Saliency Driven Perceptual Image Compression

Yash Patel[2,*], Srikar Appalaraju[1], R. Manmatha[1]
[1]Amazon Web Services, Palo Alto
[2]Visual Recognition Group, Czech Technical University in Prague
patelyas@fel.cvut.cz, (srikara,manmatha)@amazon.com

Abstract

This paper proposes a new end-to-end trainable model for lossy image compression, which includes several novel components. The method incorporates 1) an adequate perceptual similarity metric; 2) saliency in the images; 3) a hierarchical auto-regressive model. This paper demonstrates that the popularly used evaluations metrics such as MS-SSIM and PSNR are inadequate for judging the performance of image compression techniques as they do not align with the human perception of similarity. Alternatively, a new metric is proposed, which is learned on perceptual similarity data specific to image compression. The proposed compression model incorporates the salient regions and optimizes on the proposed perceptual similarity metric. The model not only generates images which are visually better but also gives superior performance for subsequent computer vision tasks such as object detection and segmentation when compared to existing engineered or learned compression techniques.

1. Introduction

In the last two decades, the number of images captured and transmitted via the internet has grown exponentially [11]. This surge has increased both data storage and transmission requirements. Compression of images can be lossless [8, 37], that is, the original image can be perfectly reconstructed. A better compression rate can be obtained by using lossy methods such as JPEG [41], JPEG-2000 [37] and BPG [5]. The objective of these lossy methods is to obtain higher compression by removing the information which is least noticeable to humans. Note that these traditional codecs are hand-crafted and are not learned from the data.

More recent methods focus on learning to compress images. While learned image compression from data using neural networks is not new [28, 17, 24], there has recently been a resurgence of deep learning-based techniques for solving this problem [3, 4, 26, 32, 22, 29, 39]. These models are trained by jointly minimizing rate (storage or transmission requirement) and distortion (reconstruction quality), leading to a Rate-Distortion trade-off [35].

The contributions of this paper are three-fold:

1. A novel hierarchical auto-regressive architecture for image compression is proposed, which is based on an encoder-decoder setup.
2. A learned perceptual similarity metric is proposed. The metric is realized by a fully convolutional network (FCN) which is trained on compression specific artefacts. Note that the metric is differentiable and it is used as a for training the image compression model.
3. The proposed model accounts for salient regions. This is achieved in two ways: (a) more bits are allocated to the salient regions and (b) higher weight is given to their reconstruction.

The motivations and background for these contributions are subsequently discussed, along with the related work.

Taxonomy of image compression models. Deep learning models for image compression can be broadly categorized into three generative models: a) Variational Auto-Encoders [19, 4, 3, 22]; b) Generative Adversarial Networks [32, 2] (GAN) [13]; c) Auto-Regressive (AR) [40, 26]. Variational auto-encoders (VAEs) and auto-regressive models operate by estimating the probability density explicitly. On the other hand, GANs have an implicit measure of the density [13]. GANs are useful in very low bit-rate settings as they can learn to synthesize the images [2]. However, their superiority over AR and VAE is unclear for higher bit-rates. A key difference between VAE and AR is that the former approximates the density, whereas auto-regressive models such as Pixel-CNNs, Pixel-RNNs [40] have an explicit and tractable measure of density either in the pixel space or in the learned quantized space.

Proposed hierarchical auto-regressive model. A hierarchical auto-regressive model with two-stages is designed,

[*]The reserch was conduceted during Y. Patel's internship at AWS.

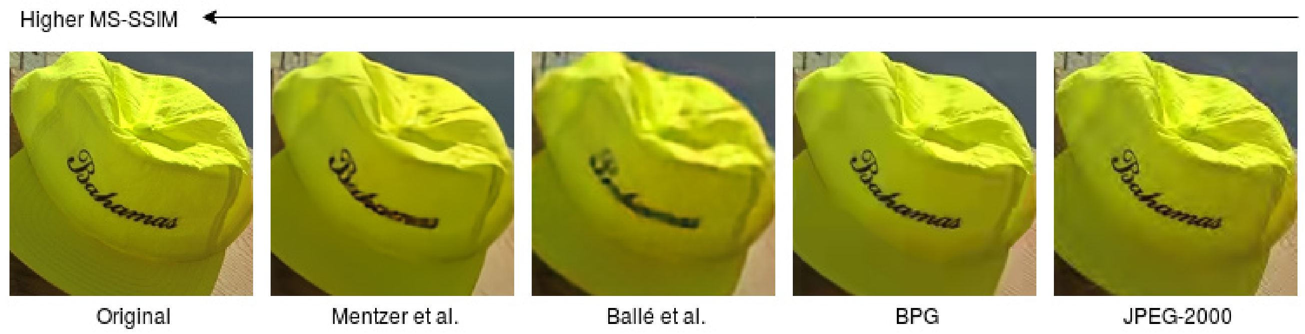


Figure 1: An example from the Kodak dataset [12]. In order of MS-SSIM values: Mentzer *et al.* [26] $>$ Ballé *et al.* [3] $>$ BPG [5] $>$ JPEG-2000 [37]. However, the order of performance based on 5 human evaluations is: BPG [5] $>$ Mentzer *et al.* [26] $>$ JPEG-2000 [37] $>$ Ballé *et al.* [3]. Visually the foreground and text in BPG is clearly better in quality.

which is realized by two 3D Pixel-CNN [40]. The Pixel-CNNs operate on the learned quantized representations. During training, the 3D Pixel-CNNs give an explicit measure of the entropy of the quantized representations, which via information theory directly relates to the bits required to store them after arithmetic coding [25]. Minimizing the estimated entropy leads to compression, while the reconstructed image should be as close to the original as possible. Note that auto-regressive models have been used before for compression [26], but, this is the first hierarchical auto-regressive model designed for the task.

