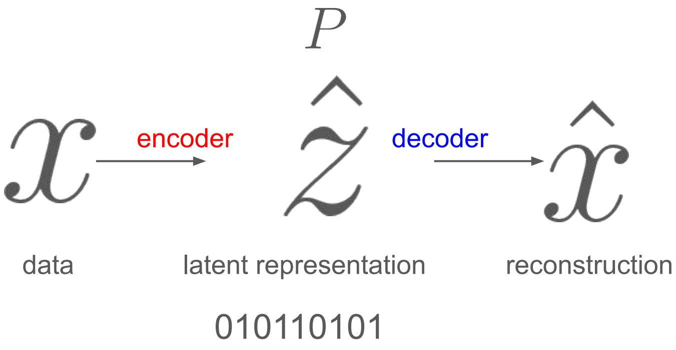


Which one is the ‘best’?

Ballé, Johannes, Valero Laparra, and Eero P. Simoncelli. "End-to-End Optimized Image Compression." arXiv, 2016, arxiv.org/abs/1611.01704.

Shannon's Rate-Distortion Theory

- Distortion Metric: $D = \mathbb{E}[d(x, \hat{x})]$
  - e.g. $d(x, \hat{x}) = ||x - \hat{x}||^2$
- $\hat{z}$ is losslessly transmitted as bits
  - Entropy model P
- Rate Metric: $R = \mathbb{E}[-\log P(\hat{z})]$



Lossy Compression Goal: Choose **encoder**, **decoder** and **entropy model** such that R and D is minimized.

Shannon’s Rate-Distortion Theory

- Shannon defines Rate-Distortion function:

$$R(D) = \inf_{p(\hat{x}|x):\mathbb{E}[d(x,\hat{x})]\leq D} I(x;\hat{x})$$

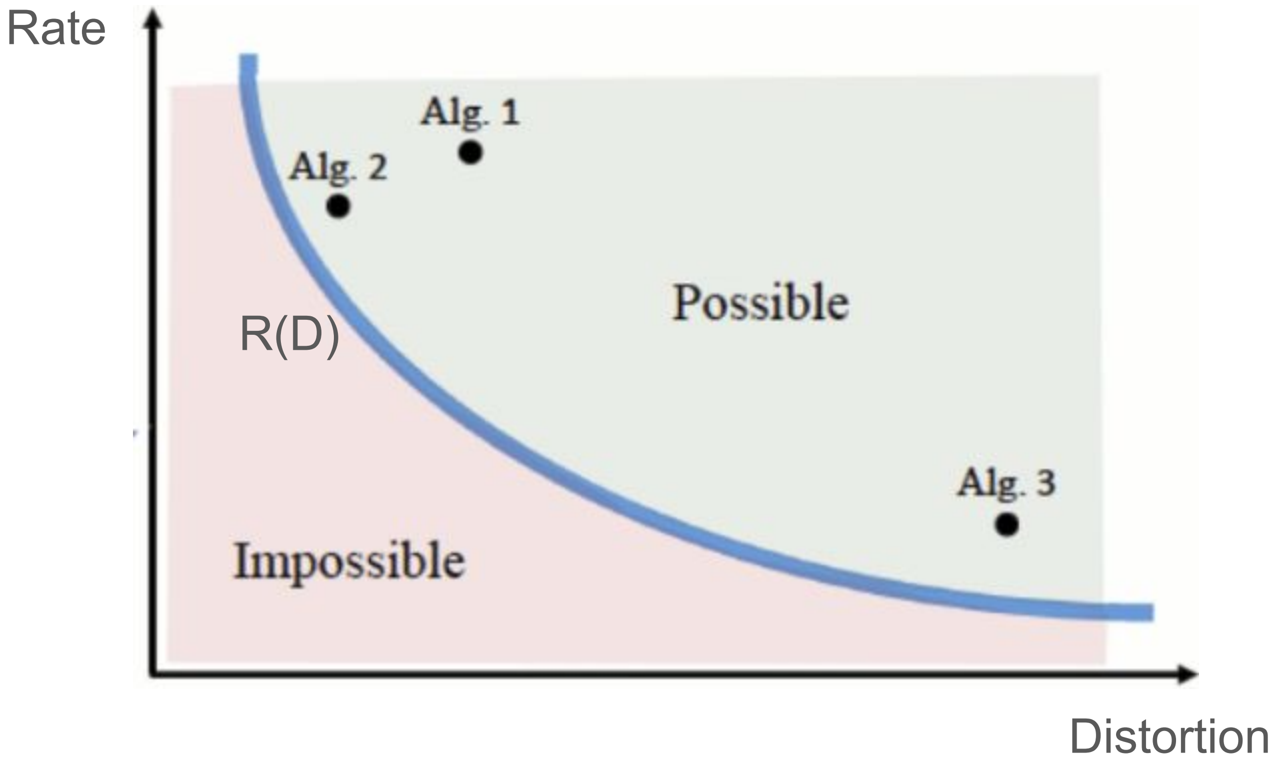
- Best operational Rate-Distortion defined as:

$$R_O(D) = \min_{(e,d,P):\mathbb{E}[d(x,\hat{x})]\leq D} \mathbb{E}[-\log P(\hat{z})]$$

- Shannon’s lossy source coding theorem:

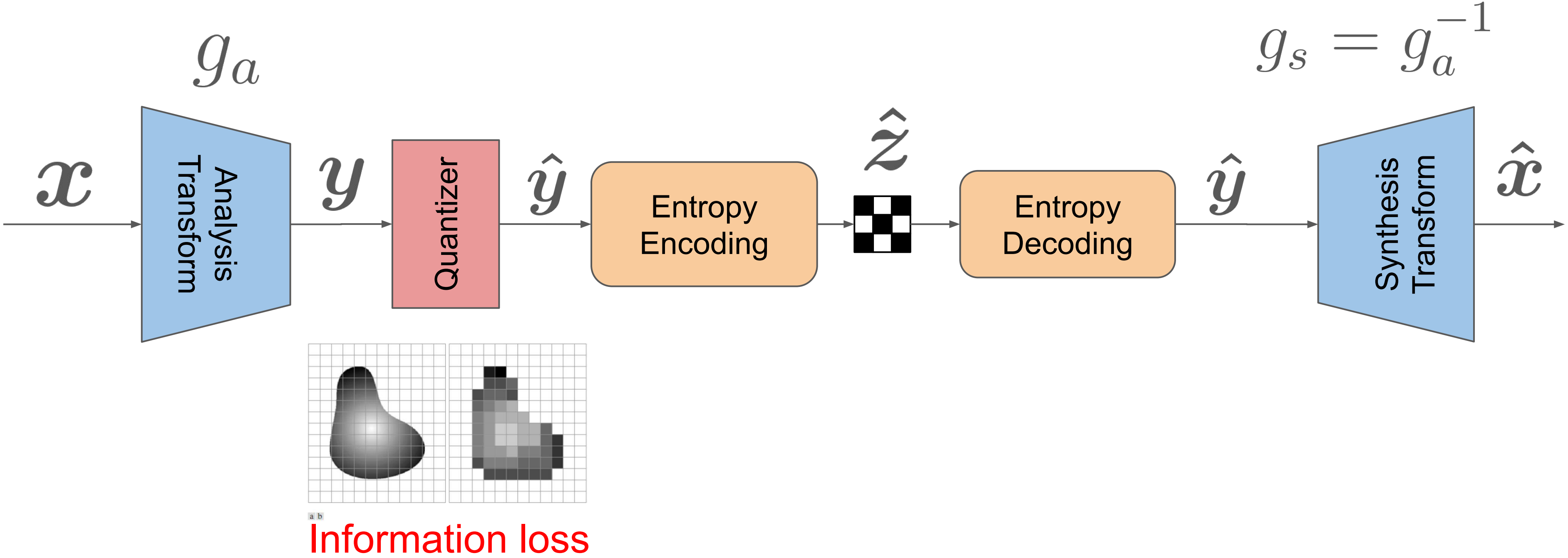
$$R_O(D) \geq R(D)$$

Shannon’s Rate-Distortion Theory



Transform Coding Lossy Compression

Transform Coding Framework



Ballé, Johannes, Valero Laparra, and Eero P. Simoncelli. "End-to-End Optimized Image Compression." arXiv, 2016, arxiv.org/abs/1611.01704.

