Supplementary: User-Guided Variable Rate Learned Image Compression

Rushil Gupta[1], Suryateja BV[1], Nikhil Kapoor[2], Rajat Jaiswal[2], Sharmila Nangi[3], Kuldeep Kulkarni[1]
[1]Adobe Research, Bengaluru, India
[2]Indian Institute of Technology Delhi, India
[3]Stanford University, USA
{rusgupta, surbv, kulkulka}@adobe.com {mt1170739, cs5170415}@iitd.ac.in srnangi@stanford.edu

1. Related Work

Traditional image compression standards, like JPEG [23], JPEG2000 [19], and HEVC [17] rely on hand-crafted modules involving discrete cosine transforms or wavelet transforms, quantization, and entropy coding. Traditional codecs like JPEG2000 compress on a per-instance level with no learning involved, thereby having poor performance at low bitrates with artifacts such as blurring, ringing.

To overcome the limitations of hand-crafted modules, several compression approaches based on RNNs [21, 22], and auto-encoders such as [1, 3, 4, 20] have been proposed, which construct visually pleasing images at high bitrates. Ideally, we expect compression methods to work effectively for all bitrates ranging from extremely low to considerably high depending on the storage budget available to a user. Some works also deal with variable-rate image compression [6, 8, 10, 22]. While [6] enables variable bitrate during test time, it is not very flexible as one may not have the knowledge of the knob parameters, λ and Δ, required to attain the desired bitrate. We propose a **single** model for a wide range of bitrates that ensures photo-realism, maintains the visual quality as per the bit budget allocated while facilitating the user with an explicit control on desired bitrate.

GANs [9] have led to rapid progress in compression algorithms they can generate photo-realistic images at extremely low bitrates [2, 14]. In [2], the authors successfully generate photo-realistic reconstructions by selectively compressing a random region in the image, and fully synthesizing other regions in the decoded image using GANs and a semantic label map. However, these methods synthesize parts of the image that may be completely different from the contents of original image. A compression algorithm is expected not to synthesize the contents of the image; thus our method avoids the full synthesis of any part. We use HiFiC [14], a GAN-based algorithm, as a baseline in our experiments.

Previous works such as [5, 11, 13, 16] assume that only a small region of the entire image is relatively more important than the other regions and is, therefore, the main area of focus. Li et. al. [11] also showed that the local information content varies spatially across the image, and the bitrate of different parts of an image should be adapted to local content. Inspired by this and the success of photo-realism of GAN reconstructions, we develop our auto-encoder and GAN-based hybrid model which facilitates a simpler and explicit bitrate control to user, and a differential bit allocation to different regions in the image guided by a user-provided importance map in contrast to a semantic label map used by [2]. We further propose a novel Equivalence Distortion Loss to maintain the output bitrate as close as possible to the target bitrate both in totality and across regions within an image.

2. Implementation Details

We provide a Pytorch-like pseudocode for the forward pass of our algorithm in Figure 1. We train our model with a discriminator. We utilize 4 Tesla V100 GPUs with 32 GB memory. We use Adam optimizer with learning rate of 0.0001 for both the model parameters and the discriminator parameters. We train all our models for 24 epochs and use a multi-step learning rate scheduler with a gamma of 0.1 and milestones at $[9, 15, 19, 22]$. Our training dataset consists of close to $24,100$ images. We train with a batch size of 16 and training one variant of our model takes close to 8 hours. A screenshot of our user interface is displayed in Figure 2. [1]

3. Additional Loss Functions

VGG Feature Loss: I and $\tilde{I}$ are both fed into the pre-trained VGG-16 network, and the features computed by the network at various intermediate layers are compared using an L_1 Loss. In our method, features are extracted from $i = 1...4$ layers in the VGG network and the corresponding L_1 losses are aggregated to L_{vgg} given by,

[1]A demo video of our user interface is provided in https://drive.google.com/file/d/1tLuDyLoU6wt9uMSdgUVqc3KGH5J1xsa/view

```
# Input I[B, C, H, W] - batch of images

I = torch.cat([I, pos_encoding], dim = 1)

# encoder - part 1
ec = encoder_1(I)       # [B, 64, H, W]

# importance map
im = importance_map(ec)# [B, 1, H/4, W/4]
mask = create_mask(im) # [B, 8, H/4, W/4]

# encoder - part 2
ec = encoder_2(ec)      # [B, 8, H/4, W/4]

# binarization
binary = binarizer(ec) # [B, 8, H/4, W/4]
ec = binary x mask      # [B, 8, H/4, W/4]

# decoder
dc = decoder(ec)        # [B, C, H, W]

return dc, im
```

Figure 1. Pseudocode for forward pass of our algorithm

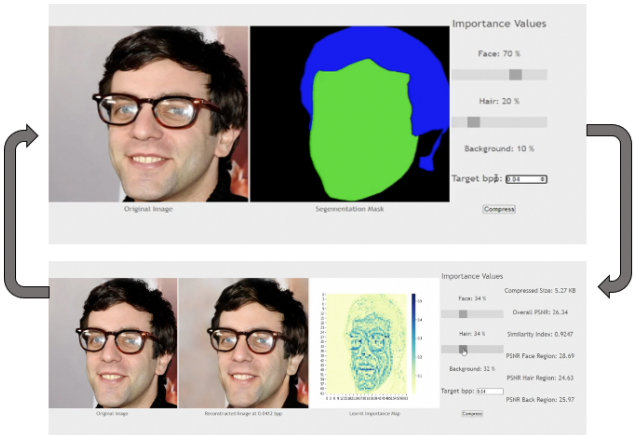
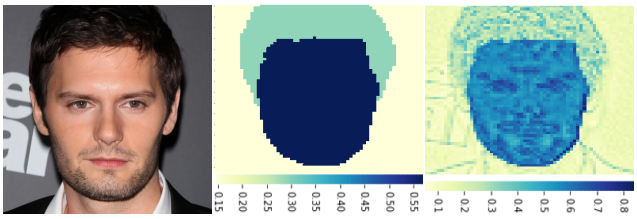


Figure 2. Our User Interface: Users can enter the relative importances for each region, and provide a desired bitrate based on their budget. If users are not satisfied with the reconstruction, they can iteratively adjust the importance.

$L_{vgg} = \sum_{i=1}^{4} L_1(\text{VGG}_i(\mathbf{I}), \text{VGG}_i(\tilde{\mathbf{I}}))$
where $\text{VGG}_i(\mathbf{I})$ gives the features computed by VGG network on input image I extracted at layer i.

Hinge Loss: Instead of the standard GAN loss, we use the hinge loss term in our loss function which is shown to produce better photorealistic samples [12, 15, 24].



(a) Input Image (b) User Input Map (c) Learned Map

Figure 3. User input importance values (b) are $[0.55, 0.3, 0.15]$ for face, hair, background respectively. In the learned importance map (c), the values get distributed in the corresponding regions. It is evident that pixels with high texture or edges within a region get greater weightage. The average learned importance values are $[0.572, 0.293, 0.145]$, thus showing that the input importance map was used effectively as a guidance while ensuring parts of greater variation within a region (eyes) have higher importance compared to smoother ones (cheeks).

GAN Feature Loss: Features learned by the discriminator for both of its inputs, i.e., the original image and the fake image, are compared with each other using L_1 Loss. This enforces the reconstructed image to be visually close to the input image so that the discriminator gets fooled and makes mistakes in its task of classifying whether an image is original/fake. Let D denote the discriminator. Then,

$$L_{gan} = L_1(\text{D}(\mathbf{I}), \text{D}(\tilde{\mathbf{I}})) \tag{1}$$

Equivalence Distortion (ED) Loss Function: Here is the mathematical formulation of the terms described in the ED Loss in the main paper (section 2.3). Let L_{mse} and L_{ssim} denote the mean-squared error and MS-SSIM between input and reconstructed images.

$$L_{mse} = m_a \odot ||I - \tilde{I}||^2 \tag{2}$$

$$L_{ssim} = 1 - \text{MS-SSIM}(I, \tilde{I}) \tag{3}$$

Then, the distortion loss L_D is given by:

$$L_D = \lambda_1 L_{mse} + \lambda_2 L_{ssim} \tag{4}$$

where $\lambda_1 = 1.25$ and $\lambda_2 = 0.1$
Let L_E denote the equivalence loss obtained when importance values of different regions between the user input importance map and learned importance map are compared. Then,

$$L_{whole} = max(0, \sum m_l - \sum m'_a) \quad (5)$$

$$(m_l^i)_{avg} = \sum m_l \odot s'_i / (\alpha + \sum s'_i), i = 1..h \quad (6)$$

$$(m_a'^i)_{avg} = \sum m'_a \odot s'_i / (\alpha + \sum s'_i), i = 1..h \quad (7)$$

$$L^1_{region} = \frac{1}{h} \sum_{i=1}^{h} max(0, (m_l^i)_{avg} - (m_a'^i)_{avg}) \quad (8)$$

$$L^2_{region} = \frac{1}{h} \sum_{i=1}^{h} max(0, (m_a'^i)_{avg} - (m_l^i)_{avg}) \quad (9)$$

$$L_E = L_{whole} + L^1_{region} + \lambda_3 L^2_{region} \quad (10)$$

where $\lambda_3 = 0.9$, h is the number of regions in an image, α is a smoothing constant with value 10^{-8} and $(m^i_*)_{avg}$ is the average importance to region i. Figure 3 further shows the effectiveness of the ED Loss function in making the learned importance map tightly follow the user-input importance values, thus allowing the output bitrate to be close to the input bitrate.