Existing evaluation metrics. Lossy image compression models are compared by plotting the rate-distortion curve, where the rate is in bits-per-pixel vs the value of the distortion function. Common choices of the distortion function for compression models are MS-SSIM and PSNR [3, 4, 26, 22]. Both of these metrics do not align well with human perception of similarity [30]. As deep learned models directly optimize on these evaluation metrics, it is natural for them to have high MS-SSIM or PSNR scores when compared to the engineered methods such as JPEG-2000 [37], BPG [5]. However, the resulting images often look worse to a human, *i.e.*, the distortion in the images is higher although the metrics report otherwise. Figure 1 shows four different techniques ranked in descending order of MS-SSIM values. The 2nd and 3rd images have many more artefacts than the last two images which imply that MS-SSIM is arguably not a good evaluation measure for compression. In Figure 1 notice that the text is not as clear in the first two approaches.

The limitations of MS-SSIM and PSNR have been investigated in the past [43]. The super-resolution methods have started using a more sophisticated learned perceptual similarity metric [7, 6] for evaluation, exactly for the mentioned reasons. Despite of the short comings, recent image compression literature continues to evaluate models using MS-SSIM and PSNR [3, 4, 26, 22, 29, 32, 38, 39]. Further, it has been observed that techniques trained on PSNR perform well when evaluated on PSNR, but poorly when evaluated using MS-SSIM (and vice-versa). This makes developing practical compression systems difficult.

Proposed evaluation metric. This paper proposes a learned perceptual similarity metric for evaluating and training image compression models. This paper also studies the compression techniques using human evaluations. The study shows that human evaluation results correlate well with the proposed evaluation metric. Furthermore, the ranking of different compression methods, as judged by humans, correlates with the performance of off-the-shelf object detection and segmentation methods, which are trained on uncompressed images.

The analysis in this paper starts by evaluating the model proposed by Zhang *et al.* [43]. The model is trained on images with several different artefacts but the only compression artefacts are from JPEG. As Patel *et al.* [29] found, using the model [43] has limitations for compression. Therefore, we create a compression specific perceptual similarity dataset. The data include images generated from popular engineered [37, 41, 5] and learned compression methods [26, 3, 4, 29]. The data consists of 6 two alternative forced choices (2AFC) per comparison (see Section 2 for details).

Saliency matters for image compression. While JPEG [41] divides an image into uniform 8×8 blocks, BPG uses a hand-crafted metric to determine homogeneity and divides the more homogeneous regions into larger 64×64 blocks. In BPG, fewer bits are allocated to homogeneous regions and more bits are allocated to non-homogeneous regions. BPG builds on the hypothesis that humans are more prone to notice artefacts in complex regions of the image. Following this hypothesis of BPG, the proposed method makes use of object saliency in two ways: 1) **rate optimization**: more bits are allocated to the salient regions; 2) **distortion optimization**: artefacts in salient regions are more heavily penalized. To the best of our knowledge, we are the first to incorporate saliency in learned compression.

The rest of the paper is structured as follows. In Section 2, the compression specific perceptual similarity dataset is presented and various metrics are compared against the human judgements. Section 3 describes the proposed approach which is evaluated in Section 4. Section 5 concludes the paper.

2. Perceptual Similarity Metrics

This section investigates various perceptual similarity metrics for both engineered and learned compression methods. We collect a compression specific perceptual similarity dataset and benchmark the existing hand-crafted evaluation metrics PSNR and MS-SSIM, along with a learned metric LPIPS [43]. Following the recent evaluation setup in super-resolution literature [7, 6], we investigate linear combinations of learned and hand-crafted perceptual similarity metrics.

2.1. Setup for Human Evaluations

The setup for collecting this dataset aligns with that of [43] and adapts two alternatives forced choices (2AFC). Annotators are presented with two reconstructed versions of the same image from different compression methods, along with the original image in the middle. They are asked to pick the image which is closer to the original. At high bit rates, the images may be very similar, thus the annotators are provided with a synchronous magnifying glass. They are instructed to scan the images as a whole in cases of uncertainty. The evaluations were hosted on Amazon Mechanical Turk, on average, the annotators spent 56 seconds on one sample.

The images are obtained using the following image compression methods: Mentzer *et al.* [26], Patel *et al.* [29], BPG [5] and JPEG-2000 [37]. A total of 200 original images are used, comparisons are made at 4 different bit-rates and all possible combinations of methods are considered, *i.e.*, 6 combinations for 4 methods. This results in $4,800$ total samples for perceptual similarity studies. We use $3,840$ samples for training and 960 held out samples for testing. For each such sample, we obtain 6 evaluations resulting in a total of $28,800$ annotations.

2.2. Deep Perceptual Metric

The utility of using deep networks as a deep perceptual similarity metric has been studied by Zhang et al. [43]. It was observed that comparing activations from deep CNNS such as VGG-16 [36] or AlexNet [21] acts as a better perceptual similarity metric when compared to MS-SSIM and PSNR. We follow Zhang's approach and make use of activations from five $ReLU$ layers after each $conv$ block in the VGG-16 [36] architecture, with batch normalization.

Feed-forward is performed on VGG-16 for both the original $(\mathbf{x})$ and the reconstructed image $(\hat{\mathbf{x}})$. Let L be the set of layers used for loss calculation (five for our setup), a function $F(\mathbf{x})$ denoting feed-forward on an input image $\mathbf{x}$. $F(\mathbf{x})$ and $F(\hat{\mathbf{x}})$ return two stacks of feature activation's for all L layers. The Deep perceptual loss is then computed as:

- $F(\mathbf{x})$ and $F(\hat{\mathbf{x}})$ are unit-normalized in the channel dimension. Let us call these, $\mathbf{z^l_x}, \mathbf{z^l_{\hat{x}}} \in \mathbb{R}^{H_l \times W_l \times C_l}$ where $l \in L$. (H_l, W_l are the spatial dimensions).
- $\mathbf{z^l_x}, \mathbf{z^l_{\hat{x}}}$ are scaled channel wise by multiplying with the vector $\mathbf{w^l} \in \mathbb{R}^{C_l}$.
- The L_2 distance is then computed and an average over spatial dimension is taken. Finally, a channel-wise sum is taken which outputs the deep perceptual loss.