Traditional Techniques

Traditional Techniques

JPEG

JPEG 2000

BPG

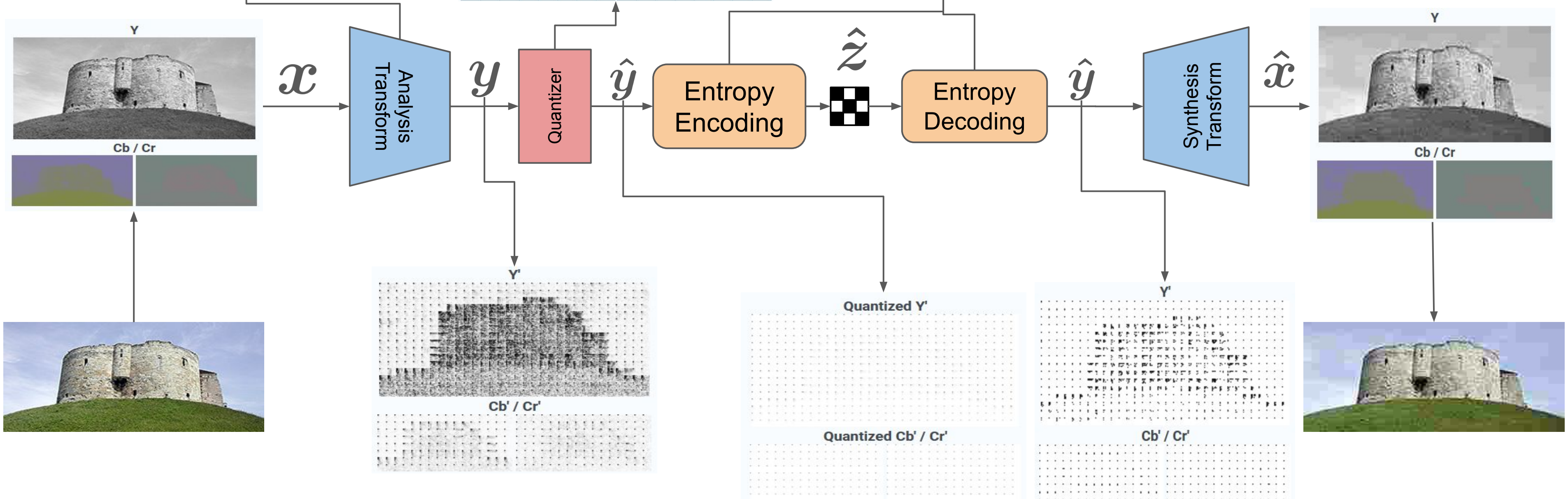
JPEG Committee. "JPEG - The Still Image Compression Standard." JPEG, jpeg.org/. Accessed 30 Aug. 2024.
Bellard, Fabrice. "BPG Image Format." BPG - Better Portable Graphics, bellard.org/bpg/. Accessed 30 Aug. 2024.

Traditional Technique Example - JPEG

$$X_k = \sum_{n=0}^{N-1} x_n \cos\left[\frac{\pi}{N} \left(n + \frac{1}{2}\right) k \right] \qquad \text{for } k = 0, \ \dots \ N-1 \, .$$

Luminance (brightness) table								Chrominance (colour) table							
76	57	66	66	85	113	231	340	80	85	113	222	467	467	467	467
52	57	61	80	104	165	302	434	85	99	123	312	467	467	467	467
47	66	76	104	175	260	368	448	113	123	264	467	467	467	467	467
76	90	113	137	264	302	411	463	222	312	467	467	467	467	467	467
113	123	189	241	321	382	486	529	467	467	467	467	467	467	467	467
189	274	269	411	514	491	571	472	467	467	467	467	467	467	467	467
241	283	326	378	486	533	566	486	467	467	467	467	467	467	467	467
288	260	264	293	363	434	477	467	467	467	467	467	467	467	467	467

Huffman Code



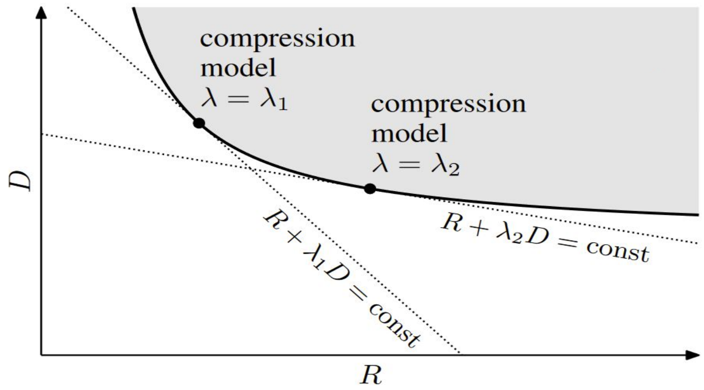
Jennings, Chris. "JPEG Compression: Visualizing the Loss of Quality." CGJennings.ca, cgjennings.ca/articles/jpeg-compression/. Accessed 30 Aug. 2024.

Basic Learning-Driven Techniques

Basic Learning-Driven Techniques

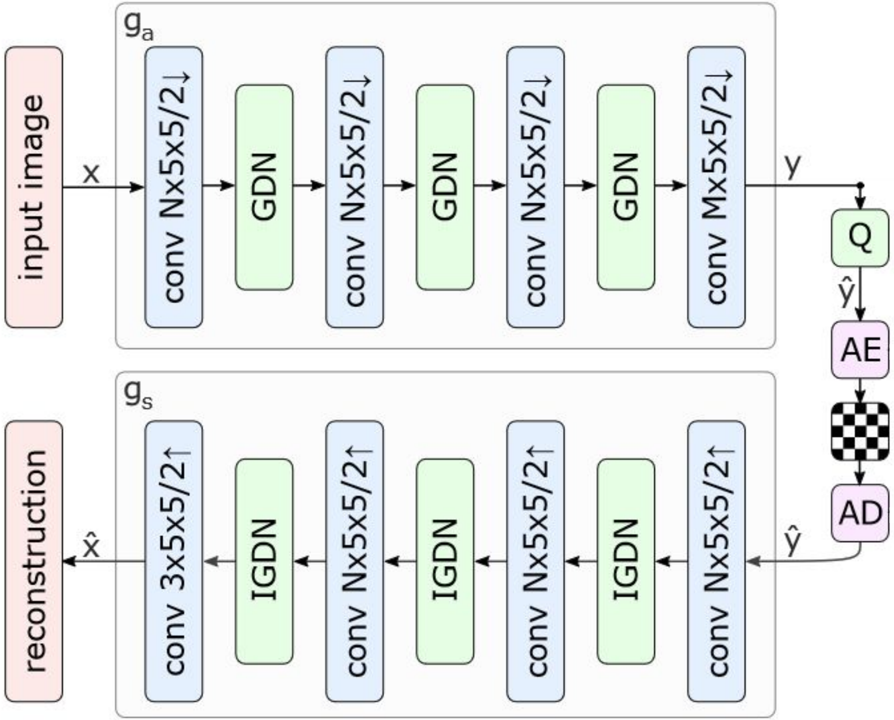
Key Principle: Use end-to-end trained neural networks for the analysis and synthesis transforms.

- Learned entropy model as in lossless compression
- The loss function is defined as: $\min_{(g_a, g_s, P)} \mathbb{E}[-\log P(\hat{z})] + \lambda \mathbb{E}[d(x, \hat{x})]$



Ballé, Johannes, Valero Laparra, and Eero P. Simoncelli. "End-to-End Optimized Image Compression." arXiv, 2016, arxiv.org/abs/1611.01704.

Example Autoencoder Architecture



Ballé, Johannes, et al. "Variational Image Compression with a Scale Hyperprior." arXiv, 2018, arxiv.org/abs/1802.01436.

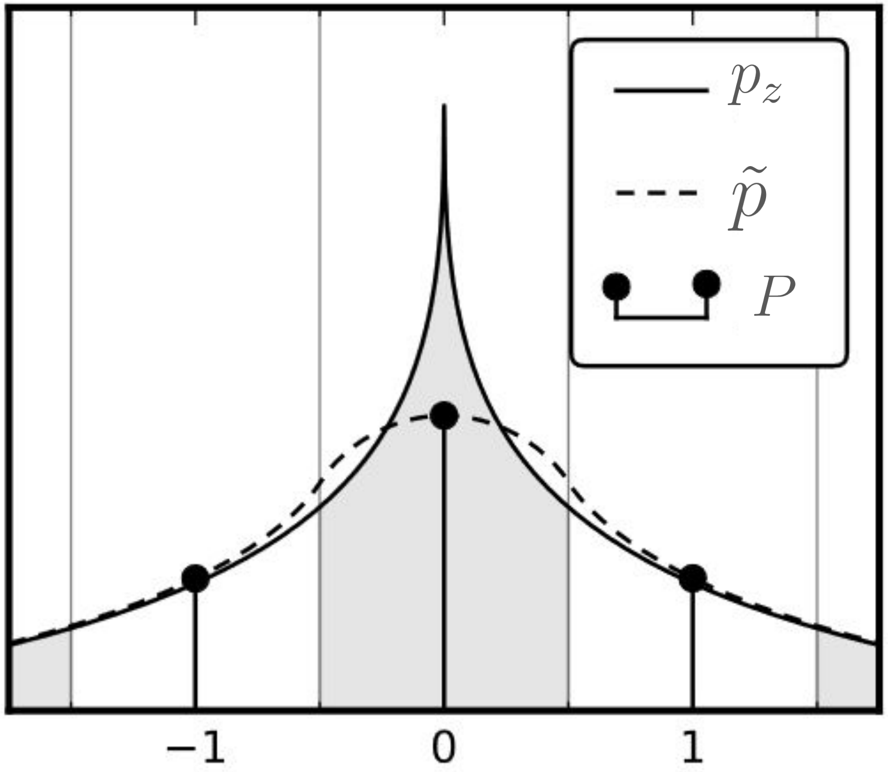
End-to-End Learning Quantization Problem