4. Architecture Details

In this section, we provide the architecture details of various components in our model. Table 1 contains the layers of the first part of our encoder that takes in the input image concatenated with importance map. Each convolution layer consists of a reflection padding and a 2D convolution, along with a Leaky ReLU activation. Table 2 contains the importance map network with its individual components, which takes the output of the first part of encoder as its input. Components of second part of the encoder are shown in Table 3. Decoder consists of ConvTranspose2D layers with kernel stride as $(2, 2)$ and stride as $(2, 2)$ and residual blocks. The decoder architecture is shown in Table 5. Residual blocks are used in each of our model components, and the layers of residual block are shown in Table 4. Multiscale-Discriminator, shown in Figure 4, consists of four convolution blocks operating at progressively lower scales.

Layer	In Dimensions	Out Dimensions
conv_0	[C, H, W]	[64, H, W]
conv_1	[64, H, W]	[128, H/2, W/2]
conv_2	[128, H/2, W/2]	[256, H/4, W/4]
res_block_0	[256, H/4, W/4]	[256, H/4, W/4]
res_block_1	[256, H/4, W/4]	[256, H/4, W/4]

Table 1. Layers of Encoder - Part 1

Layer	In Dimensions	Out Dimensions
layer_0	[256, H/4, W/4]	[512, H/4, W/4]
im_res_block_0	[512, H/4, W/4]	[512, H/4, W/4]
layer_1	[512, H/4, W/4]	[1024, H/4, W/4]
im_res_block_1	[1024, H/4, W/4]	[1024, H/4, W/4]
layer_2	[1024, H/4, W/4]	[1, H/4, W/4]

Table 2. Layers of Importance Map Network

Layer	In Dimensions	Out Dimensions
conv_3	[256, H/4, W/4]	[512, H/4, W/4]
res_block_2	[512, H/4, W/4]	[512, H/4, W/4]
res_block_3	[512, H/4, W/4]	[512, H/4, W/4]
conv_4	[512, H/4, W/4]	[8, H/4, W/4]

Table 3. Layers of Encoder - Part 2; where $C/n^2 = 8/16 = 0.5$

Layer	In Dimensions	Out Dimensions	Kernel Size	Stride
layer_0	[C, H, W]	[C, H, W]	(3, 3)	(1, 1)
layer_1	[C, H, W]	[C, H, W]	(3, 3)	(1, 1)

Table 4. Each Residual Block

Layer	In Dimensions	Out Dimensions
deconv_0	[8, H/4, W/4]	[512, H/2, W/2]
res_block_0	[512, H/2, W/2]	[512, H/2, W/2]
res_block_1	[512, H/2, W/2]	[512, H/2, W/2]
dconv_1	[512, H/2, W/2]	[256, H, W]
res_block_0	[256, H, W]	[256, H, W]
res_block_1	[256, H, W]	[256, H, W]
dconv_2	[256, H, W]	[128, H, W]
dconv_3	[128, H, W]	[3, H, W]

Table 5. Layers of Decoder

5. Analysis of Input Bitrate

In this section, we analyze how the effective input bitrate varies as user-provided input bitrate t and relative importance values change. Suppose we have an image I of size (H, W), and compressed latent representation of size $(H/n, W/n)$ with C channels to hold the bits. Without loss of generality, let us consider three regions with areas A_1, A_2, A_3 in the original image, and $A_1^{(r)}, A_2^{(r)}, A_3^{(r)}$ in the latent representation. Let the proportion of their areas in the image be k_1, k_2, k_3. During training, we sample input bitrate t and importance values $[p_1, p_2, p_3]$ from Uniform distribution and Dirichlet distribution respectively as a proxy for user-provided values, given by $t \sim \mathrm{U}(0.01, 0.5)$ and $\mathbf{p} \sim \mathrm{Dir}([1.0, 1.0, 1.0])$

Our goal is to obtain weights (absolute importance values)

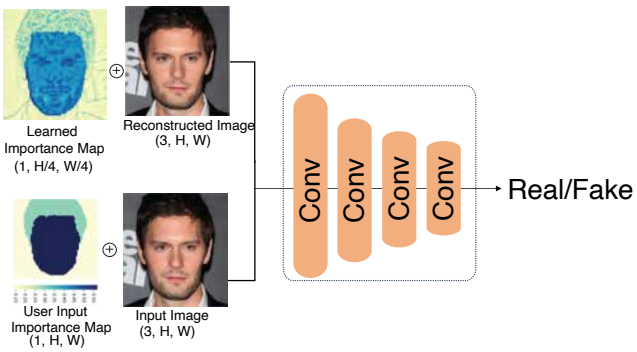


Figure 4. Multiscale Discriminator takes in the input image concatenated with input importance map, and reconstructed image concatenated with learned importance map. It predicts which of the images is real or fake.

$[w_1, w_2, w_3]$, each of which corresponds to the fraction of non-zero bits in $A_i^{(r)}C$. Thus, we require each $w_i \leq 1$ and $t \times H \times W = w_1 A_1^{(r)} C + w_2 A_2^{(r)} C + w_3 A_3^{(r)} C$.
The right hand side of equation represents the total bits in latent representation [2]. Simplifying, we get

$$t = (w_1 k_1 + w_2 k_2 + w_3 k_3) \times \frac{C}{n^2} \tag{11}$$

When all the weights are equal to 1, we get the maximum bitrate possible via our method, that is, C/n^2. In all our models, we ensure that C/n^2 is equal to 0.50. For any other combination of weights, we get a bitrate t less than 0.50.
Let us assume that the relative importance values correspond to the relative proportions of weights. Then, $t = k(p_1 k_1 + p_2 k_2 + p_3 k_3) \times \frac{C}{n^2}$, $k = \frac{t}{C/n^2 \times (\sum_{i=1}^{3} p_i k_i)}$ and

$$w_i = \min(1, \frac{t p_i}{C/n^2 \times (\sum_{i=1}^{3} p_i k_i)}) \qquad \forall i \tag{12}$$

Since $t \leq 0.5$, $p_i \leq 1$, $C/n^2 = 0.5$, and $\sum_{i=1}^{3} p_i k_i \leq 1$, the value of w_i can go higher than 1, making it necessary to clip w_i. Due to such clipping, we sometimes don't match the input bitrate t exactly.
While our method of controlling the output bitrate is easier as compared to previous methods, we suffer a residue Δt for high input bitrates to incorporate user-guided importance values. We intend to make the bounds tighter in future work. One way is to allocate residual bits to other regions in inverse proportion of their areas.

6. Discussion on Input Bitrate

In this section, we explain our choice of sampling with Uniform distribution for target bitrate and Dirichlet distribution for relative importance values. We wanted a generative approach for training that represents how a user might interact with a compression system - by providing a target bpp and a choice of relative importance values. We provide three cases of data generation process during training.

6.1. Case (1)

We sample the raw weights $[w_1, w_2, w_3]$ for input importance map from a Unif$(0, 1)$ distribution. We know that the effective input bpp is given by

$$(w_1 k_1 + w_2 k_2 + w_3 k_3) \times \frac{C}{n^2}$$

We ensure that the effective input bpp is always less than or equal to C/n^2 with our current sampling procedure. We sample the CelebA-HQ faces dataset for multiple epochs and take an average across epochs to get the distribution of effective input bpps and raw importance values. The distribution is plotted in Figure 5. It is evident that the weights are all uniform. However, the effective input bpp peaks around 0.25 and tapers on both the extremes. This is explained by the fact that the effective input bpp is the sum of three scaled random uniform variables - and is expected to look close to a Normal distribution [7]. So, our models are trained well for bpps in the range $[0.15, 0.35]$ but perform poorly for bpps outside this range.

6.2. Case (2) - No Clipping

Here, we explicitly sample the target bpp from a Uniform distribution and relative importance values from a Dirichlet distribution. A Dirichlet distribution with concentration parameters $\alpha = [1.0, 1.0, 1.0]$ ensures that all relative importance triplets are sampled with equal probability. As shown in Figure 6, we do achieve uniformity in the effective input bpps ensuring that we can train the models well for all bpps; but the weights for importance values go out of the $[0, 1]$ range. For example, the weight for hair region takes a maximum value of 7.7 for some images, and the weight for background region takes a maximum value of 6.4. Note that the equation for weights is given by,

$$w_i = \frac{t p_i}{C/n^2 \times (\sum_{i=1}^{3} p_i k_i)} \qquad \forall i$$

When t is large, and $\sum_{i=1}^{3} p_i k_i$ is small, weights can shoot much higher than 1. For instance, when $t = 0.4$, sampled relative importance values are $[0.5, 0.9, 0.35]$, and the proportion of areas are $[0.6, 0.005, 0.395]$, the weight for hair region ends up being 13.27, larger than 1. Such weights don't make sense when computing the channel masks in our model.