Equation 1 and Figure 2 summarize the Deep perceptual loss computation. Note that the weights in F are learned for image classification on the ImageNet dataset [34] and are kept fixed. $\mathbf{w}$ are the linear weights learned on top of F on the perceptual similarity dataset using a ranking loss function [43]. In the next subsection, the *LPIPS* metric is referred to learning $\mathbf{w}$ on Berkeley-Adobe Perceptual Patch Similarity Dataset [43] and *LPIPS-Comp (Ours)* is referred to the setup when $\mathbf{w}$ is learned on the compression specific similarity data (Section 2.1). Note that *LPIPS-Comp* is used as the Deep perceptual loss (DPL).

$$DPL(\mathbf{x}, \hat{\mathbf{x}}) = \sum_l \frac{1}{H_l W_l} \sum_{h,w} ||\mathbf{w_l} \odot (\mathbf{z^l_{\hat{x},h,w}} - \mathbf{z^l_{x,h,w}})||^2_2 \quad (1)$$

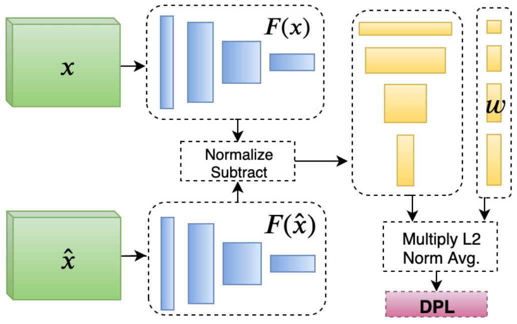


Figure 2: Deep Perceptual Loss: To compute perceptual similarity distance between the original x and the reconstructed $\hat{x}$ images - first compute the deep embeddings $F(x)$ and $F(\hat{x})$, normalize along the channel dimensions, scale each channel vector w (learned on perceptual similarity dataset) and take the ℓ_2 norm. Finally average across spatial dimensions and sum across channels.

2.3. Analysing Metrics for Image Compression

A comparison of various metrics is provided in Figure 3. The analysis starts by computing the 2AFC score of a human annotator. Then, the performance of two hand-crafted

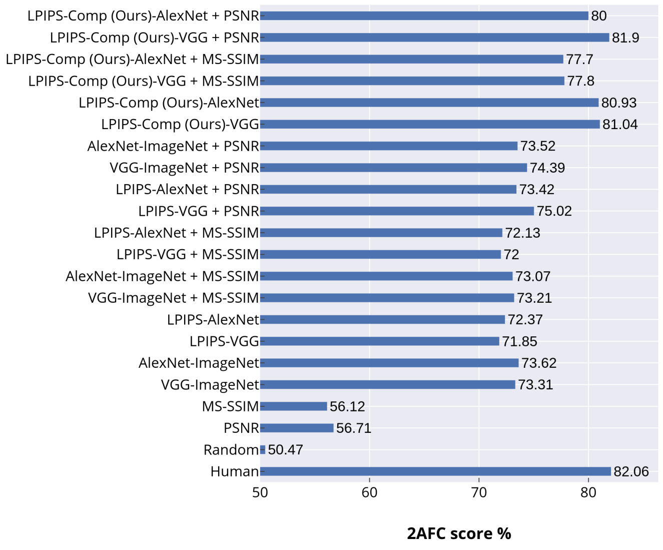


Figure 3: Comparison of various hand-crafted and learnt perceptual similarity metrics on the test set.

metrics PSNR and MS-SSIM are reported. Note that these are the most popular metrics used in the compression literature to report the state-of-the-art [26, 22, 4, 3, 32, 27]. However, as shown in the figure, the 2AFC scores for them (MS-SSIM and PSNR) are fairly low and do not align well with the human perception of similarity.

A naive similarity metric can be obtained by using AlexNet [21] or VGG-16 [36] trained on ImageNet. These features act as a better similarity metric [43] compared to PSNR or MS-SSIM. The features can be adapted better for perceptual similarity by following the framework presented in Section 2.2, *i.e.*, linearly re-weighting the channels with the weight vector on a perceptual similarity dataset. When these weights are learned on a generic dataset with a large collection of distortions (*LPIPS* in Figure 3) such as the Berkely-Adobe dataset [43], the performance is slightly worse compared to directly using the ImageNet model. This indicates a domain gap and establishes that using the similarity data of a different nature can have adverse effects. When the weights are trained on compression specific data (*LPIPS-Comp (Ours)* in 3), the learned metric aligns much better with the human judgements as can be clearly seen in Figure 3.

Finally, the linear combination of hand-crafted metrics PSNR and MS-SSIM with a learned metric is investigated. Unlike [7, 6], the weights given to each metric are learned on the compression specific similarity dataset. This is achieved by solving a linear optimization problem and employing RANSAC. We refer to the supplementary material for a detailed explanation of learning these weights. It is observed that *LPIPS-Comp (Ours)*, when combined with PSNR almost achieves close to human 2AFC score.

3. Proposed Method

The objective function of any lossy image compression technique is defined by a rate-distortion trade-off:

$$\min \sum_{\mathbf{x}\sim\chi} (\alpha Rate(\mathbf{x}) + \beta Distortion(\mathbf{x}, \hat{\mathbf{x}})) \tag{2}$$

where $\mathbf{x}$ is the image to be compressed drawn from a collection of images χ, $\hat{\mathbf{x}}$ is the reconstructed image, $Rate(\mathbf{x})$ is the storage requirement for the image and $Distortion(\mathbf{x}, \hat{\mathbf{x}})$ is a measure of distortion between the original and the reconstructed images.