- Loss function depends on quantized data:

$$\min_{(g_a, g_s, P)} \mathbb{E}[-\log P(\hat{z})] + \lambda \mathbb{E}[d(x, \hat{x})]$$

- Quantized data is discrete
- Entropy model is discrete (PMF)
- Discrete data -> gradient = 0 almost everywhere
- Gradient descent is ineffective

End-to-End Learning Quantization Problem



- Add uniform noise to z to approximate quantized data

$$\hat{z} \approx \tilde{z} = z + u \qquad u \sim \mathcal{U}(-\frac{1}{2}, \frac{1}{2})$$

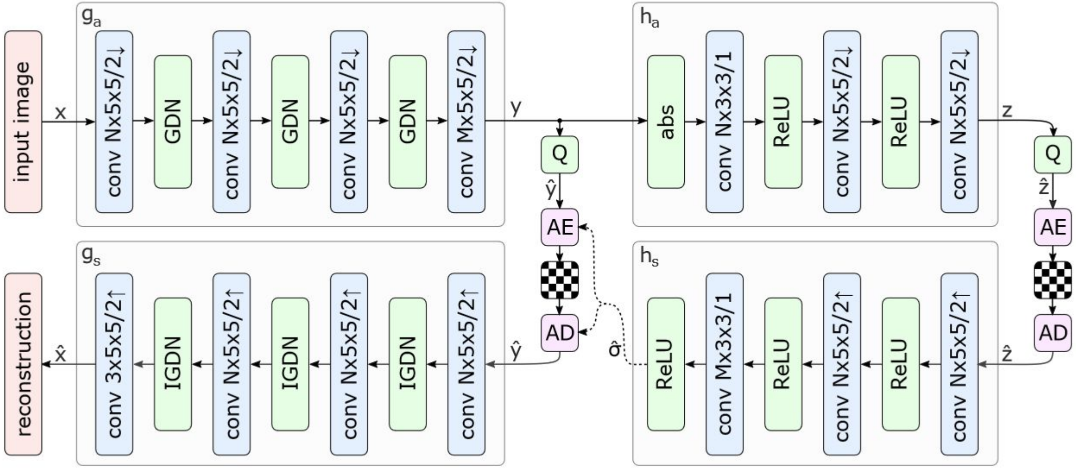
$$\tilde{p}(\cdot) = P(\cdot)$$

$$\min_{(g_a, g_s, \tilde{p})} \mathbb{E}[-\log \tilde{p}(\tilde{z})] + \lambda \mathbb{E}[d(x, \tilde{x})]$$

Objective function for SGD

Ballé, Johannes, Valero Laparra, and Eero P. Simoncelli. "End-to-End Optimized Image Compression." arXiv, 2016, arxiv.org/abs/1611.01704.

Example Variational Autoencoder Architecture



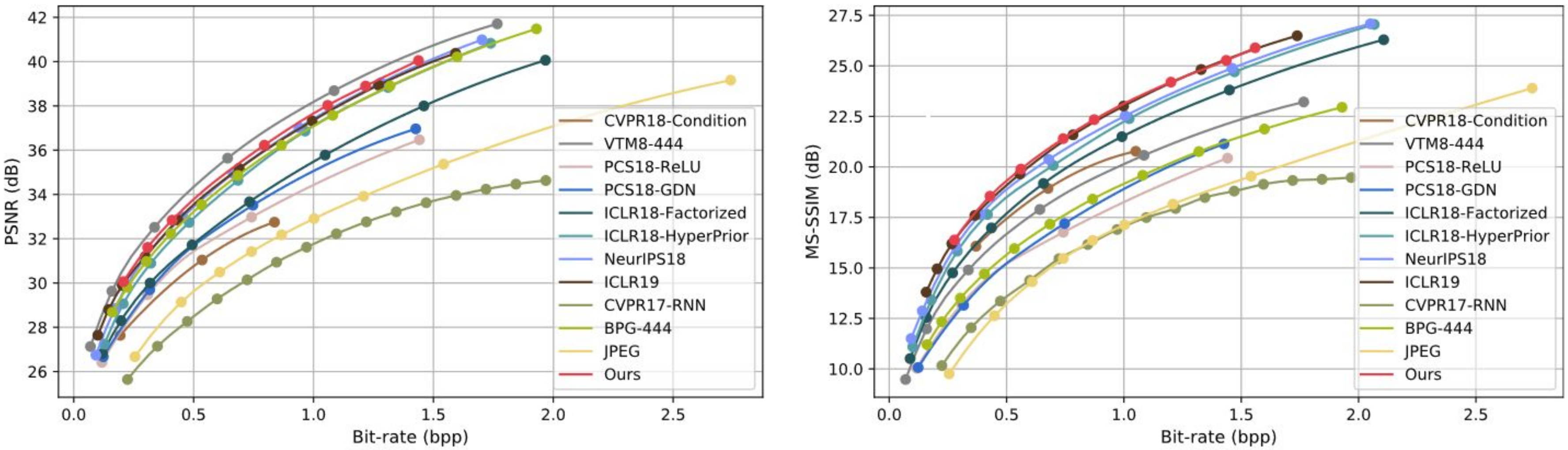
$$y \sim \mathcal{N}(0, \sigma^2)$$

$$\mathbb{E}_{\boldsymbol{x}\sim p_{\boldsymbol{x}}} D_{\mathrm{KL}}[q \parallel p_{\tilde{\boldsymbol{y}}|\boldsymbol{x}}] = \mathbb{E}_{\boldsymbol{x}\sim p_{\boldsymbol{x}}} \mathbb{E}_{\tilde{\boldsymbol{y}}\sim q}\Big[\cancelto{0}{\log q(\tilde{\boldsymbol{y}} \mid \boldsymbol{x})} \underbrace{- \log p_{\boldsymbol{x}|\tilde{\boldsymbol{y}}}(\boldsymbol{x} \mid \tilde{\boldsymbol{y}})}_{\text{weighted distortion}} \underbrace{- \log p_{\tilde{\boldsymbol{y}}}(\tilde{\boldsymbol{y}})}_{\text{rate}}\Big] + \text{const.} \quad (3)$$

Ballé, Johannes, et al. "Variational Image Compression with a Scale Hyperprior." arXiv, 2018, arxiv.org/abs/1802.01436.

Current Research and State-of-the-Art Techniques

Rate-Distortion Curves of SOTA Models



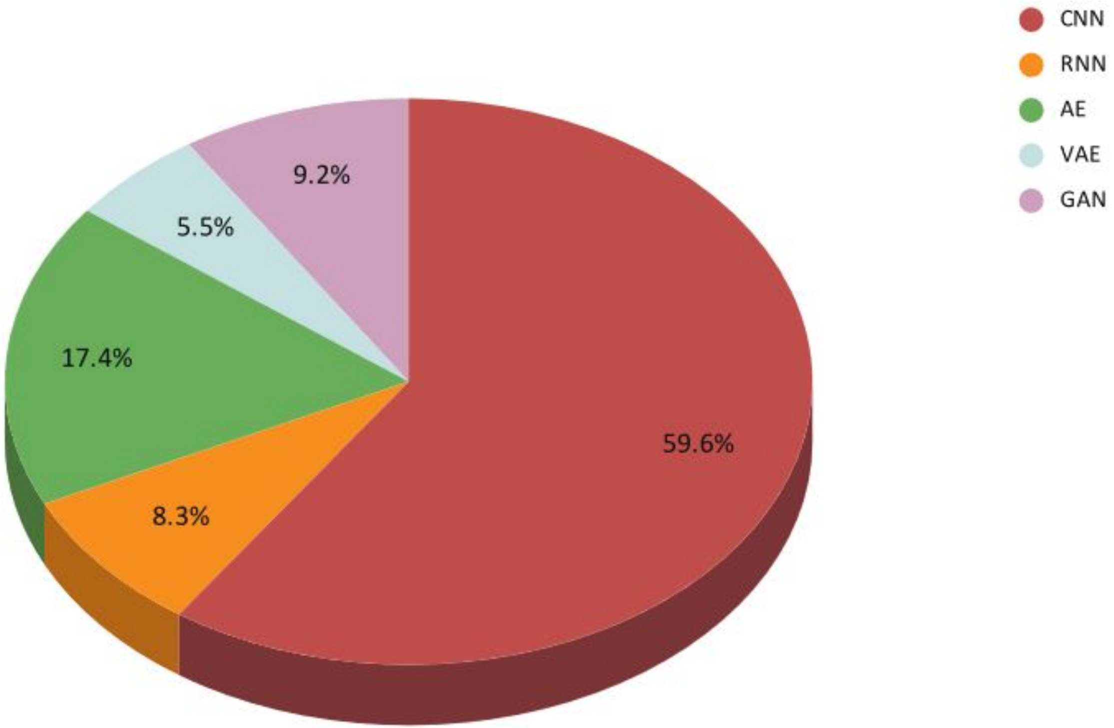
(a) Kodak, PSNR

(b) Kodak, MS-SSIM

The limit of how much we can improve is R(D)

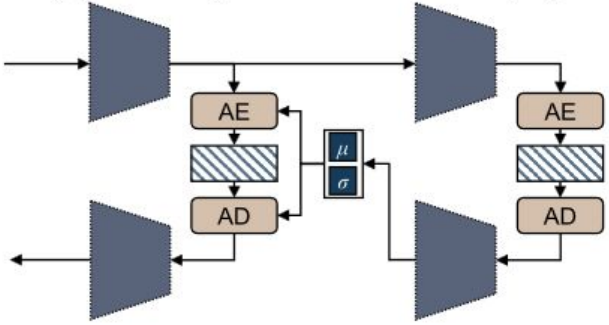
Hu, Yueyu, et al. "Learning End-to-End Lossy Image Compression: A Benchmark." arXiv, 2020, arxiv.org/abs/2002.03711.