[2] The formula for bitrate or bits-per-pixel is given by, $t = \frac{\text{total bits in latent rep}}{\text{total number of pixels}} = \frac{w_1 A_1^{(r)} C + w_2 A_2^{(r)} C + w_3 A_3^{(r)} C}{HW}$

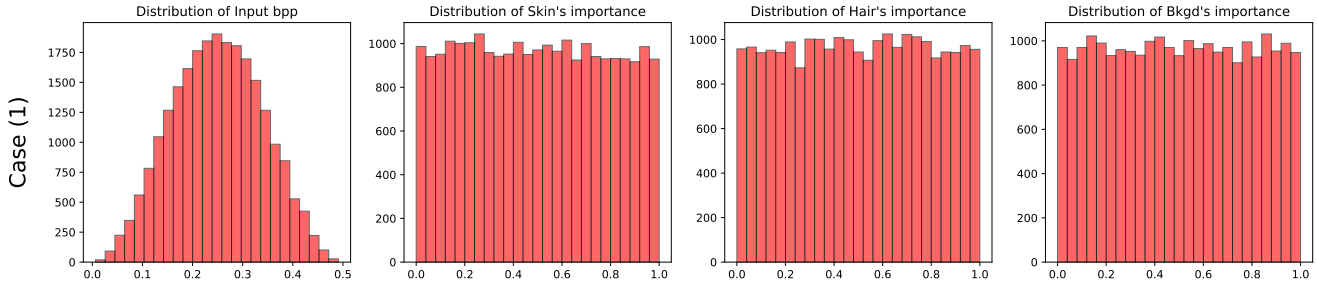


Figure 5. Case (1): Sampling each raw importance value (weight) from a uniform distribution

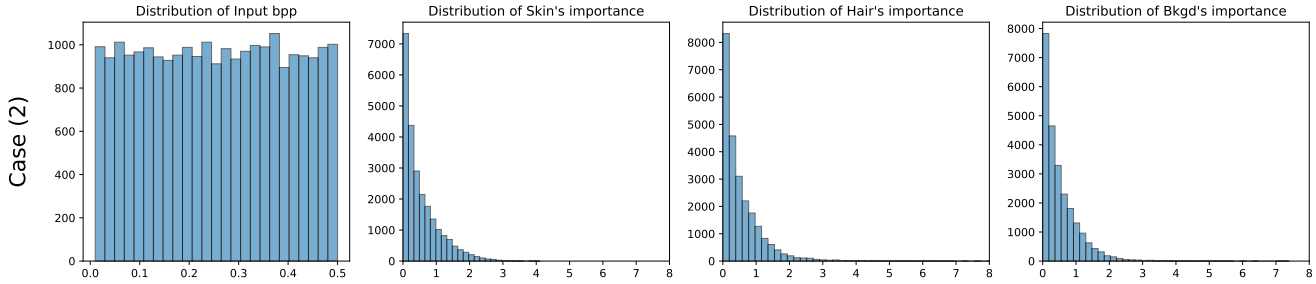


Figure 6. Case (2): Sampling the target bpp from a uniform distribution and relative importance values from a Dirichlet Distribution with concentration parameter = 1.0; no clipping of weights

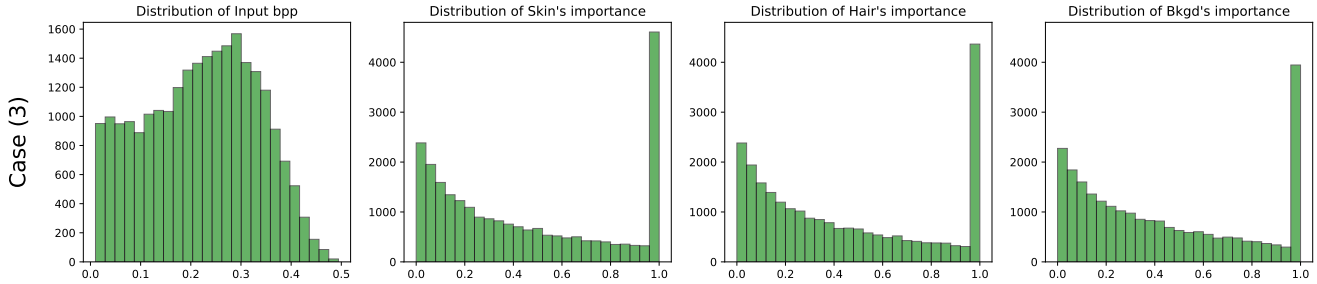


Figure 7. Case (3): Sampling the target bpp from a uniform distribution and relative importance values from a Dirichlet Distribution with concentration parameter = 1.0; **and** clipping the weights. We use this method in the current work

6.3. Case (3) - With Clipping

We sample the weights similar to Case (2), but clip the weights. This again skews the effective input bpp distribution - we see a tapering for higher target bpps. This is expected given the clipping of weights. Higher target bpps (> 0.4) typically lead to weights larger than 1, which in turn get clipped, leading to a tapering at one extreme. So, our models are well trained for bpps in the range $[0.05, 0.35]$, which is larger than Case (1), but perform poorly for higher bpps. The deviation from the expected $y = x$ line at the extremes is higher. This phenomenon can also be seen in other evaluation metrics, where the performance drops for larger bpps.

6.4. Residual Bitrate Δt

In Figure 8, we keep p_2 fixed at 3 different values $[0, 1/3, 2/3]$ and vary p_1 in the range $[0, 1 - p_2]$. We observe that low bitrates (≤ 0.25) have $\Delta t = 0$. Higher bitrates coupled with extreme relative importances get clipped and lead to a gap. For instance, in the first plot, when $[p_1, p_2, p_3] = [0.1, 0.0, 0.9]$ at input bitrate $t = 0.35$, we see that Δt is approximately 0.15. In the third plot, the

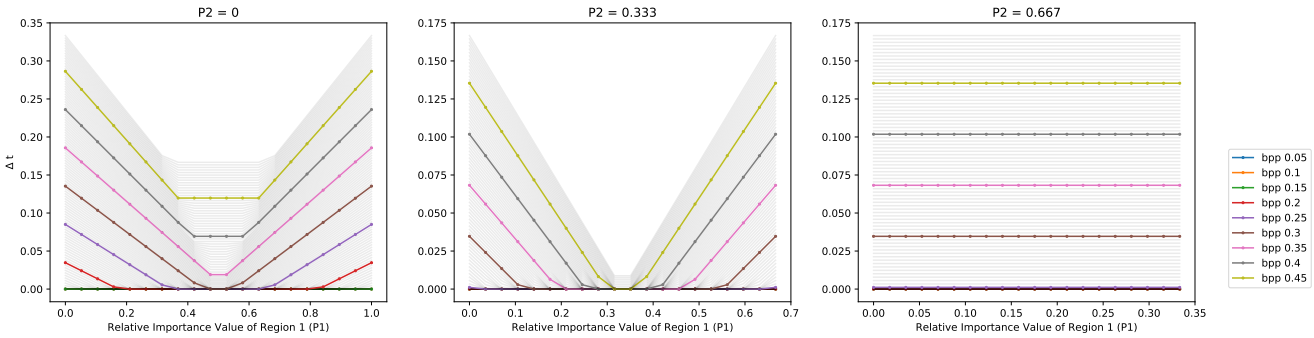


Figure 8. Plot of Δt vs p_1 for different values of input bitrate (bpp) t. In each of these plots, we keep the relative importance p_2 fixed, and vary p_1 in the range $[0, 1 - p_2]$

equation for Δt is given by,

$$\Delta t = \begin{cases} 0 & \text{if} \quad t \leq 0.25 \\ 2t/3 - 1/6 & \text{if} \quad t > 0.25 \end{cases} \tag{13}$$

because $\min(C/3n^2, tp_i)$ for $(i = 1, 3)$ is always equal to tp_i[3], and $\min(C/3n^2, tp_2)$ is equal to $1/6$ for $t > 0.25$.