As shown in Figure 4a, our method consists of two encoders, two decoders, two quantization stages and two auto-regressive models for entropy estimation. All models are trained jointly and in an end-to-end fashion. The first encoder takes the image as input and outputs latent representation $\mathbf{y} = E_1(\mathbf{x}) : \mathbb{R}^{W\times H\times 3} \rightarrow \mathbb{R}^{\frac{W}{8}\times\frac{H}{8}\times C_1+1}$ (Section 3.1). Note that the number of channels in the bottleneck, C_1, is one of the hyper-parameters to train the models to obtain different bits-per-pixel values. A pre-trained network outputs the object saliency for the input image $\mathbf{s} = S(\mathbf{x}) : \mathbb{R}^{W\times H\times 3} \rightarrow \mathbb{Z}_2^{\frac{W}{8}\times\frac{H}{8}\times 1}$ (Section 3.4). The latent representations are first masked by saliency driven priors $\mathbf{y_m} = m_1(\mathbf{y}, \mathbf{s})$ (Section 3.4) and then quantized $\tilde{\mathbf{y}} = Q_1(\mathbf{y_m}) : \mathbb{R} \rightarrow \{c_{(1,1)}, ..., c_{(1,L)}\}$ (Section 3.2) and fed to stage-two. Within stage two, the second encoder outputs the latent representations $\mathbf{z} = E_2(\tilde{\mathbf{y}}) : \mathbb{R}^{\frac{W}{8}\times\frac{H}{8}\times C_1} \rightarrow \mathbb{R}^{\frac{W}{32}\times\frac{H}{32}\times C_2+1}$ which are also masked $\mathbf{z_m} = m_2(\mathbf{z})$ (independent of saliency) and quantized with different centers $\tilde{\mathbf{z}} = Q_2(\mathbf{z_m}) : \mathbb{R} \rightarrow \{c_{(2,1)}, ..., c_{(2,L)}\}$.

An auto-regressive image compression model operating on a quantized latent representation [26] factorizes the discrete representation using a basic chain rule [20]:

$$P(\tilde{\mathbf{y}}) = \prod_{i=1}^{N} p(\tilde{y}_i | \tilde{y}_{i-1}, ..., \tilde{y}_1) \tag{3}$$

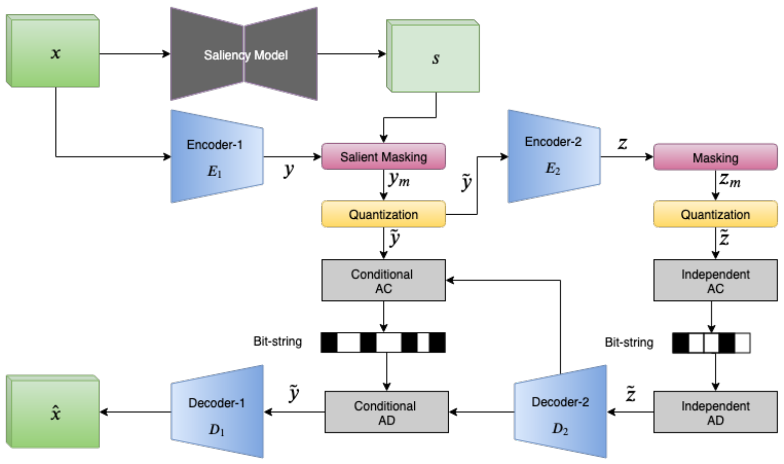
Our idea is to jointly learn an extra set of auxiliary representations $\tilde{\mathbf{z}}$ to factorize joint distribution using:

$$P(\tilde{\mathbf{y}}, \tilde{\mathbf{z}}) = P_\Theta(\tilde{\mathbf{y}}|\tilde{\mathbf{z}}) P_\Phi(\tilde{\mathbf{z}}) \tag{4}$$

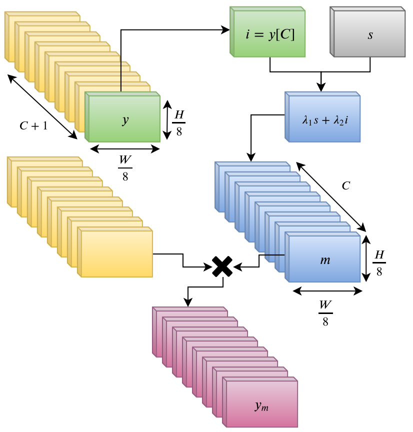
Here Θ and Φ are the parameters of two 3D Pixel-CNN models where $P(\tilde{\mathbf{z}})$ is the second stage which is decoded first during the decompression stage. Thus for the first stage, quantized representations $\tilde{\mathbf{y}}$ are encoded and decoded by assuming that the $\tilde{\mathbf{z}}$ is available. In the subsequent sections, each component of the proposed method is described.

3.1. Encoder-Decoder

The method consists of two encoders and two decoders. The first encoder is a fifteen residual blocks [15] fully-convolutional network with three non-local/self-attention



(a) **Overall Method**: In the first stage, the input image is fed to the first encoder and the saliency model, the features are masked using an importance mask and the saliency mask. The masked features are then quantized and fed to the second-stage. Within the second stage the features are fed through another encoder, quantized and compressed independently in a lossless manner using adaptive arithmetic coding. A transformed version of these compressed representations is used to condition the compression (entropy estimation) of the first stage's representation (standard adaptive arithmetic coding is used). Finally the compressed quantized representation are fed to the decoder to generate the reconstructed image.



(b) **Salient Masking**: The last channel of the bottleneck $E_1(\mathbf{x})[C_1]$ from the first stage is used as an importance map $\mathbf{i}$. This importance map is linearly combined with the saliency mask $\mathbf{s}$ and is expanded to match the dimensions of the bottleneck. Finally the bottleneck is masked using point-wise multiplication.