Breakdown of Research by Model Architecture

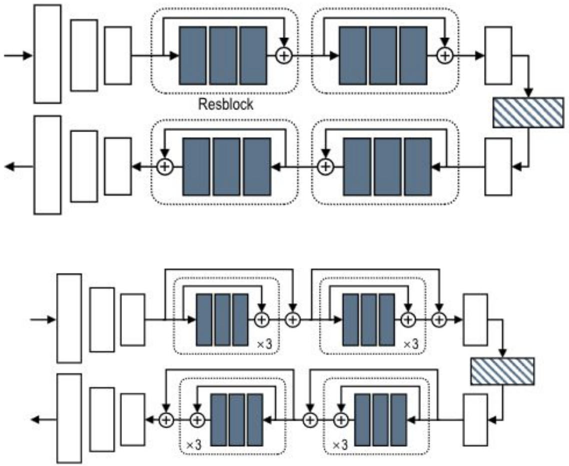


Mishra, Dipti, Satish Kumar Singh, and Rajat Kumar Singh. "Deep Architectures for Image Compression: A Critical Review." Signal Processing, vol. 191, 2022, article 108346, doi:10.1016/j.sigpro.2022.108346.

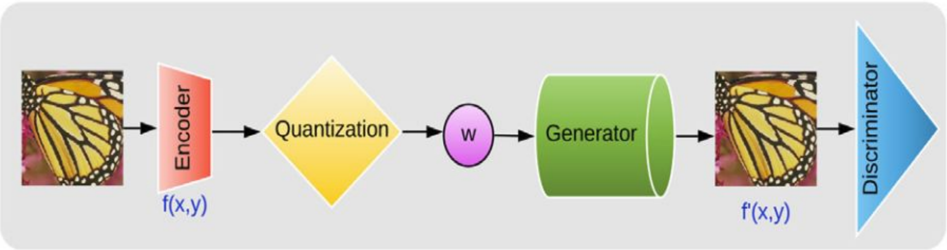
Generic Architectures



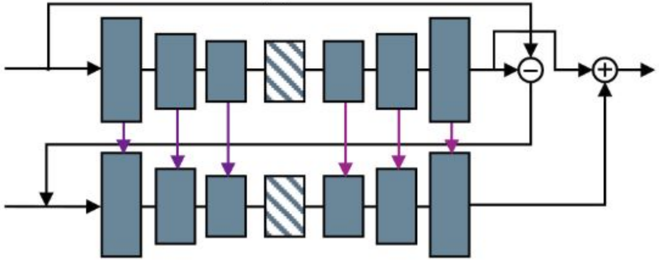
(V)AE



CNN



GAN



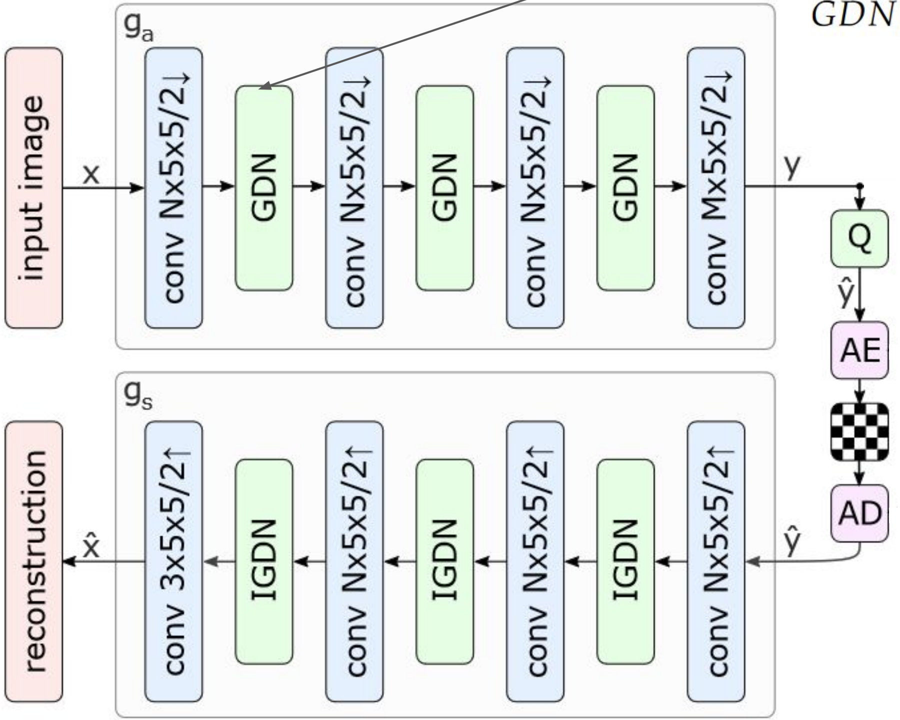
RNN

Hu, Yueyu, et al. "Learning End-to-End Lossy Image Compression: A Benchmark." arXiv, 2020, arxiv.org/abs/2002.03711.
Mishra, Dipti, Satish Kumar Singh, and Rajat Kumar Singh. "Deep Architectures for Image Compression: A Critical Review." Signal Processing, vol. 191, 2022, article 108346, doi:10.1016/j.sigpro.2022.108346.

Notable Works

Balle et al. 2017 (AE)

Generalized Divisive Normalization



$$GDN(v_i(k,l)) = \frac{v_i(k,l)}{(\beta_i + \sum_{j=1}^{N} \gamma_{ij} v_j^2(k,l))^{1/2}} \text{ for } i = 1, \ldots, N.$$

- GDN Intuition: decorrelates different features, i.e. reduces redundancy
- Better estimates optimal transform b.c. non-linear

Balle et al. 2017

JPEG, 4283 bytes (0.121 bit/px), PSNR: luma 24.85 dB/chroma 29.23 dB, MS-SSIM: 0.8079

JPEG 2000, 4004 bytes (0.113 bit/px), PSNR: luma 26.61 dB/chroma 33.88 dB, MS-SSIM: 0.8860

Proposed method, 3986 bytes (0.113 bit/px), PSNR: luma 27.01 dB/chroma 34.16 dB, MS-SSIM: 0.9039