7. Classes in Cityscapes Dataset

Cityscapes dataset comprises of images of street scenes captured in different cities. Each street scene has a variety of objects. There are in total 34 categories of objects which are recognized in the dataset. For the purpose of our experiments on Cityscapes, we divide the 34 classes of objects into 5 broad classes,

- **Human:** person, rider
- **Vehicle:** car, truck, bus, on rails, motorcycle, bicycle, caravan, trailer
- **Object:** pole, pole group, traffic sign, traffic light
- **Construction:** building, wall, fence, guard rail, bridge, tunnel
- **Others:** road, sidewalk, parking, rail track, vegetation, terrain, sky, ground, dynamic, static

8. Additional Results

8.1. Model Variations

We experiment with different values of C and n that can give us a maximum bitrate of 0.5. Figure 10 depicts the performance of those model variations on PSNR and MS-SSIM. We also experiment a variation all these models with positional encodings [18], which get concatenated to the input image along with the user input importance map. We use the Gaussian fourier feature mapping to map our pixel coordinates to a Fourier space. Let $\mathbf{v} = [x, y]$ and $\gamma(\mathbf{v})$ be its positional encoding, then

$$\gamma(\mathbf{v}) = [\cos(2\pi \mathbf{B}\mathbf{v}), \sin(2\pi \mathbf{B}\mathbf{v})]^{\mathrm{T}} \tag{14}$$

where each entry in $\mathbf{B} \in \mathbb{R}^{m \times d}$ is sampled from a Normal Distribution $\mathcal{N}(0, \sigma^2)$ with $\sigma = 0.5$. After a detailed study, we empirically conclude that the proposed model with $C = 8$ channels and a $n^2 = 16$ factor reduction, maintaining C/n^2 ratio to 0.5, gives the best performance on our dataset (see Figure 8). Hence, all the further experiments and results are reported on the same model.

8.2. Gains from User Guidance

We test our hypothesis of controlling output bitrate (t_{out}) via user-guided relative importance values by considering two alternatives that **do not** have any user guidance or segmentation masks: (a) *Identity mapping* which is equivalent to removing both L_E term (Equation 5) and importance map network and directly replacing the learned importance map with a constant map of value tn^2/C (Equation 12) where t is the input bitrate, (b) *No-ROI mapping* which is equivalent to setting $L_E = L_{whole}$ and replacing the input importance map with a constant map of value tn^2/C. Note that (a) and (b) serve as ablations for importance map network and ED Loss respectively.

While (a) is the best at tracking input bitrate t exactly, it performs poorly on all quantitative metrics as seen in Table 6 with FID values taking upto 98.00 at $t = 0.05$. Since the learned map is constant, bits are not allocated contextually resulting in poor performance especially at low bitrates.

Importance map network is incorporated in (b), with an additional loss L_{whole} on t_{out} to make it closer to t. From the graph in Figure 11, we see that (b) is not able to track t, and t_{out} hovers around 0.216 with a small increasing trend.

[3]Note that $C/n^2 = 0.5$

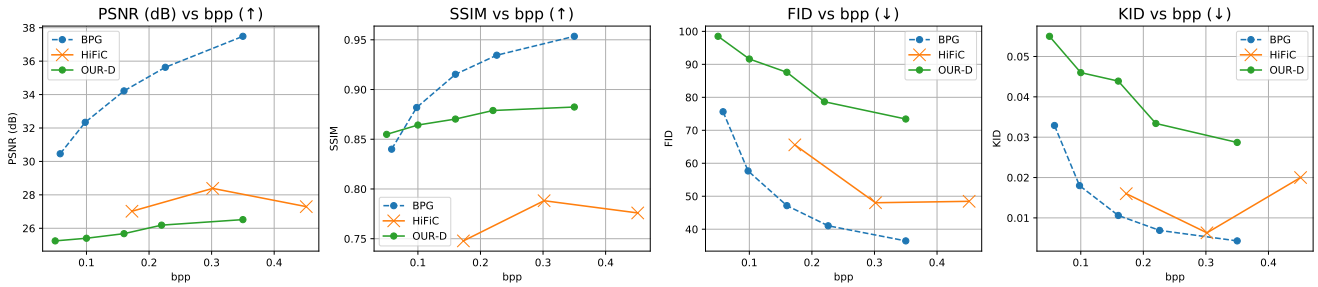


Figure 9. Comparison of our proposed method **Our-D** on Cityscapes dataset against baselines across (a) PSNR, (b) SSIM, (c) FID, and (d) KID metrics (SSIM and FID in the main paper also). Models are evaluated at different bitrates ranging from 0.05 to 0.5 (maximum bitrate possible).

User Input Bitrate (t)	PSNR (↑)			R_0 PSNR (↑)			FID (↓)		
	Identity (a)	No ROI (b)	With ROI (Our)	Identity (a)	No ROI (b)	With ROI (Our)	Identity (a)	No ROI (b)	With ROI (Our)
0.05	10.8134	25.6913	**25.8777**	12.2543	26.0890	**26.6052**	98.34 (± 4.821)	38.99 (± 0.440)	**18.23** (± 0.394)
0.15	23.3128	25.7425	**26.4593**	23.7337	26.1675	**27.1736**	78.73 (± 2.722)	40.21 (± 0.622)	**15.29** (± 0.343)
0.30	24.9450	25.7712	**27.5145**	25.1587	25.3770	**28.1576**	64.54 (± 2.082)	39.12 (± 0.483)	**13.85** (± 0.295)

Table 6. Quantitative comparison of alternatives (a) Identity mapping, (b) No-ROI mapping for controlling output bitrate without any user guidance. Values are computed on CelebA test set where R_0 corresponds to face-region. (↑) arrow indicates that higher values are better.

User Input Bitrate	R_1 PSNR ↑			R_2 PSNR ↑			SSIM ↑			KID ↓		
	Identity (a)	No ROI (b)	With ROI (Our)	Identity (a)	No ROI (b)	With ROI (Our)	Identity (a)	No ROI (b)	With ROI (Our)	Identity (a)	No ROI (b)	With ROI (Our)
0.05	12.1932	**25.3386**	24.1934	11.2482	26.8067	**27.3108**	0.4650	**0.8642**	0.8531	0.4029 (0.016)	0.0244 (0.006)	**0.0054 (0.009)**
0.15	23.2416	**25.3370**	24.9001	25.9152	26.8291	**27.5653**	0.8374	0.8643	**0.8840**	0.0280 (0.007)	0.0239 (0.005)	**0.0028 (0.006)**
0.30	24.6950	25.4280	**25.9472**	26.0419	26.9231	**28.6331**	0.8540	0.8650	**0.9045**	0.0347 (0.009)	0.0241 (0.006)	**0.0018 (0.006)**

Table 7. Quantitative comparison of alternatives (a) Identity mapping, (b) No-ROI mapping for controlling output bitrate without any user guidance. Values are computed on CelebA test set where R_1, R_2 correspond to hair and background region respectively. (↑) arrow indicates that higher values are better.

Since t is uniformly distributed (Section 5), $E(t) = 0.245$ and gradients from the averaged-L_{whole} over entire train set push the network towards giving a t_{out} close to $E(t)$ leading to values in the range $[0.20, 0.24]$. While the values in Table 6 look much better as compared to (a), note that (b) uses additional bits worth ~ 0.1 in [0.01 - 0.15] range. Despite taking up additional bits, the performance is worse compared to our method. For instance at $t = 0.05$, our method provides a $+20.12$ reduction on FID while taking lesser bits ($t_{out} = 0.07$). Again, due to the lack of region-specific contextual information, bits are allocated poorly in method (b) as seen in the learned map in Figure 12. Our method gives a $+1.6$ gain in R_0 PSNR at $t = 0.15$ in spite of allocating fewer bits to R_0 (face).

These findings verify the benefits of using our method which incorporates user-guided ROI via ED Loss, and is able to divert bits to useful locations within a region (like high texture) while staying within the limits of user-provided bit budget (t). From Figure 11, we see that our method is able to track t very well in the range of [0.07 - 0.42] but tapers at extremes due to clipping (discussed in Section 5). Quantitative results from Table 6 indicate the superior performance of our method at all bitrates owing to a much-improved bit allocation. From Figure 12, visual quality of reconstructed images is strikingly better than the other two cases with the learned map containing bits at regions of high variability. Thus, we have shown that user-guided relative importance values help in optimally leveraging the available bit budget (unlike (b)) by allocating bits to appropriate locations and maximising the reconstructed image quality (unlike (a)), confirming our hypothesis.

Table 7 shows additional results comparing alternatives that do not consider user guidance. It is evident that our method outperforms both the alternatives in all the metrics. De-

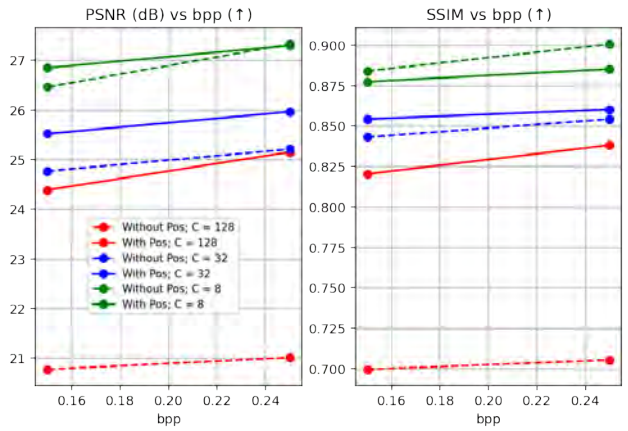


Figure 10. Different architecture explorations to achieve a Maximum bpp of $= 0.5$. We find that ($C = 8, n^2 = 16$) without position encodings is the best combination

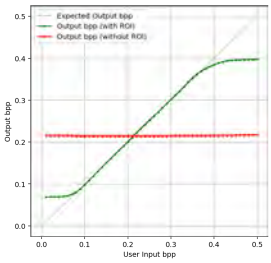


Figure 11. Variation of Output bitrate as t increases from 0.01 to 0.5. Without-ROI corresponds to *no-ROI mapping*; With-ROI corresponds to our method with user guidance.