Figure 4: 4a provides an overview of our method while 4b illustrates the proposed saliency driven masking.

layers [42]. The first encoder involves down-sampling of the input image $\mathbf{x}$ by a factor of 8. The second encoder takes the quantized representations from the first stage $\tilde{\mathbf{y}}$ as input, feed-forwards through five residual blocks [15], a non-local layer [42] and involves a further down-sampling by a factor of 4. $\tilde{\mathbf{z}}$ is $\frac{W}{32} \times \frac{H}{32}$ and fairly small compared to the input $\mathbf{x}$ of $W \times H$. Thus, the number of bits required to store the second stage bit-string is very low (roughly 5% of the total storage). The decoders corresponding to these two encoders are their mirror.

The number of channels in the bottlenecks $\tilde{\mathbf{y}}$ or $\tilde{\mathbf{z}}$ is a hyper-parameter used to control the bits-per-pixel. In practice, the number of channels is kept the same, for both of these bottlenecks, *i.e.*, $C_1 = C_2$. Both bottlenecks have an extra channel for a hybrid of saliency mask and an importance map (Section 3.4).

3.2. Quantization

Quantization is a non-differentiable function, with gradients being either zero or infinite, thus any deep learning model with quantization cannot be trained using backpropagation [33]. Thus, soft vector quantization [1] is adapted in our model. More specifically, given a set of cluster centers $C_1 = \{c_1, ..., c_{L_1}\}$ the feed-forward is determined by:

$$\tilde{y}_i = Q_{C_1}(y) = argmin_j ||y_i - c_j|| \tag{5}$$

during backpropagation, a soft cluster assignment is used:

$$\hat{y}_i = \sum_{j=1}^{L_1} \frac{exp(-\sigma||y_i - c_j||)}{\sum_{l=1}^{l=L_1} exp(-\sigma||y_i - c_j||)} \tag{6}$$

Note that the quantization process is the same for both stages, but with different sets of centres.

3.3. Hierarchical Auto-Regressive Model

First Stage. The representations of the first stage are encoded and decoded by conditioning on the second stage and may be fully factorized as:

$$P(\tilde{\mathbf{y}}|\tilde{\mathbf{z}}) = \prod_{i=1}^{i=N} P(y_i|y_{i-1}, ..., y_1, D_2(\tilde{z})) \tag{7}$$

The quantized representations of the first stage are losslessly compressed using standard arithmetic coding where the conditional probabilities are estimated by a 3D pixel-CNN [20] which is conditioned on extra auxiliary representations $D_2(\tilde{z})$. The 3D pixel-CNN is trained with cross-entropy minimization for a correct quantized centre assignment:

$$P_{i,l}(\tilde{\mathbf{y}}) = p_\Theta(\tilde{y}_i = c_l|\tilde{y}_{i-1}, ..., \tilde{y}_1, D_2(\tilde{\mathbf{z}})) \tag{8}$$

Thus the total entropy for the bottleneck is estimated using cross-entropy as:

$$CE = H_1(\tilde{\mathbf{y}}|\tilde{\mathbf{z}}) = \mathbb{E}_{\tilde{\mathbf{y}}\sim P(\tilde{\mathbf{y}}|\tilde{\mathbf{z}})}[\sum_{i=1}^{i=N} -log(P_{i,l}(\tilde{y}_i))] \quad (9)$$

Second Stage. The representations of the second stage are encoded independently and the distribution is factorized as a product of conditionals:

$$P(\tilde{\mathbf{z}}) = \prod_{i=1}^{i=M} P(\tilde{z}_i|\tilde{z}_{i-1}, ..., \tilde{z}_1) \quad (10)$$

The second stage uses a separate 3D pixel-CNN [20], which is trained by minimizing:

$$CE = H_2(\tilde{\mathbf{z}}) = \mathbb{E}_{\tilde{\mathbf{z}}\sim P(\tilde{\mathbf{z}})}[\sum_{j=1}^{j=M} -log(P_{j,l}(\tilde{z}_j)] \quad (11)$$

The objective of the second stage is to learn the auxiliary features which help in compressing the first stage representations. Thus the gradients from Equation 9 are propagated to the second stage along with additional gradients from a reconstruction loss $mse(\tilde{\mathbf{y}}, D_2(\tilde{\mathbf{z}}))$.

Joint Optimization. The overall rate of optimization $Rate(\mathbf{x})$ incorporates the masks from both stages, that is $\mathbf{m_1}$ and $\mathbf{m_2}$ as the weights to the cross-entropy computation for a given index in the bottleneck. The overall entropy is thus given by:

$$Rate(\mathbf{x}) = H = \mathbf{m_1} H_1(\tilde{\mathbf{y}}|\tilde{\mathbf{z}}) + \mathbf{m_2} H_2(\tilde{\mathbf{z}}) \quad (12)$$

3.4. Incorporating Object Saliency

The saliency mask $\mathbf{s}$ such that $s_i \in \{0, 1\}$ is predicted by an off-the-shelf object saliency model [16], which was trained on MSRA10K data [9]. It is used in our compression model in two ways. Firstly, to mask quantized representations of the first stage, that is, more bits are allocated to the salient regions. Secondly, during the computation of distortion loss, to give higher weight to the reconstruction of the salient regions.

Salient Masking. The generated saliency mask is combined with an importance mask which helps in navigating the bit-rate convergence to a certain target value [26]. Similar to [26], the last channel of the bottleneck is treated as the importance mask $\mathbf{i} = E_1(\mathbf{x})[C_1]$. This importance mask is linearly combined with the saliency mask $\mathbf{s}$ to make the compression driven by saliency. As illustrated in Figure 4b, the final mask used is given by $\mathbf{m}_1 = \lambda_1\mathbf{s} + \lambda_2\mathbf{i}$. In practice, $\lambda_1 = \lambda_2 = 1$, this way the model is able to incorporate saliency while at the same time it is able to converge to a specified target bit-rate value.