Balle et al. 2017

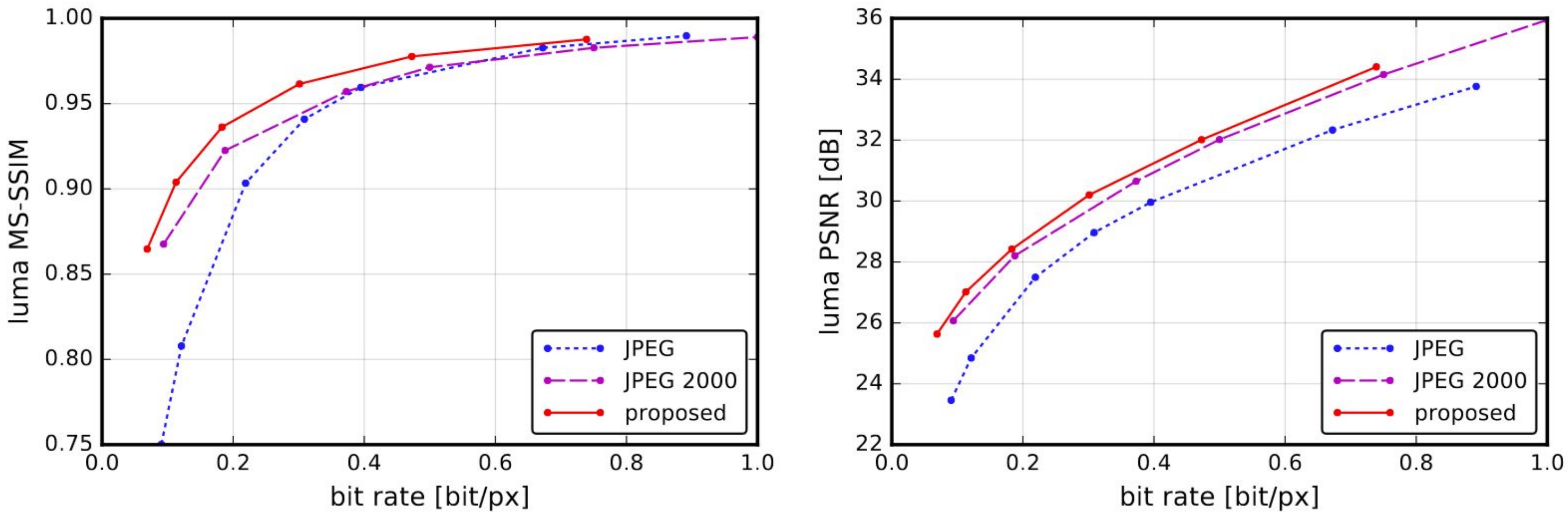
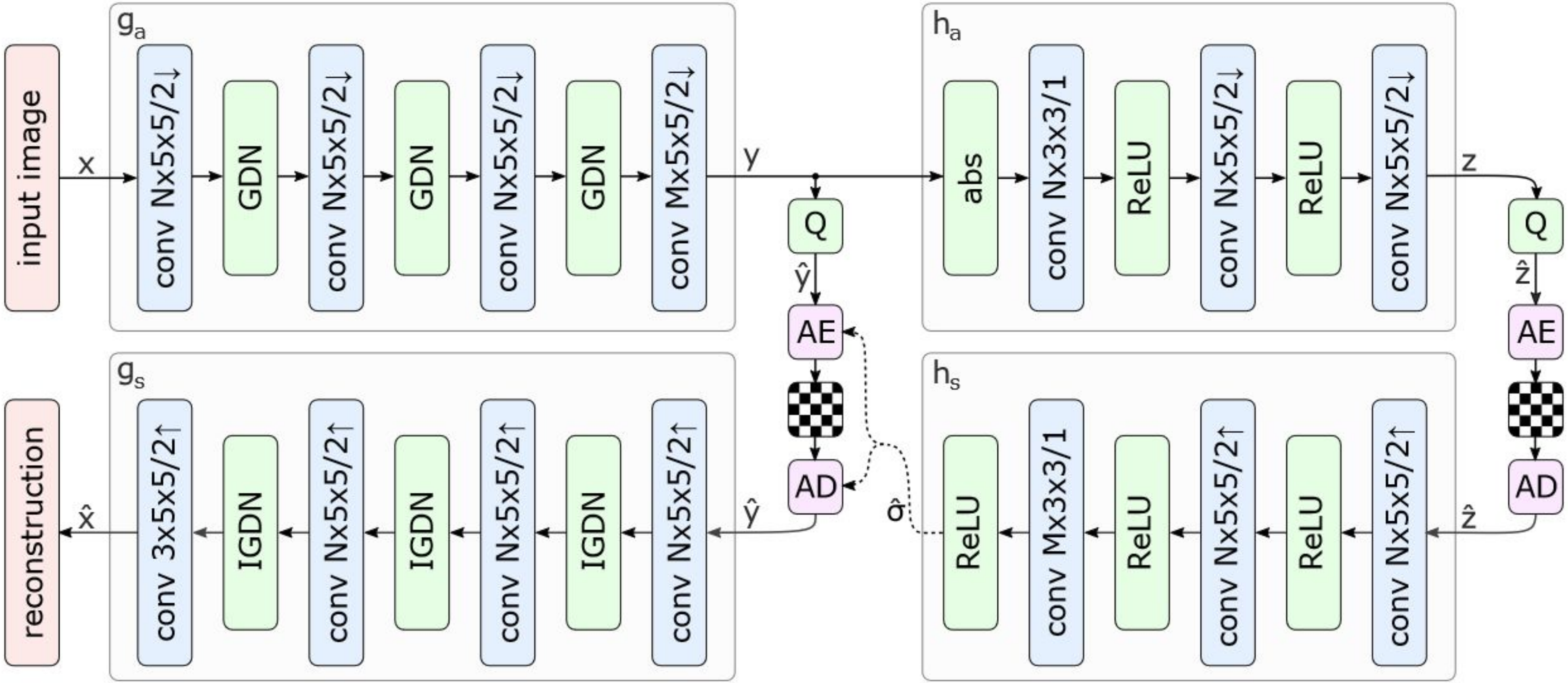


Figure 7: Rate–distortion curves for the luma component of image shown in figure 5. Left: perceptual quality, measured with multi-scale structural similarity (MS-SSIM; Wang, Simoncelli, and Bovik (2003)). Right: peak signal-to-noise ratio ($10 \log_{10}(255^2/\mathrm{MSE})$).

Balle et al. 2018 (VAE)

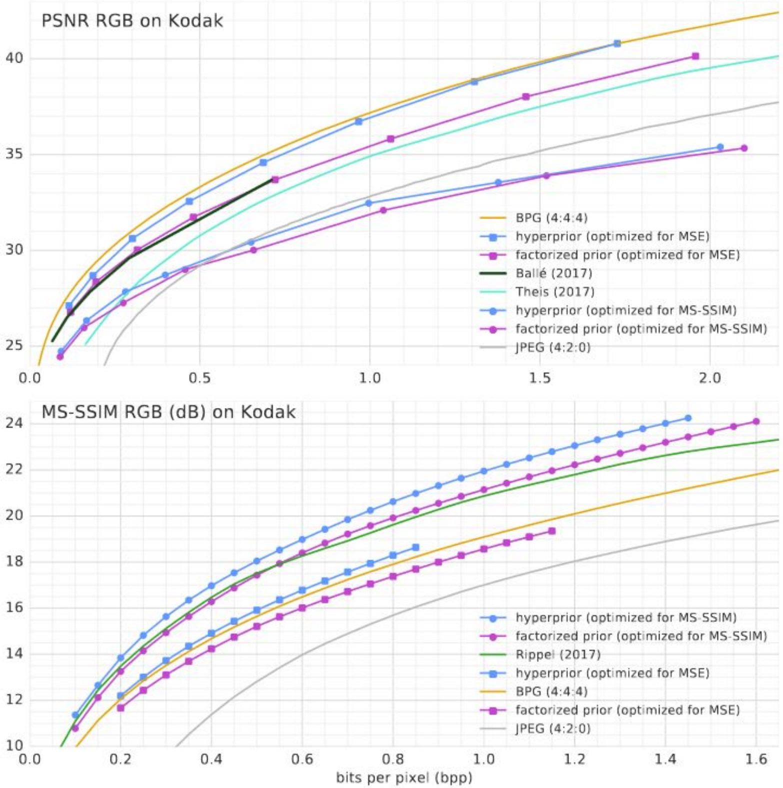


Balle et al. 2018

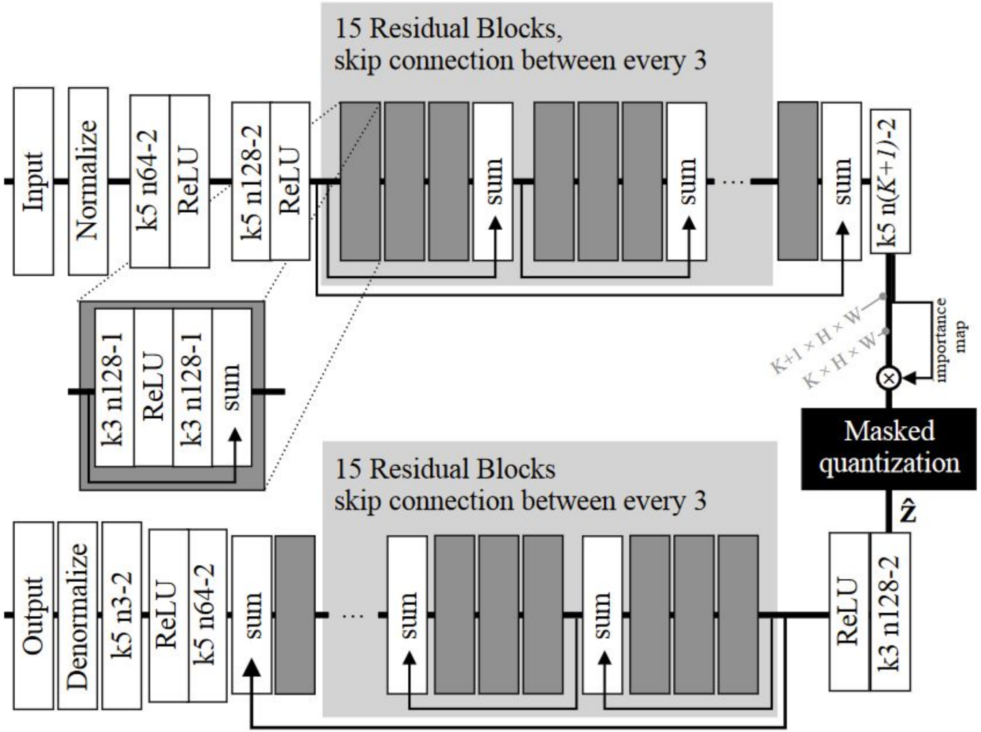
(0.1864 bpp, PSNR=27.99, MS-SSIM=0.9803)
trained on MS-SSIM