Figure 12. Qualitative comparison of reconstructed images and their learned maps from alternatives at $t = 0.15$: (a) Identity mapping, (b) No-ROI mapping, (our) method is with user-guided map

spite taking much fewer bits for each region, the region-wise PSNR is higher in both hair and background regions. Identity mapping at low bitrates is very bad owing to the lack of importance map network; SSIM and KID values are significantly lower than other two methods. No-ROI mapping performs slightly better than our method in R_1 PSNR at $t = 0.05$ and $t = 0.15$ because it does not stay within the limits of bit budget. Output bitrate t_{out} tracks the input bitrate exactly for our method, whereas *No-ROI mapping* alternative fixates around $t_{out} = 0.216$. Thus, at $t = 0.05$, our method uses only $t_{out} =\sim 0.05$ to give R_1 PSNR of 24.1934 points, whereas No-ROI alternative uses $t_{out} = 0.216$ to give R_1 PSNR of 25.3386 (+1.1 gain). To iterate, this happens because *No-ROI mapping* does not stay within the desired bit budget limits.

8.3. Quantitative Comparison with Baselines on Cityscapes

Since the maximum attainable bitrate in our scenario is 0.5, we experiment with the following values of bitrates - $[0.05, 0.1, 0.15, 0.23, 0.35]$, while keeping the relative importance values constant and equal ($[0.2, 0.2, 0.2, 0.2, 0.2]$ for Cityscapes dataset) to allow for comparison against the baselines. From Figure 9, we see that our method **Our-D** might be performing inferior to the baselines on PSNR and KID. We intend to work on this in our future work. Despite of the low numbers on quantitative metrics, our model is preferred at par with the baselines by the users, as concluded from the human evaluation. Additionally, it is imperative to note that both our baselines - HiFiC [14] and BPG require either a new model to be trained for each of the target bitrates (low, medium and high regimes for HiFiC) or need manual control over the quality parameter (-q parameter in BPG). In both the cases, the user can't estimate the output bitrate right at the start of compression algorithm and has to do multiple iterations to achieve the desired bitrate whereas our model facilitates the user with an explicit control over the target bitrate and subsequently, adheres to the target set by the user during the compression. Furthermore, our model doesn't need to be retrained for different target bitrates - it is a single model that works for a range of bitrates, thereby reducing the training computation cost incurred to train a model for each new bitrate.

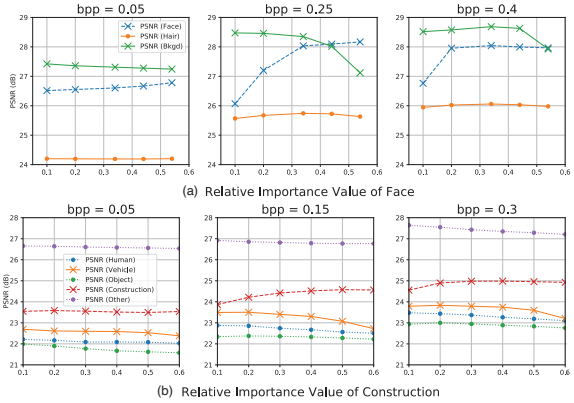


Figure 13. Comparison of PSNR of regions as we vary relative importance values for (a) faces (↑) and background (↓) in CelebA keeping it fixed for hair (b) construction (↑) and vehicle ↓) regions in CityScapes while keeping it fixed for the other classes. Note that PSNR decreases for all bitrates for background and vehicles due to decreasing relative importance and increases for faces and construction respectively

8.4. Varying User-Input Importance at Multiple Bitrates

Figure 13 shows the trend for PSNR values for different object classes when the user-input relative importance values are varied at different bitrates. Figure 14 shows some more qualitative examples. We present the learned importance maps and the reconstructed images by our model at different bitrates and across different user-specific importance values. The gradual variation of the increasing importance given to face as against the background (specified by the user input importance), is clearly evident in all the learned importance maps of the images (refer to the importance map rows at each bitrate). The resulting impact on the reconstruction can also be observed, for instance, facial detailing like wrinkles on forehead in (ii), dimples on cheeks in (iii) are better preserved with a higher importance to face (column (e)) at the same bitrate. Simultaneously, the background objects which are specified un-important by the user, are not reconstructed perfectly. This can be observed through the background in (iv) and the earrings of the women in (i), (iii). This effect is also measured quantitatively through the region-wise PSNR values of Face and Background. Note the trend of increasing PSNR of face and decreasing PSNR of the background in each row.

Similarly, we can see some more qualitative examples at different target bitrates and relative importance values for Cityscapes Dataset in figure 17. As the user input relative importance of Construction region is increased ((a) to (c) in each row), it gets allocated more bit budget which is reflected in the learned importance maps while the bit budget to vehicle region gets decreased as is visible in the examples on moving from left to right. Effects of this change are clearly reflected in the examples: the cars in front of white house in (i) and the one near umbrella in (ii) degrade sharply while the windows and other features in the buildings on right become more sharper as we go from (a) to (c); the red slant roof in (iii) is best reconstructed in (c) as compared to (a) and the red car gets smudgy in (iv) while the features in the buildings on right get sharper and better as we move from (a) to (c). The decreasing trend in the reported region wise PSNR values of vehicle and increasing trend for construction further validates our visual observations.

Observing the importance maps column-wise also shows the effect of bitrate on the importance map weights. As we allot more bits to the images, the weights given to the regions in the importance map also increases (visualized through darker blue color as we go down the column). For the reconstructions at 0.4 input bpp, we observe that the reconstructed images do not make use of the entire allotted bitrate budget. The generated images are of high visual quality, but at a lower bitrate than provided.

8.5. Human Evaluation

We pay our annotators at 10 dollars per hour. In the third survey, we examine how well our algorithm incorporates user importances by displaying three images with low/equal/high immportance to face/vehicle region at various bitrates. We ask the annotators to pick the image that has the best reconstruction of face/vehicle region. This enables us to obtain the preferential alignment of user importances with model reconstructions. We find that our model produces a superior output when the specified region is given higher importance, which also aligns with the perception of users (annotators). For all bitrates and dataset types, the specified region with "high" importance is perceived to have the best reconstruction. While Survey-2 shows that our method fares better at reconstructing specified region with high importance as compared to baseline, this survey shows the ability of our method to align with user preferences.

8.6. Qualitative Comparison with Baselines

Figure 15,16, 18 and 19 present the qualitative comparison of compressed images with our proposed model against the baselines BPG and HiFiC. In Figure, 15, we compare the reconstructions at extremely low bitrates. It is visually evident that our model produces compression of better quality even at such low bitrates of 0.06 and 0.095. Similarly, in figure 18 we carefully observe that BPG inherently does smooth out a lot of regions especially in the vegetation which sometimes hides the objects in its foreground (the circle in traffic sign pole almost gets lost in the smoothing of vegetation). However, due to region wise bit allocation we tend to maintain some texture in vegetation and hence perform at par with BPG in terms of SSIM. Note that HiFiC model is trained for 3 specific variations - low, med, high and hence, the lowest bitrate attained through this baseline is around 0.2. Hence, we couldn't compare against this model at low bitrates. Figure 16, 19 show the comparison with the results from HiFiC at 3 levels. We also report the closest attainable image through BPG. It is observed that BPG results are not very good visually at low bitrates. When compared with HiFiC, our model gives results on par with it for CelebA and close enough for Cityscapes (intend to improve in future work). Our model produces reconstructions with reasonably high SSIM as compared to our baselines. Also, the most important advantage is that our model is trained only once, while the baselines had to be trained multiple times to obtain reconstructions at varied bitrates.

Class Pref (→) Input BPP (↓)	Dataset	Low	Equal	High
Low (0.2)	CelebA	15	**15**	13
	Cityscapes	10	14	**22**
Medium (0.3)	CelebA	11	14	**18**
	Cityscapes	12	**19**	16
High (0.4)	CelebA	10	15	**20**
	Cityscapes	11	15	**20**

Table 8. Human Evaluation Survey Results. Values correspond to respondent counts for the query: "Which image reconstructs the face/vehicle region better?"