This two-dimensional mask is expanded to match the dimensionality of the bottleneck [26]. Finally, the bottleneck is masked by a pointwise multiplication with the binarization of $\mathbf{m}_1$ as $\mathbf{y}_m = \mathbf{y} \odot \lceil \mathbf{m}_1 \rceil$.

Weighted Distortion Losses. We hypothesise that humans, in general, pay more attention to salient regions in the image. Thus, during training, a higher priority is given to the reconstruction of salient regions. This is achieved by decomposing the original and reconstructed images into salient and non-salient parts. Here distortion loss is computed on both separately and then linearly combined as:

$$w_1 D(\mathbf{x}\odot\mathbf{s}, \hat{\mathbf{x}}\odot\mathbf{s}) + w_2 D(\mathbf{x}\odot(\mathbf{1}-\mathbf{s}), \hat{\mathbf{x}}\odot(\mathbf{1}-\mathbf{s})) \quad (13)$$

Where $w_1 > w_2$, in practice, $w_1 = 0.75$ and $w_2 = 0.25$. Refer to the supplementary material for an illustration.

3.5. Model Optimization

The overall optimization is a rate-distortion trade-off (Equation 2). The rate is determined by Equation 12, the distortion is saliency driven and is governed by Equation 13 where the distortion function D is a linear combination of the Deep perceptual loss DPL (Equation 1) (LPIPS-Comp) and the mean-squared error between the original and the reconstructed images.

Training Details. Adam optimizer [18] is used, with an initial learning rate of 4×10^{-3} and a batch-size of 30. The learning rate is decayed by a factor of 10 in every two epochs (step-decay). Further, similar to [26], the rate term is clipped as $max(t, \beta R)$, to make the model converge to a certain bit-rate t. The training is done on the training set of ImageNet dataset the from Large Scale Visual Recognition Challenge 2012 (ILSVRC2012) [34], with the mentioned setup, convergence was observed in six epochs.

By varying the model hyper-parameters such as the number of channels in the bottlenecks (that is C_1 and C_2), the weight for distortion loss (α), the target bit-rate (t), multiple models were obtained in the bit-per-pixel range of 0.15 to 1.0. Similarly, the models for [26, 3] were reproduced at different bits-per-pixel values. Note that, in the case of [3], we used an MS-SSIM loss instead of MSE loss as was done in the original paper but this does not change the general conclusions of the paper. For Lee *et al.* [22], the authors provided us with the images from the Kodak dataset for human evaluations.

4. Results

The results of image compression are shown using human evaluations. Another way to compare compression methods is to judge how well the reconstructed images do on a computer vision task. The model for this task is pretrained on uncompressed images, specifically, object detec-

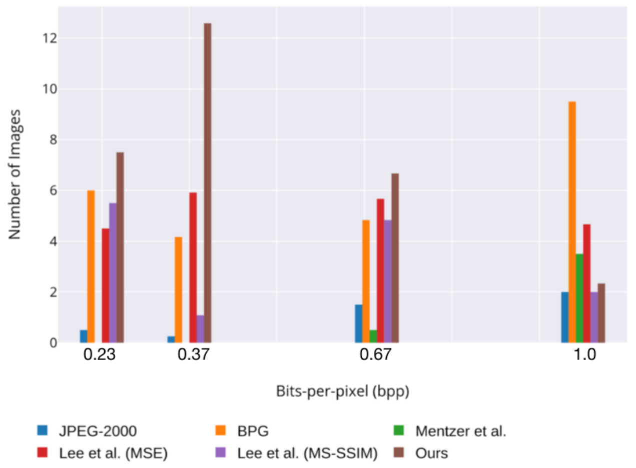


Figure 5: Human evaluations on the Kodak dataset. The y-axis shows the number of images for which a given method performs best. The x-axis shows the BPP values at which the comparisons were performed.

tion and segmentation are investigated. Each lossy compression method creates a certain kind of artefacts that impact the task. For example, Dwibedi *et al.* [10] show that for object detectors such as Faster-RCNN [31] region-based consistency is important and pixel-level artefacts can affect the performance. Thus a compression method which distorts the region based consistency will perform poorly for object detection. We view this as another evaluation of a compression technique. We demonstrate that on all three metrics our approach outperforms competing methods. Section 2 shows that the widely used metrics PSNR and MS-SSIM are inadequate for judging different compression methods. However, for completeness, results on PSNR and MS-SSIM are also reported.

Comparison with other compression methods. We compare the proposed method with the state-of-the-art image compression models from Lee *et al.* [22] based on variational autoencoders. We also compare to a single-level auto-regressive compression method Mentzer et al [26] and two engineered methods BPG [5] and JPEG-2000 [37]. We use the Kakadu[1] implementation for JPEG-2000 and use BPG in the 4:4:4 chroma format following [26, 32].

Quantitative Human Evaluations. An extensive human evaluation study is performed using five compression approaches, across four different bits-per-pixel values (0.23, 0.37. 0.67, 1.0) on the Kodak dataset [12]. For the human evaluation, we follow the setup described in Section 2.1. The comparison is made in a pair-wise manner for all 15 possible combinations of the six methods ($^6C_2$). For each such pair-wise comparison, we obtain five evaluations

[1]http://kakadusoftware.com/

Method	**0.23**	**0.37**	**0.67**	**1.0**
JPEG-2000 [37]	23.2	29.1	34.4	36.8
BPG [5]	25.2	32.5	35.4	37.7
Mentzer *et al.* [26]	25.5	30.2	34.5	36.6
Lee *et al.* [22] (MSE)	28.3	-	36.2	37.6
Lee *et al.* [22] (MS-SSIM)	27.2	32.5	-	37.6
Ours (MSE + DPL)	**29.3**	**33.7**	**36.6**	**37.9**

Table 1: Object Detection on MS-COCO 2017 [23] validation set using Faster-RCNN [31]. The performance is reported using AP@[.5:.95], that is an average over different scales of IoU. Note that the performance on the original (uncompressed) images is 40.1%.

