Where's Waldo?: Visual Question Answering on Extreme Lossy Compressed Images using a Variational Approach with Scale Hyperprior

Arnas Uselis Partha Ghosh Sandeep Inuganti

{arnas.uselis, partha.ghosh, sandeep.inuganti}@student.uni-tuebingen.de

Eberhard Karls Universität Tübingen

Abstract

Image compression has seen a new turn with the introduction of deep neural network models like Variational Autoencoders and their variants. In this work, we attempt to use these deep probabilistic compression models and perform Visual Question Answering (VQA) at various compression levels with a focus on extremely lossy image compression. We train the compression models with and without back-propagated supervision from the VQA task at various model specifications and analyze the qualitative and quantitative results. We achieve similar performance on VQA task with the original (uncompressed) images as with lossy reconstructed images. We also compare our model to classical image compression methods like JPEG and report the corresponding results on VQA, and compression metrics: bits-per-pixel (BPP) and peak signal-to-noise ratio (PSNR).

1. Introduction

Image compression is one of the fundamental problems in data compression, which has many practical uses in storing and transmitting data. Although image compression algorithms can be classified into lossless and lossy compression, this work aims to experiment on the latter. Lossy image compression is still dominated by classical, i.e. non machine learning-based compression methods [23, 6] to some extent. Moreover, in high-quality regime classical algorithms are still somewhat difficult to beat. However, machine learning based methods [4, 5] shine in the low-quality regime. In this work, we aim to achieve an extremely lossy compression method on images i.e. having an extremely low bit rate. On top of that, the compression should be in such a way that we can reconstruct the image with some similar visual cues as the original image. In this way, we can be able to perform Visual Question Answering [2] (VQA) task on the reconstructed image.

To achieve an end-to-end model we use a variational autoencoder [15] based compression technique introduced in [4]. An end-to-end model is required so that the errors from the VQA task are back-propagated through to the autoencoder-based compression method. We choose an improvement over [4] by the same authors in [5] which uses a scale hyperprior, we elaborate more on this in section 2. As for the VQA task, we choose a strong baseline method [14]. Finally, we combine these two models in an end-to-end way to achieve our goals. With this work, we aim to get interesting insights on deep probabilistic compression models, as the work is at the intersection of data compression (VAE based) and interpretability (via VQA). This is also a work where the qualitative results are somewhat more important than quantitative results considering the interpretability and reconstruction problems. In this work we discuss and report on the following:

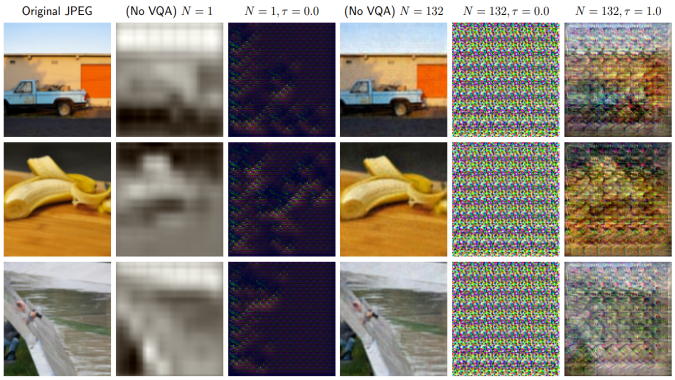

Figure 1. Some initial reconstruction results from our model with different latent sizes and reconstruction loss weighing (τ). This figure is analyzed more in section 4.3.

- Quantitative comparison of VQA task performance with original, reconstructed images with [5] and other classical methods like JPEG [23], AVIF [6]

- Analyze the effect of different latent vector sizes on the compression metrics (e.g. bits per pixel) and its effect on the quality of reconstructed images.

- Analyze the effect of weighing the different loss functions in our overall objective and observe the change in the respective qualitative and quantitative results.

2. Related Work

2.1. Classical Image Compression

Transform coding [8] is the leading approach when it comes to compressing image data. In this technique, a single image's pixel data is mapped into an alternative representation, which gets quantized and losslessly compressed using entropy coding methods, such as arithmetic coding [24], or less popularly, Huffman coding [12]. Note that even though the quantized representation is compressed losslessly, the initial mapping to the latent space does not have to be.

Concretely, in the encoding step, we assume that an image $\mathbf{x}$ gets transformed into the latent space vector $\mathbf{y}$ by a parametric analysis transform $g_a : \mathbf{y} = g_a(\mathbf{x})$ and gets quantized into $\hat{\mathbf{y}}$ by the quantizer $Q : \hat{\mathbf{y}} = Q(\mathbf{y})$, which can then be compressed losslessly using an entropy model $p_{\hat{\mathbf{y}}}(\hat{\mathbf{y}})$, and transmitted. At the other end in the decoding step, the quantized representation can be decoded lossessly to recover the $\hat{\mathbf{y}}$ and an approximation to the original image can be carried out by the means of the parametric synthesis transform $g_s : \mathbf{x} \approx g_s(\hat{\mathbf{y}})$. In classical methods, the parametric analysis $g_a(\cdot)$ and synthesis $g_s(\cdot)$ transforms are usually hand-engineered and/or linear.