References

[1] Eirikur Agustsson, Fabian Mentzer, Michael Tschannen, Lukas Cavigelli, Radu Timofte, Luca Benini, and Luc Van Gool. Soft-to-hard vector quantization for end-to-end learning compressible representations, 2017. 1

[2] Eirikur Agustsson, Michael Tschannen, Fabian Mentzer, Radu Timofte, and Luc Van Gool. Generative adversarial networks for extreme learned image compression. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 221–231, 2019. 1

[3] Johannes Ballé, Valero Laparra, and Eero P Simoncelli. End-to-end optimized image compression. *arXiv preprint arXiv:1611.01704*, 2016. 1

[4] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. *arXiv preprint arXiv:1802.01436*, 2018. 1

[5] Chunlei Cai, Li Chen, Xiaoyun Zhang, and Zhiyong Gao. End-to-end optimized roi image compression. *Trans. Img. Proc.*, 29:3442–3457, 2020. 1

[6] Yoojin Choi, Mostafa El-Khamy, and Jungwon Lee. Variable rate deep image compression with a conditional autoencoder. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3146–3154, 2019. 1

[7] John D. Cook. Sum of uniform random variables. 2009. 4

[8] Emilien Dupont, Adam Goliński, Milad Alizadeh, Yee Whye Teh, and Arnaud Doucet. Coin: Compression with implicit neural representations, 2021. 1

[9] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. 1

[10] Nick Johnston, Damien Vincent, David Minnen, Michele Covell, Saurabh Singh, Troy Chinen, Sung Jin Hwang, Joel Shor, and George Toderici. 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*, pages 4385–4393, 2018. 1

[11] Mu Li, Wangmeng Zuo, Shuhang Gu, Debin Zhao, and David Zhang. Learning convolutional networks for content-weighted image compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3214–3223, 2018. 1

[12] Jae Hyun Lim and Jong Chul Ye. Geometric gan, 2017. 2

[13] Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, and Luc Van Gool. Conditional probability models for deep image compression. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4394–4402, 2018. 1

[14] Fabian Mentzer, George Toderici, Michael Tschannen, and Eirikur Agustsson. High-fidelity generative image compression, 2020. 1, 8

[15] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks, 2018. 2

[16] Yash Patel, Srikar Appalaraju, and R. Manmatha. Saliency driven perceptual image compression. *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 227–236, 2021. 1

[17] G. J. Sullivan, J. Ohm, W. Han, and T. Wiegand. Overview of the high efficiency video coding (hevc) standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1649–1668, 2012. 1

[18] Matthew Tancik, Pratul P Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *arXiv preprint arXiv:2006.10739*, 2020. 6

[19] David Taubman and Michael Marcellin. *JPEG2000 Image Compression Fundamentals, Standards and Practice*. Springer Publishing Company, Incorporated, 2013. 1

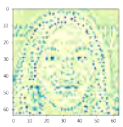
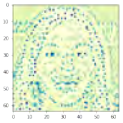
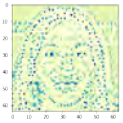
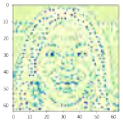
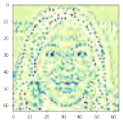
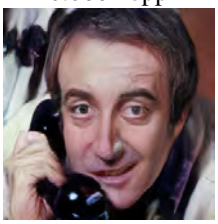
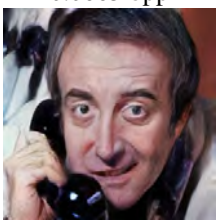
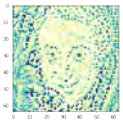
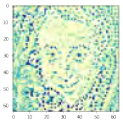
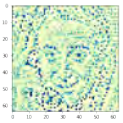
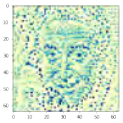
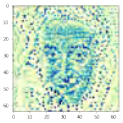
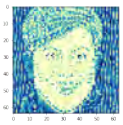
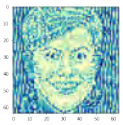
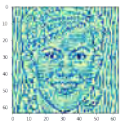
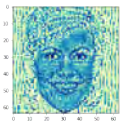
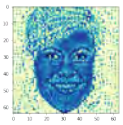
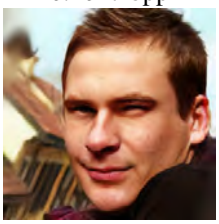
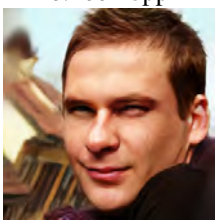
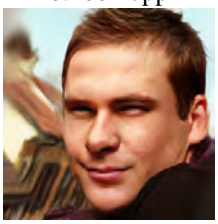
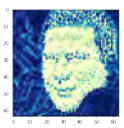
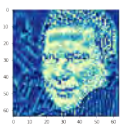
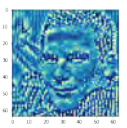
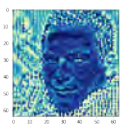
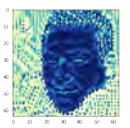
[20] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. *arXiv preprint arXiv:1703.00395*, 2017. 1

[21] George Toderici, Sean M O'Malley, Sung Jin Hwang, Damien Vincent, David Minnen, Shumeet Baluja, Michele Covell, and Rahul Sukthankar. Variable rate image compression with recurrent neural networks. *arXiv preprint arXiv:1511.06085*, 2015. 1

[22] George Toderici, Damien Vincent, Nick Johnston, Sung Jin Hwang, David Minnen, Joel Shor, and Michele Covell. Full resolution image compression with recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5306–5314, 2017. 1

[23] G. K. Wallace. The jpeg still picture compression standard. *IEEE Transactions on Consumer Electronics*, 38(1):xviii–xxxiv, 1992. 1

[24] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks, 2019. 2

Original Image	(a)	(b)	(c)	(d)	(e)
					
(i) I/p bpp = 0.05					
	P_{face} - 25.06 P_{bkg} - 27.35 0.0681 bpp	P_{face} - 25.19 P_{bkg} - 26.27 0.0661 bpp	P_{face} - 25.38 P_{bkg} - 26.34 0.0665 bpp	P_{face} - 25.48 P_{bkg} - 25.76 0.0691 bpp	P_{face} - 25.88 P_{bkg} - 26.32 0.0734 bpp
					
(ii) I/p bpp = 0.10					
	P_{face} - 23.60 P_{bkg} - 25.18 0.1096 bpp	P_{face} - 24.57 P_{bkg} - 25.24 0.1037 bpp	P_{face} - 25.35 P_{bkg} - 24.36 0.1043 bpp	P_{face} - 25.60 P_{bkg} - 24.23 0.1073 bpp	P_{face} - 26.15 P_{bkg} - 24.06 0.1177 bpp
					
(iii) I/p bpp = 0.20					
	P_{face} - 26.99 P_{bkg} - 28.06 0.2008 bpp	P_{face} - 28.14 P_{bkg} - 28.17 0.2003 bpp	P_{face} - 28.52 P_{bkg} - 28.15 0.2017 bpp	P_{face} - 29.18 P_{bkg} - 27.42 0.2001 bpp	P_{face} - 29.30 P_{bkg} - 26.72 0.2001 bpp
					
(iv) I/p bpp = 0.30					
	P_{face} - 27.18 P_{bkg} - 27.55 0.283 bpp	P_{face} - 28.25 P_{bkg} - 27.87 0.2953 bpp	P_{face} - 29.02 P_{bkg} - 27.91 0.301 bpp	P_{face} - 29.05 P_{bkg} - 27.70 0.3025 bpp	P_{face} - 29.08 P_{bkg} - 25.57 0.2749 bpp

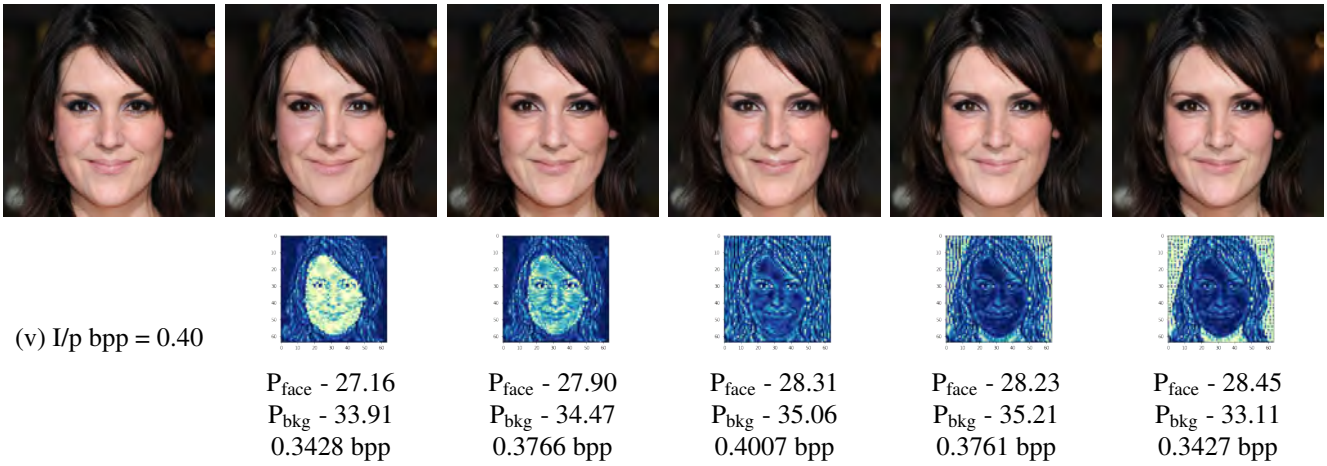


Figure 14. Reconstructions and Learned Importance maps obtained by our model at different bitrates (ranging from 0.05 to 0.4) and user-input importance maps. We present the user-input importance map variations of face, hair, background (p_1, p_2, p_3) at (a) [0.1, 0.33, 0.57] (b) [0.2, 0.33, 0.47] (c) [0.34, 0.33, 0.33] (d) [0.44, 0.33, 0.23] (e) [0.54, 0.33, 0.13]