Method	**0.23**	**0.37**	**0.67**	**1.0**
JPEG-2000 [37]	20.2	25.4	30.1	32.2
BPG [5]	22.0	28.5	30.8	32.2
Mentzer *et al.* [26]	9.3	10.5	11.9	22.0
Lee *et al.* [22] (MSE)	25.4	-	32.2	**33.2**
Lee *et al.* [22] (MS-SSIM)	25.1	28.9	-	**33.2**
Ours (MSE + DPL)	**26.1**	**30.0**	**32.3**	**33.2**

Table 2: Instance segmentation on MS-COCO 2017 [23] validation set using Mask-RCNN [14]. The performance is reported using an average over multiple IoU values. Note that the performance on the original (uncompressed) images is 35.2%.

and determine the better performing method. Across different pair-wise comparisons, the method which wins the most number of times performs best for the given image at a bit-rate. Thus, at each bit-rate, we count the number of images for which a method performs best among the set of competing methods. Figure 5 shows that our method is best according to the human evaluation across three bit-rate values (0.23, 0.37. 0.67). At a relatively higher BPP of 1.0 BPG out-performs all the other methods.

Qualitative Comparison. Please see Figure 6 where a comparison from Kodak dataset is shown. Notice that, our method preserves the fine-grained details of the face better than the other methods (see above the lip, paint patterns around the eye). For more such examples we refer the reader to the supplementary material.

Object Detection. A pre-trained Faster-RCNN [31] model with a $ResNet-101$[15] based backbone is used. With the original MS-COCO images, this model attains a performance of 40.1% AP. For each compression method, we compress and reconstruct the image at four different bit-rate values: $0.23, 0.37, 0.67, 1.0$ (same values as used for human evaluation) and the reconstructed images are evaluated for object detection. The performance of competing compression methods are reported in Table 1. It can be seen

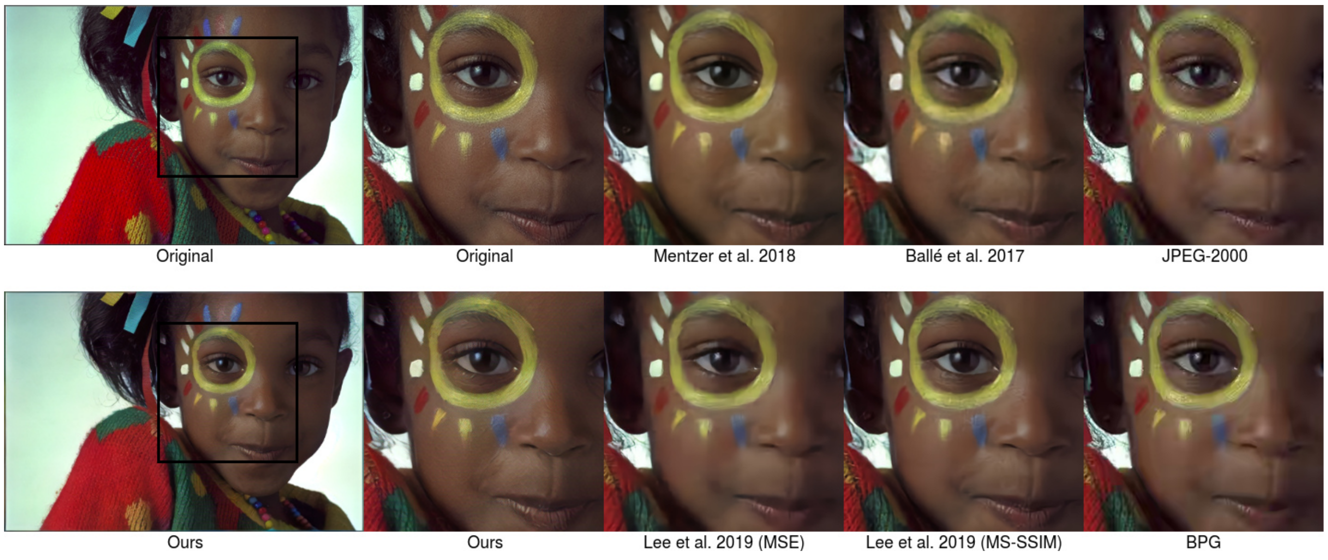


Figure 6: A qualitative example from the Kodak dataset [12] at **0.23** BPP. Notice our method better captures the fine-grained details (lines above the lip, yellow circle around the eye) better than other approaches.