(0.1932 bpp, PSNR=32.26, MS-SSIM=0.9713)
trained on MSE

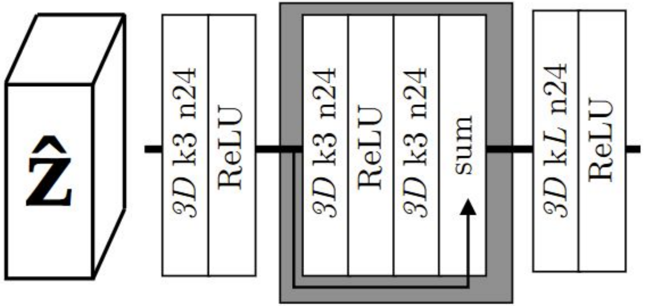
Balle et al. 2018



Mentzer et al. 2019 (CNN)

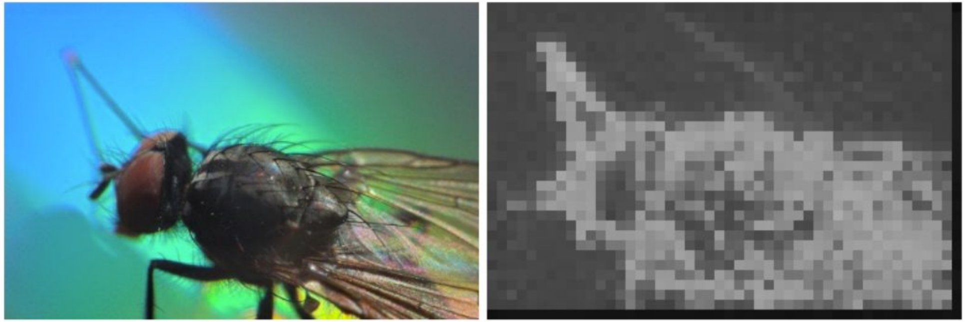


- Deeper NN with resblocks and skip connections
- Uses learned importance map for efficient bit allocation
- Uses 3D-CNN context model to estimate entropy



Mentzer et al. 2019

Input | Importance map of M



$$\mathbf{y} \in \mathbb{R}^{\frac{W}{8} \times \frac{H}{8} \times 1} \longrightarrow \mathbf{m} \in \mathbb{R}^{\frac{W}{8} \times \frac{H}{8} \times K}$$

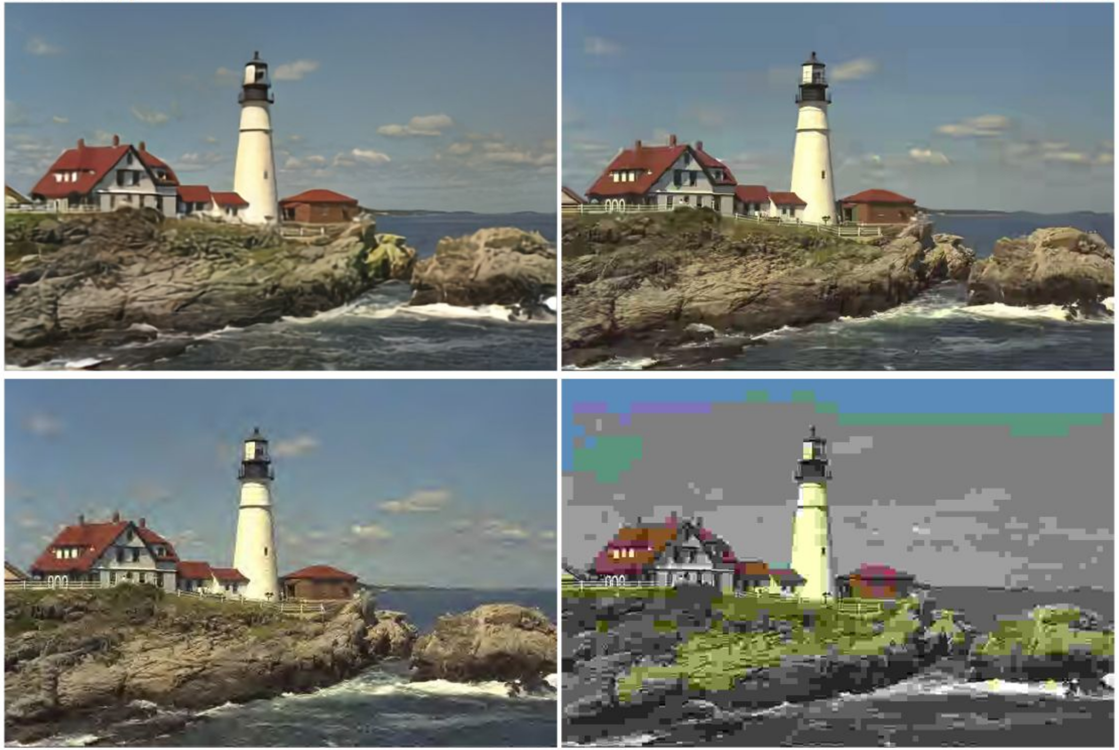
$$m_{i,j,k} = \begin{cases} 1 & \text{if } k < y_{i,j} \\ (y_{i,j} - k) & \text{if } k \leq y_{i,j} \leq k+1 \\ 0 & \text{if } k+1 > y_{i,j} \end{cases}$$

$$\mathbf{z} \leftarrow \mathbf{z} \odot \lceil \mathbf{m} \rceil$$

- As a result, parts of the feature map z are “zeroed out”
- Allocates more bits to important parts of the image

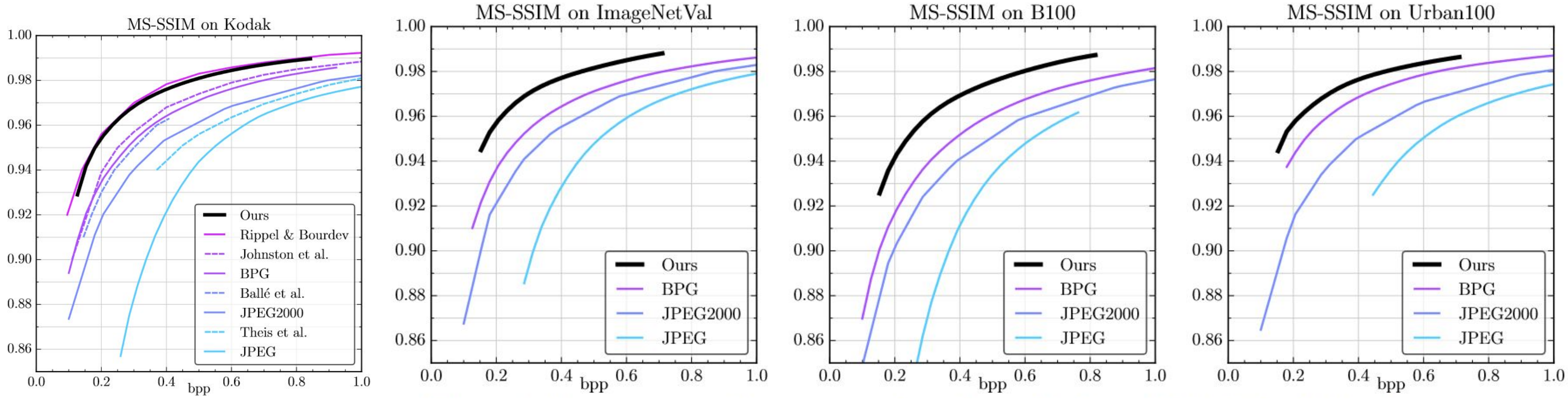
Mentzer et al. 2019

Ours 0.124bpp 0.147 bpp **BPG**

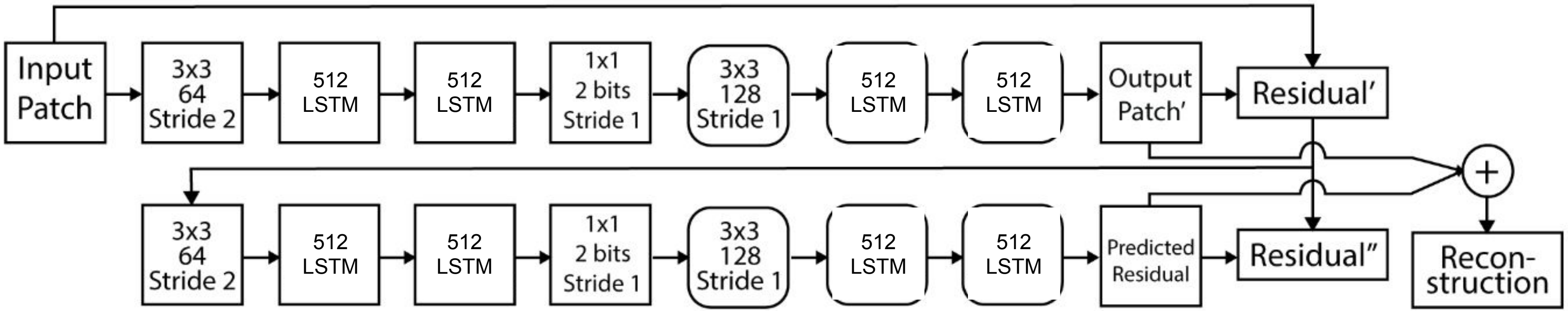


JPEG2000 0.134bpp 0.150bpp **JPEG**

Mentzer et al. 2019



Toderici et al. (RNN)



- Sharp rectangles: convolution layers, rounded rectangles: deconvolution layers
- Uses Long Short-Term Memory (LSTM) to learn sequential dependencies
- Does not explicitly use entropy coding, but it would improve compression for larger images

Toderici et al.