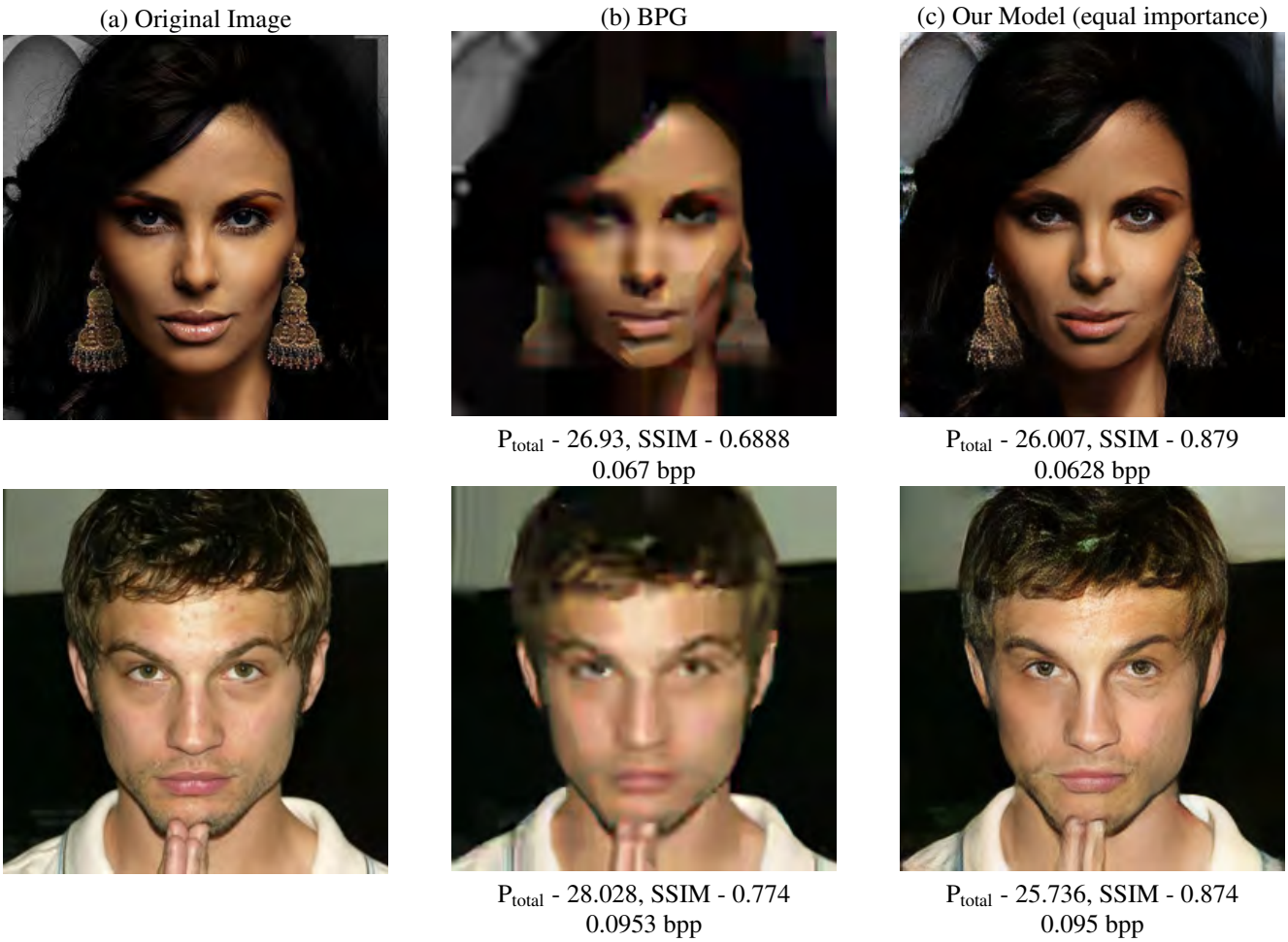


Figure 15. Qualitative comparison with BPG at extremely low bitrates. The PSNR and SSIM of the compressed images and the bitrates are reported.