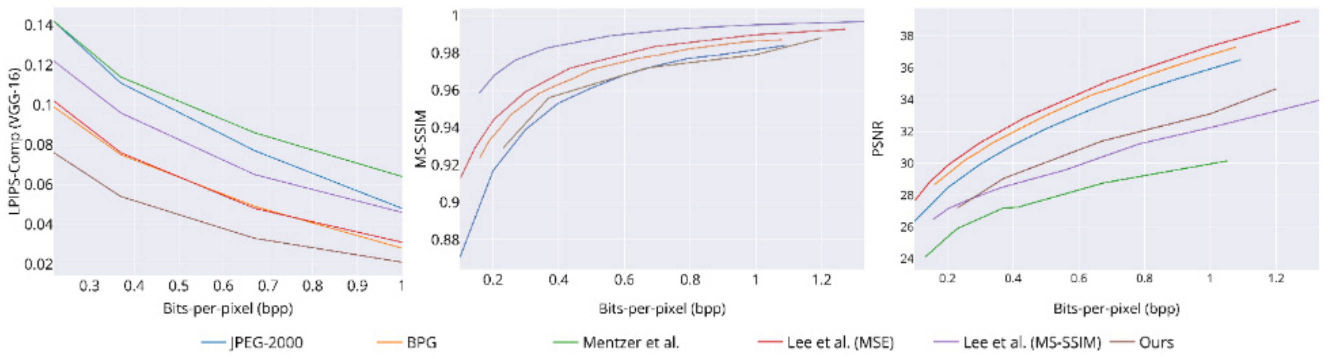


Figure 7: Different metrics on the Kodak dataset [12]. ***Left***: (Perceptual Similarity) LPIPS-Comp (VGG-16) (Section 2.2) vs bits-per-pixel (BPP) (Lower is better). ***Middle***: MS-SSIM vs bits-per-pixel (BPP). (Higher is better). ***Right***: PSNR vs bits-per-pixel (BPP). (Higher is better). Best viewed in color.

that the proposed method outperforms the competing methods at all bit-rates.

Instance segmentation. Mask-RCNN [14] with a *ResNet-101* backbone is used for the task. The performance of different compression techniques is reported in Table 2. It is observed that while Mentzer *et al.* [26] perform comparable to engineered methods on object detection, it performs far worse on Instance segmentation. It can be seen in Table 2, that our method outperforms the competing methods at lower bit-rates while [22] performs identically at 1.0 bits-per-pixel.

PSNR / MS-SSIM / LPIPS-Comp. For completeness, the performance of these models are shown on standard PSNR and MS-SSIM metrics. Figure 7 (right) for PSNR, Figure 7 (middle) for MS-SSIM and Figure 7 (left) for LPIPS-Comp (Section 2.2).

5. Conclusions

An approach is proposed for deep image compression. The method trains and evaluates using a deep perceptual metric. It uses a hierarchical auto-regressive framework and takes object saliency into account. On human evaluations, the model performs better than the state-of-the-art on low bit rates. Images obtained from our model provide the best object detector and image segmentation results when compared to the other image compression schemes.

Acknowledgement

The authors thank Joel Chan and Peter Hallinan for helping with the human evaluations.

References

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

[2] Eirikur Agustsson, Michael Tschannen, Fabian Mentzer, Radu Timofte, and Luc Van Gool. Generative adversarial networks for extreme learned image compression. *arXiv preprint arXiv:1804.02958*, 2018.

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

[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.

[5] Fabrice Bellard. Bpg image format, 2014.

[6] Yochai Blau, Roey Mechrez, Radu Timofte, Tomer Michaeli, and Lihi Zelnik-Manor. The 2018 pirm challenge on perceptual image super-resolution. In *ECCV*, 2018.

[7] Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In *CVPR*, 2018.

[8] Thomas Boutell. Png (portable network graphics) specification version 1.0. Technical report, 1997.

[9] Ming-Ming Cheng, Niloy J Mitra, Xiaolei Huang, Philip HS Torr, and Shi-Min Hu. Global contrast based salient region detection. *TPAMI*, 2014.

[10] Debidatta Dwibedi, Ishan Misra, and Martial Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *ICCV*, 2017.

[11] Forbes. How much data do we create every day? the mind-blowing stats everyone should read, 2018.

[12] Rich Franzen. Kodak lossless true color image suite. *source: http://r0k. us/graphics/kodak*, 1999.

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014.

[14] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[16] Qibin Hou, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, and Philip HS Torr. Deeply supervised salient object detection with short connections. In *CVPR*, 2017.

[17] J Jiang. Image compression with neural networks–a survey. *Signal processing: image Communication*, 1999.

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[20] Alexander Kolesnikov and Christoph H Lampert. Pixelcnn models with auxiliary variables for natural image modeling. In *ICML*, 2017.

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

[22] Jooyoung Lee, Seunghyun Cho, and Seung-Kwon Beack. Context-adaptive entropy model for end-to-end optimized image compression. *ICLR*, 2019.

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[24] SP Luttrell. Image compression using a neural network. In *IGARSS*, 1988.

[25] Detlev Marpe, Heiko Schwarz, and Thomas Wiegand. Context-based adaptive binary arithmetic coding in the h. 264/avc video compression standard. *TCSCT*, 2003.

[26] Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, and Luc Van Gool. Conditional probability models for deep image compression. In *CVPR*, 2018.

[27] David Minnen, Johannes Ballé, and George D Toderici. Joint autoregressive and hierarchical priors for learned image compression. In *NeurIPS*, 2018.

[28] PAUL Munro and DAVID Zipser. Image compression by back propagation: an example of extensional programming. *Models of cognition: rev. of cognitive science*, 1989.

[29] Yash Patel, Srikar Appalaraju, and R Manmatha. Deep perceptual compression. *arXiv preprint arXiv:1907.08310*, 2019.

[30] Yash Patel, Srikar Appalaraju, and R Manmatha. Human perceptual evaluations for image compression. *arXiv preprint arXiv:1908.04187*, 2019.

[31] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *TPAMI*, 2017.

[32] Oren Rippel and Lubomir Bourdev. Real-time adaptive image compression. *arXiv preprint arXiv:1705.05823*, 2017.

[33] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 1986.

[34] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

[35] Claude Elwood Shannon. A mathematical theory of communication. *Bell system technical journal*, 1948.

[36] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[37] Athanassios Skodras, Charilaos Christopoulos, and Touradj Ebrahimi. The jpeg 2000 still image compression standard. *IEEE Signal processing magazine*, 2001.

[38] Lucas Theis and Matthias Bethge. Generative image modeling using spatial lstms. In *NeurIPS*, 2015.

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

[40] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. In *NeurIPS*, 2016.

[41] Gregory K Wallace. The jpeg still picture compression standard. *IEEE TCE*, 1992.

[42] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018.

[43] R Zhang, P Isola, E. Efros A.A., Schectman, and Wang O. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.