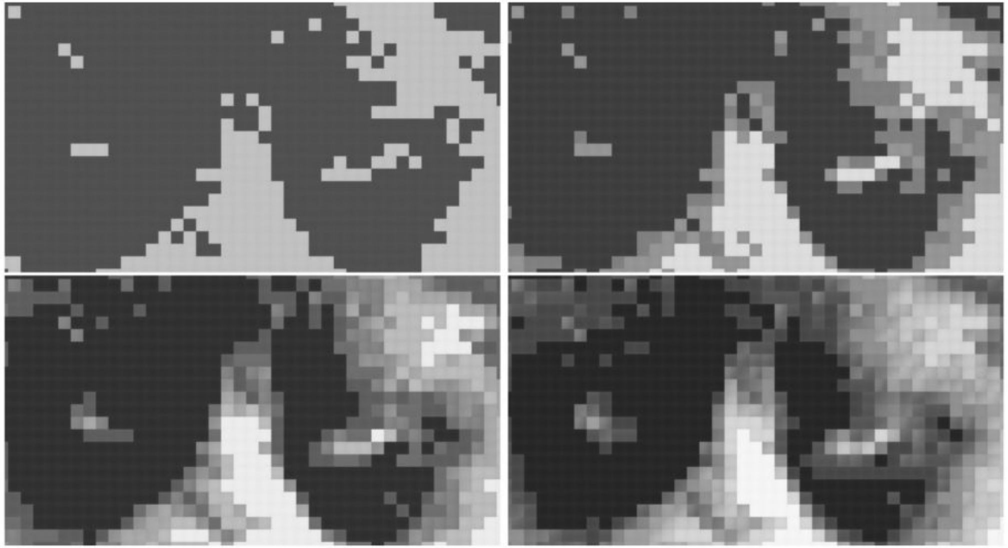
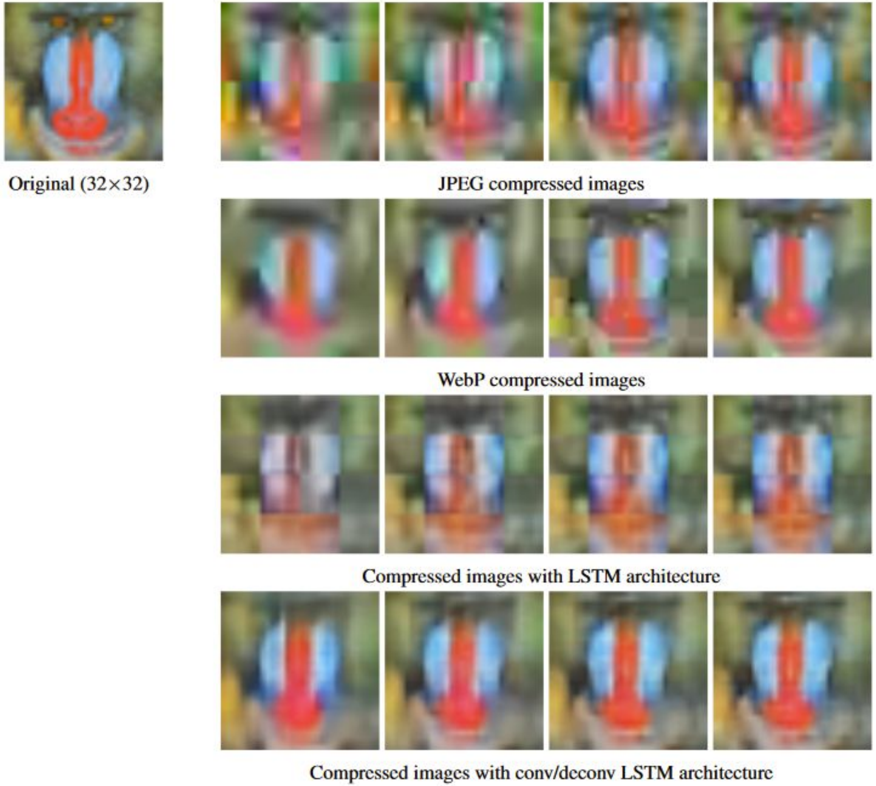


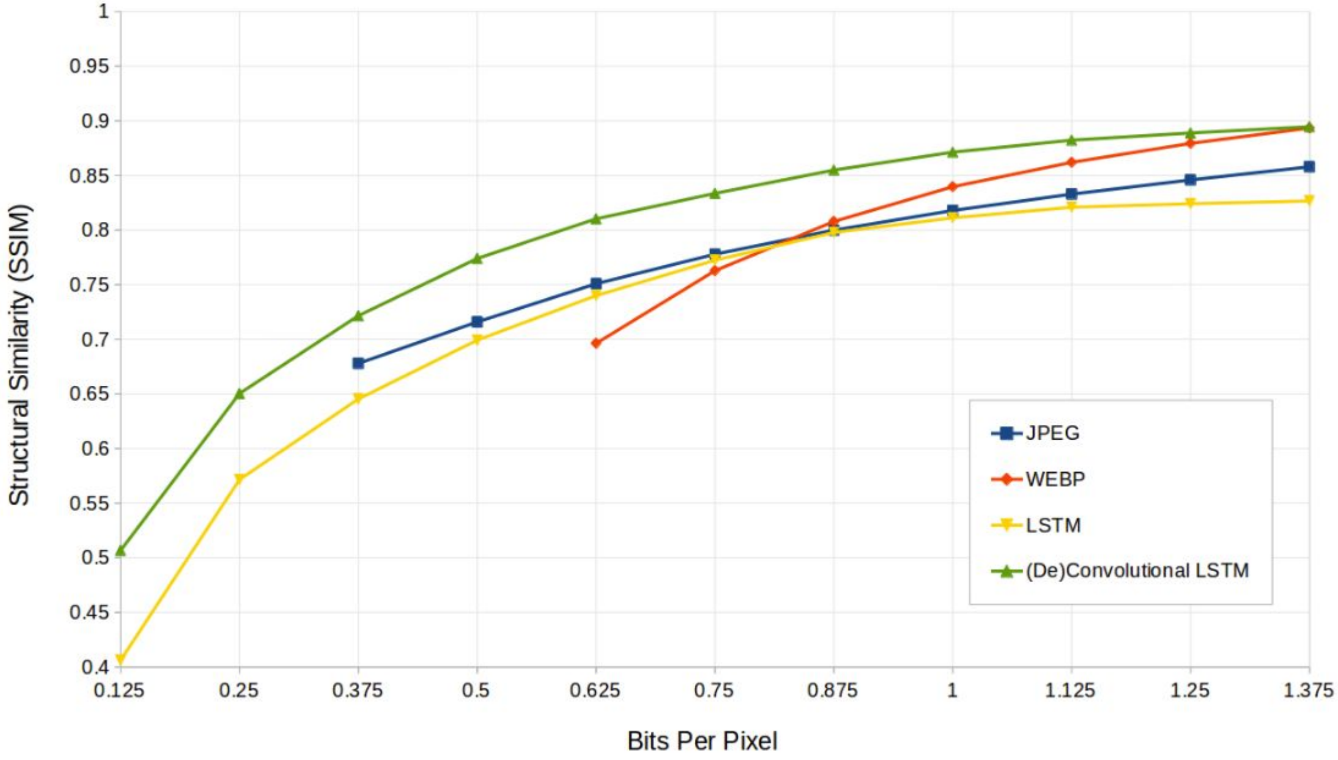
Figure 6: The effect of the first four bits on compressing a cat image. The image on the top left has been created by using a single bit for each 8×8 block. The subsequent images add one additional bit to be processed by the LSTM decoder (the ordering is top-left going to bottom-right). The final image (bottom right) has been created by running four steps of the algorithm, thus allowing a total of four bits to be used to encode each 8×8 block.

Toderici et al.



		From left to right			
Average bits per pixel (bpp)	JPEG	0.641	0.875	1.117	1.375
	WebP	0.789	0.914	1.148	1.398
	LSTM	0.625	0.875	1.125	1.375
	(De)Convolutional LSTM	0.625	0.875	1.125	1.375

Toderici et al.