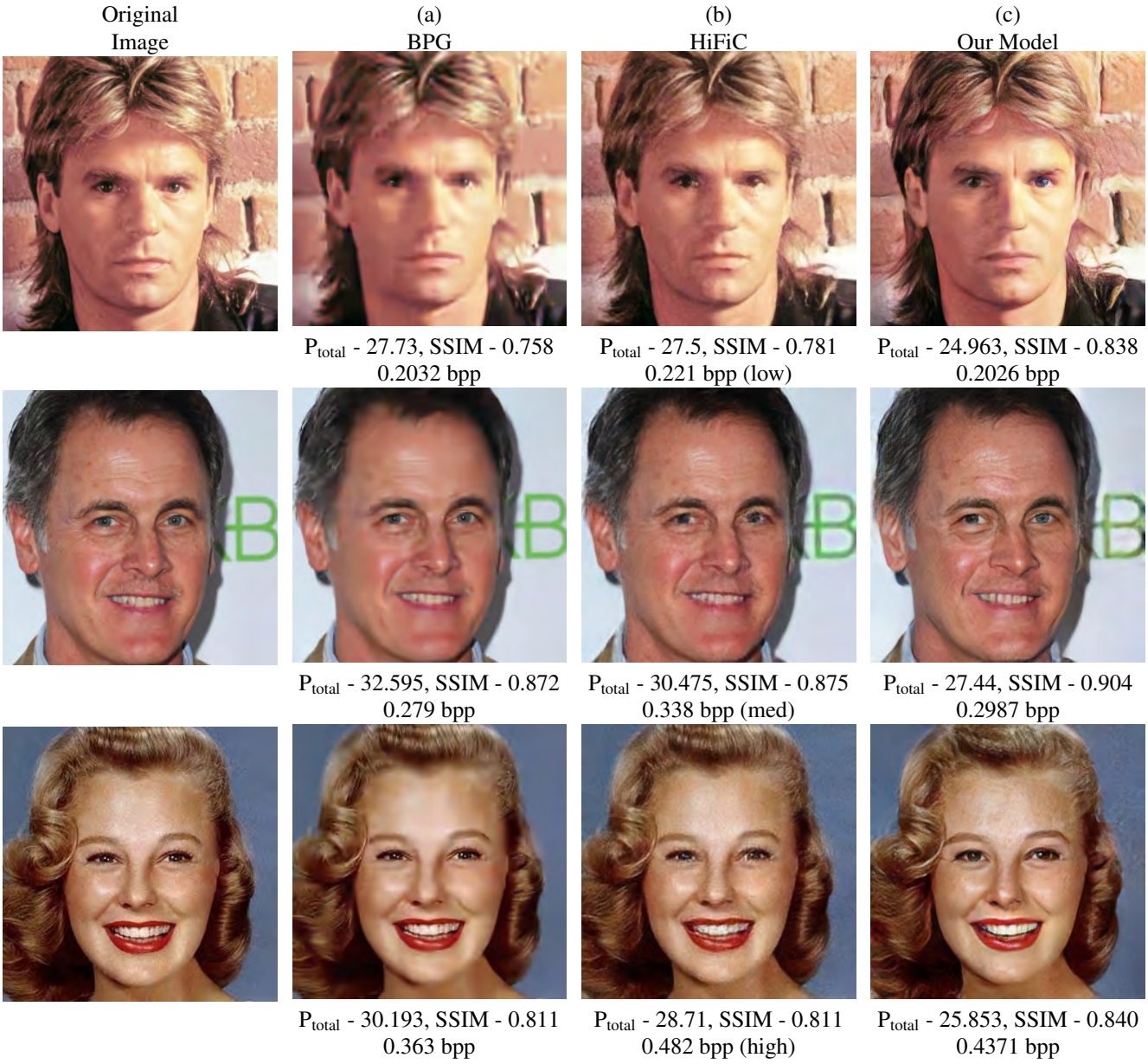
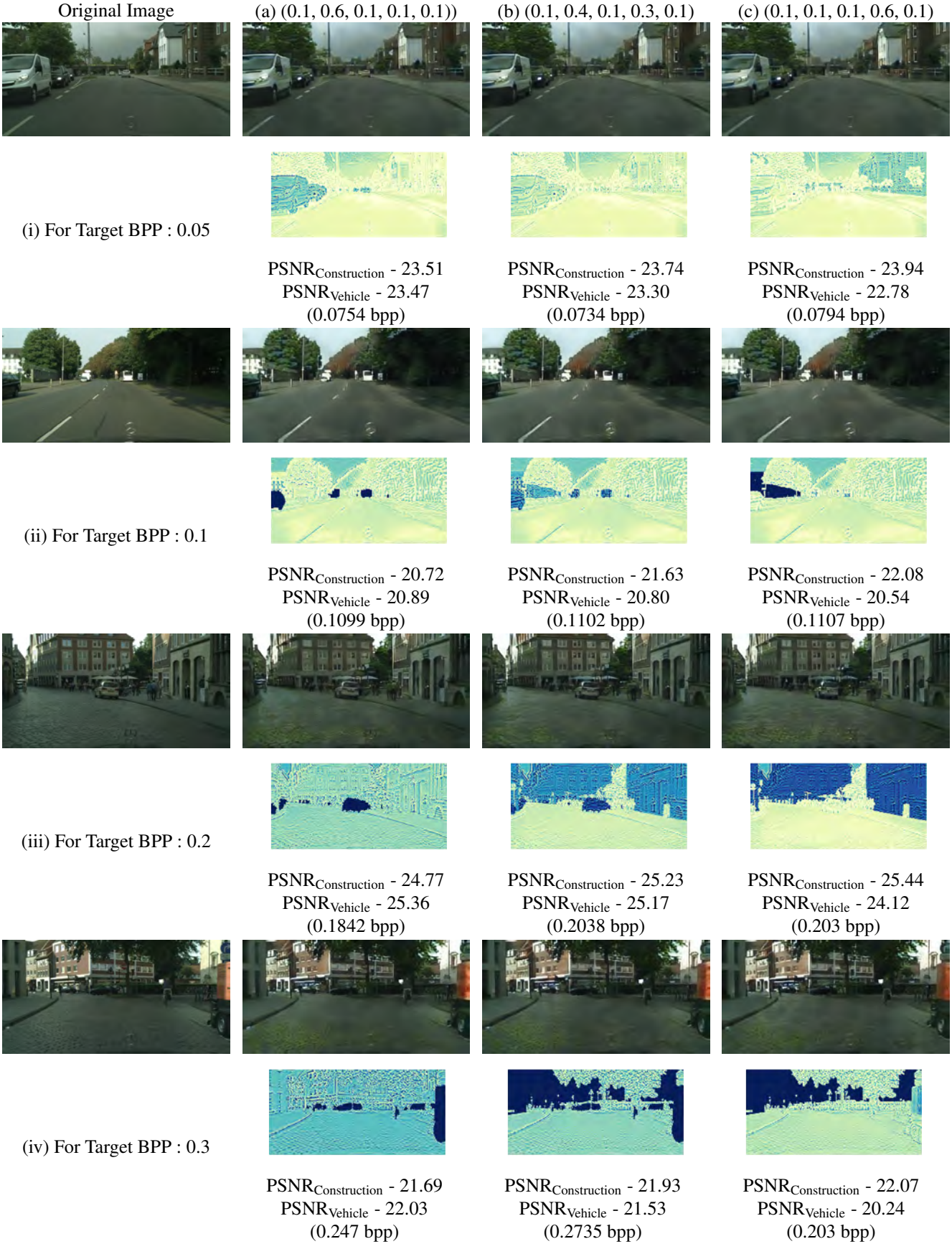
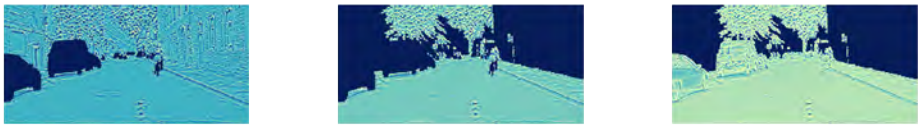


Figure 16. Qualitative comparison of baselines against our model(c). We compare against (a)BPG and (b)HiFiC at different bitrates. The PSNR and SSIM are reported for each of the compressed reconstruction.


Original Image
(a) (0.1, 0.6, 0.1, 0.1, 0.1))
(b) (0.1, 0.4, 0.1, 0.3, 0.1)
(c) (0.1, 0.1, 0.1, 0.6, 0.1)
(i) For Target BPP : 0.05
PSNR$_{Construction}$ - 23.51
PSNR$_{Vehicle}$ - 23.47
(0.0754 bpp)
PSNR$_{Construction}$ - 23.74
PSNR$_{Vehicle}$ - 23.30
(0.0734 bpp)
PSNR$_{Construction}$ - 23.94
PSNR$_{Vehicle}$ - 22.78
(0.0794 bpp)
(ii) For Target BPP : 0.1
PSNR$_{Construction}$ - 20.72
PSNR$_{Vehicle}$ - 20.89
(0.1099 bpp)
PSNR$_{Construction}$ - 21.63
PSNR$_{Vehicle}$ - 20.80
(0.1102 bpp)
PSNR$_{Construction}$ - 22.08
PSNR$_{Vehicle}$ - 20.54
(0.1107 bpp)
(iii) For Target BPP : 0.2
PSNR$_{Construction}$ - 24.77
PSNR$_{Vehicle}$ - 25.36
(0.1842 bpp)
PSNR$_{Construction}$ - 25.23
PSNR$_{Vehicle}$ - 25.17
(0.2038 bpp)
PSNR$_{Construction}$ - 25.44
PSNR$_{Vehicle}$ - 24.12
(0.203 bpp)
(iv) For Target BPP : 0.3
PSNR$_{Construction}$ - 21.69
PSNR$_{Vehicle}$ - 22.03
(0.247 bpp)
PSNR$_{Construction}$ - 21.93
PSNR$_{Vehicle}$ - 21.53
(0.2735 bpp)
PSNR$_{Construction}$ - 22.07
PSNR$_{Vehicle}$ - 20.24
(0.203 bpp)

(v) For Target BPP : 0.4



$PSNR_{Construction}$ - 26.57
$PSNR_{Vehicle}$ - 26.88
(0.2915 bpp)

$PSNR_{Construction}$ - 26.75
$PSNR_{Vehicle}$ - 26.59
(0.3291 bpp)

$PSNR_{Construction}$ - 26.96
$PSNR_{Vehicle}$ - 26.13
(0.2652 bpp)

Figure 17. Reconstructions and Learned Importance maps obtained by our model at different bitrates (ranging from 0.05 to 0.4, 0.15 shown in main text) and user-input relative importance values $(p_1, p_2, p_3, p_4, p_5)$ for Construction (p_4) and Vehicle (p_2) class keeping other regions constant

(a) Original Image

(b) BPG

P_{total} - 30.38, SSIM - 0.8452
0.0559 bpp

(c) Our Model (equal importance)

P_{total} - 25.49, SSIM - 0.8609
0.0599 bpp

Figure 18. Qualitative comparison with BPG at an extremely low bitrate of 0.05 bpp. The PSNR and SSIM of the compressed images and the corresponding bitrates are reported.

Original Image

(a) BPG

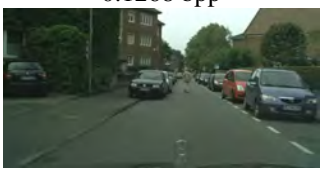
P_{total} - 34.98, SSIM - 0.9188
0.1268 bpp

(b) HiFiC

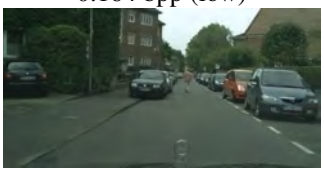
P_{total} - 27.70, SSIM - 0.7189
0.164 bpp (low)

(c) Our Model

P_{total} - 27.91, SSIM - 0.9012
0.1463 bpp



P_{total} - 38.14, SSIM - 0.9475
0.280 bpp



P_{total} - 27.41, SSIM - 0.699
0.288 bpp (med)

P_{total} - 28.66, SSIM - 0.9025
0.2297 bpp

P_{total} - 38.37, SSIM - 0.9560
0.3449 bpp

P_{total} - 28.30, SSIM - 0.7727
0.404 bpp (high)

P_{total} - 28.43, SSIM - 0.911
0.3496 bpp

Figure 19. Qualitative comparison of baselines against our model(c). We compare against (a)BPG and (b)HiFiC at different bitrates. The PSNR and SSIM are reported for each of the compressed reconstruction.