The most prominent example of such lossy methods is the JPEG standard [23], which, on a high level, works by dividing the image into fixed-size blocks, transforming each of them using Discrete Cosine Transform (DCT) and discarding high-frequency information.

A more recent and advanced image compression standard is AVIF [6] which uses, High Efficiency Video Coding (HEVC) [18] for compression, an extension of the widely used compression technique known as Advanced Video Coding (AVC) [19] which is a video compression standard based on block-oriented, motion-compensated integer-DCT coding. In practice, AVIF has been shown to achieve the highest bitrate savings across many objective metrics among popular compression techniques [7].

2.2. Deep Learning-Based Methods

Similarly to the classical techniques, most of the modern compression methods follow transform coding, but the transform functions $g_a(\cdot), g_s(\cdot)$ are usually specified by neural networks, which in image compression, usually take the form of CNNs, that exploit spatial dependencies in the input domain. The architecture roughly follows an autoencoder [10] structure: an input gets non-linearly reduced into a lower-dimensional latent representation in the encoding step by $\mathbf{y} = g_a(\mathbf{x})$, and its quantized representation $\hat{\mathbf{y}} = Q(\mathbf{y})$ gets upsampled into the input domain to form an approximate reconstruction of the input $\mathbf{x} \approx g_s(\hat{\mathbf{y}})$.

Since we allow for trainable transformations, the training objective is to minimize the expected bit length of the encoded code (in the quantized space) (referred to as the rate) as well as the reconstruction error (referred to as distortion), giving rise to the following loss function [17]:

$$R + \lambda D = \underbrace{\mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}[-\log_2 p_{\hat{\mathbf{y}}}(\lfloor g_a(\mathbf{x})\rceil)]}_{\text{rate}} + \lambda \cdot \underbrace{\mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}[d(\mathbf{x}, g_s(\lfloor f(\mathbf{x})\rceil))]}_{\text{distortion}} \quad (1)$$

where now we assume the quantizer to take the rounding to the nearest integer operation, $Q(\mathbf{y}) = \lfloor \mathbf{y} \rceil$, all the expectations are taken with respect to unknown image distribution $p_{\mathbf{x}}(\mathbf{x})$, $d(\cdot, \cdot)$ is the distance function, usually mean squared distance, and λ determines the rate-distortion trade-off. The rate is the lower-bound of the exepected bitstream length and is minimized when the the true marginal of the latent representation is exactly equal to the marginal specified by the quantized analysis network.

This type of image compression is usually achieved by utilizing autoencoders [3, 5, 20, 17]. However, a probabilistic view of some of those models [3, 5] show that they can equivalently be framed as variational autoencoders [15]. Under this view, [3] can be seen as implementing its entropy model as a fully factorized probability distribution. Ballé et al. [5] address this by introducing a hyperprior, under which the (conditional) entropy model becomes independent, which requires to send additional information, but has been shown to produce better results. This line of work has been improved by further extending [5] by [17] learning hierarchical priors and making the encoding context-dependent.

One challenge of optimizing the (1) objective directly, is the parameters optimization using backpropagation through the rounding quantization function $Q(\cdot)$ due to the gradients being zero everywhere except at integer-values, where the gradients are undefined. However, relaxing the problem by adding uniform noise [3] at training time instead of integer-rounding, or approximating the gradient of the rounding operation [20], have been shown to alleviate the problem.

2.3. Visual Question Answering

Visual Question Answering, first introduced in [2] is one of the prominent tasks in visual scene understanding, and

connecting natural language processing techniques to computer vision. Given a scene (image) and a question in natural language related to the scene, the VQA task is to output an informed answer based on the visual cues and the subject in the scene. Therefore, a VQA model typically needs a more intricate understanding of the scene together with complex reasoning than other models in tasks like image captioning [22]. After the introduction of this task, there have been many advances in this area however a strong yet simple baseline was introduced in [14].

Whereas the original work [2] only had an LSTM [11] embedding over a question and a Hadamard product of that embedding with a convolutional neural network embedding of the image, in this work [14], the authors incorporate attention mechanism [21] over the input image's feature maps and the question embedding from an LSTM and then use this attention mechanism to answer the question.

3. Methodology

3.1. Image Compression Using VAEs

We use the compression method introduced in [5][1] and make changes to its optimization function. Since [5] builds upon the work of [3], we exposit the methodology of this work first, discuss the shortcomings, and overview the improvements introduced in [5].