Mentzer et al. (GAN)

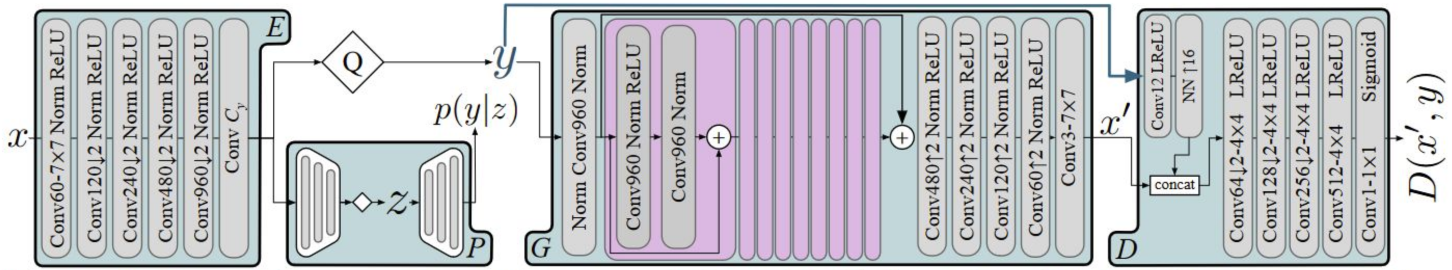
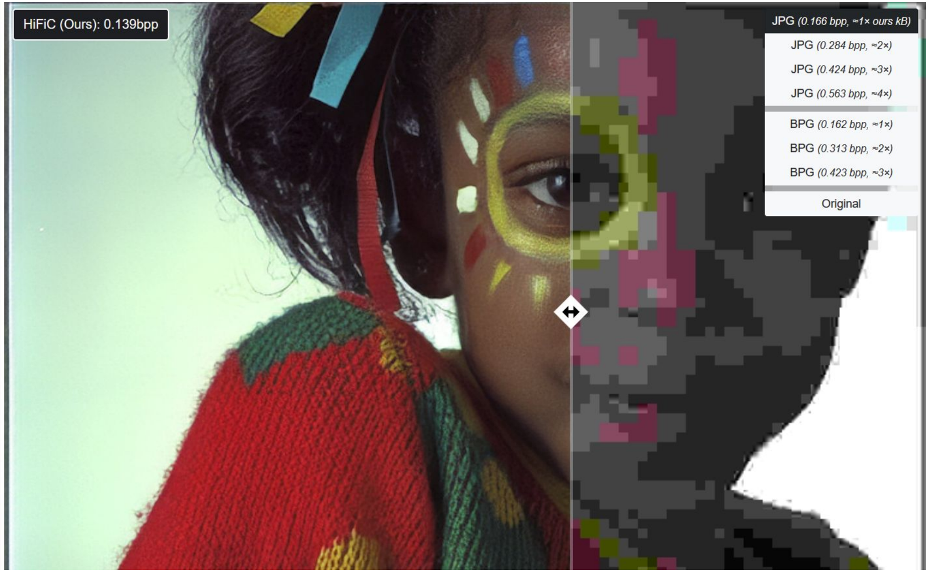
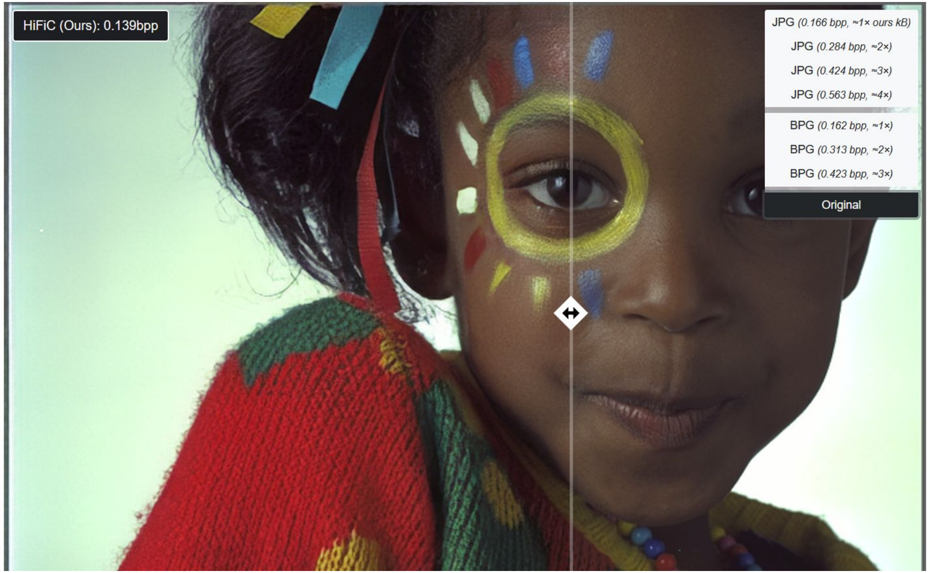


Figure 2: Our architecture. *ConvC* is a convolution with C channels, with 3×3 filters, except when denoted otherwise. ↓ 2, ↑ 2 indicate strided down or up convolutions. *Norm* is ChannelNorm (see text), *LReLU* the leaky ReLU [56] with α=0.2, *NN*↑16 nearest neighbor upsampling, Q quantization.

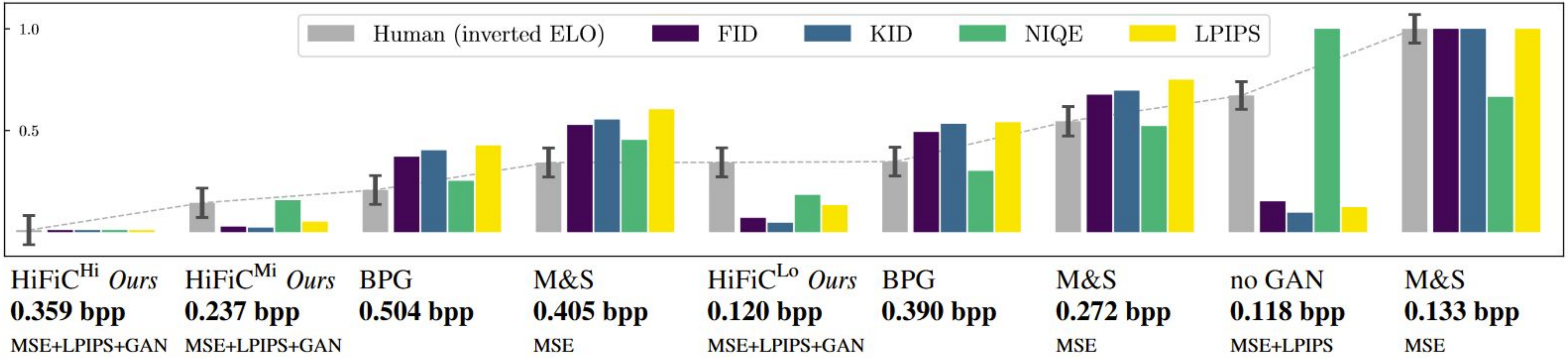
- Encoder E, hyperprior entropy model P, generator G, discriminator D
- Use a combination loss function
    - Rate-distortion function
    - Adversarial (GAN) loss
    - MSE
    - LPIPS

Mentzer et al.





Mentzer et al. (GAN)

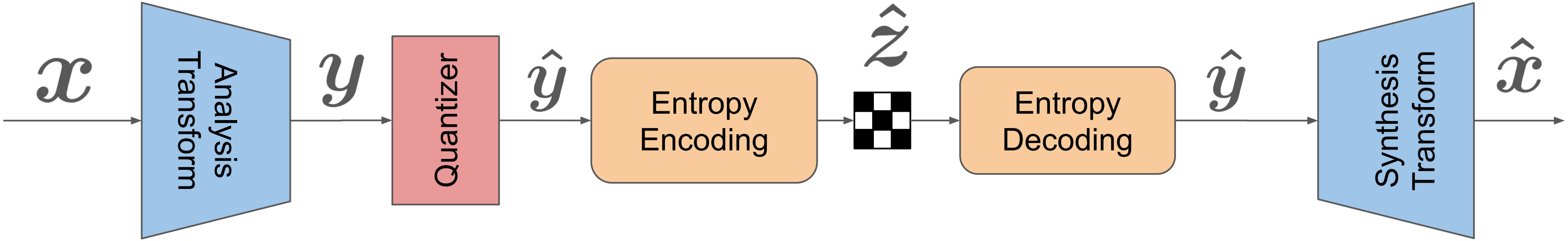


Discussion

Architecture Comparison

Architecture	Advantages	Challenges
AE	• Simple to implement • Easy to train • Small size, fast processing	• Cannot capture complex patterns • Blurry reconstruction
VAE	• Easy to train • Small size, fast processing • Better generalization than AEs	• More complex than AEs, require more computation • Blurry reconstruction
CNN	• Can capture more complex patterns • Fast processing • High performance (low rate)	• Can be very large and computationally expensive • Prone to overfitting
RNN	• Produce variable bit-rates • Can be used for video compression	• Prolonged training leads to distorted behavior • Slow training • Not naturally suited for still image processing, less efficient
GAN	• High-quality, sharp reconstructions even at low bit-rates • High compression efficiency	• Very large size and high computational costs • Complex architecture and difficult to train • Can “hallucinate” features not present in original image

Summary



$$\min_{(g_a, g_s, P)} \mathbb{E}[-\log P(\hat{z})] + \lambda \mathbb{E}[d(x, \hat{x})]$$

- Fundamental goal: rate-distortion optimization
- Transform coding is most commonly used
- Many possible model architectures: AE, VAE, CNN, RNN, GAN
- There is a limit to how much we can optimize rate-distortion