Due to the problems of quantization in optimizing the (1), alluded in subsection 2.2, we adopt the approach introduced in [3] of adding component-wise uniform noise to the latent representation $\mathbf{y}$ in the training phase, and denote such representation as $\tilde{\mathbf{y}}_i = \mathbf{y}_i + u_i, u_i \sim \mathcal{U}(-\frac{1}{2}, \frac{1}{2})$.

We adopt the probabilistic viewpoint and carry out the variational inference procedure to approximate the true unknown posterior $p_{\tilde{\mathbf{y}}|\mathbf{x}}(\tilde{\mathbf{y}}|\mathbf{x})$ using a tractable distribution $q(\tilde{\mathbf{y}}|\mathbf{x}) = \prod_i \mathcal{U}(\tilde{y}_i|y_i - \frac{1}{2}, y_i + \frac{1}{2})$, where $\mathbf{y} = g_a(\mathbf{x})$, by minimizing the expectation of KL divergence over the true data distribution $p_{\mathbf{x}}$:

$$\begin{aligned}\mathbb{E}_{\boldsymbol{x}\sim p_{\mathbf{x}}} D_{\mathrm{KL}}\left[q \| p_{\tilde{\boldsymbol{y}}|\boldsymbol{x}}\right] = \mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}\mathbb{E}_{\tilde{\mathbf{y}}\sim q}\left[\log\left(\frac{q(\tilde{\mathbf{y}}|\mathbf{x})}{p(\tilde{\mathbf{y}}|\mathbf{x})}\right)\right] = \\ \underbrace{\mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}\mathbb{E}_{\tilde{\mathbf{y}}\sim q}[\log(q(\tilde{\mathbf{y}}|\mathbf{x}))]}_{\mathbb{E}_{\tilde{\mathbf{y}}\sim q}[\log(\prod_i \mathcal{U}(\tilde{y}_i|y_i-\frac{1}{2},y_i+\frac{1}{2}))]=0} - \mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}\mathbb{E}_{\tilde{\mathbf{y}}\sim q}[\log p(\tilde{\mathbf{y}}|\mathbf{x}))] = \\ -\mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}\mathbb{E}_{\tilde{\mathbf{y}}\sim q}[\log(p(\tilde{\mathbf{y}})p(\mathbf{x}|\tilde{\mathbf{y}}))] + \mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}[\log p(\mathbf{x})] = \\ -\mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}\mathbb{E}_{\tilde{\mathbf{y}}\sim q}[\log(p(\tilde{\mathbf{y}})] - \mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}\mathbb{E}_{\tilde{\mathbf{y}}\sim q}[\log p(\mathbf{x}|\tilde{\mathbf{y}}))] \\ +\text{const} \end{aligned} \tag{2}$$

where in the first equality we used the definition of $\mathrm{D_{KL}}$, and in the second expanded the terms using Bayes' theorem and used the fact that the density of unit-width uniform distribution is equal to 1. Defining the likelihood as a Normal distribution $p_{\mathbf{x}|\tilde{\mathbf{y}}}(\mathbf{x}|\hat{\mathbf{y}}) = \mathcal{N}(\mathbf{x}; \tilde{\mathbf{x}}, (2\lambda)^{-1}\mathbb{1})$, where $\tilde{\mathbf{x}}$ is the reconstructed image, $\tilde{\mathbf{x}} = g_s(\tilde{\mathbf{y}})$. Evaluating (2) using this specification, yields precisely the the original loss function (1) with an MSE distance function and the rate specified by the prior. Under this interpretation, the λ parameter controls how sharp the reconstructions are expected to be. It was shown that defining the prior as component-wise fully factorized discrete probability distribution yields the formulation of the original model in [3]. Concretely, the prior is assumed to satisfy $p(\tilde{\mathbf{y}}) = \prod_i \left(p_{y_i|\psi^{(i)}}(\psi^{(i)}) * \mathcal{U}(-\frac{1}{2}, \frac{1}{2})\right)(\tilde{y}_i)$, where the uniform noise is convoled with the density of p_y to enable a better match to the prior of the true marginal [3]. In order to carry out arithmetic coding we need to be able to evaluate p_{y_i}, which requires to know $\boldsymbol{\psi}^{(i)}$. To do that, we impose that each parameter $\boldsymbol{\psi}^{(i)}$ would consist of 10 values per unit interval, using which piecewise linear interpolation can be carried out at any point of the marginal. We can then optimize these parameters by minimizing negative expected likelihood using stochastic gradient descent: $L_\psi(\{\boldsymbol{\psi}^{(i)}\}_{i=0}) = -\mathbb{E}_{\tilde{\mathbf{y}}}[\sum_i p_{\tilde{y}_i}(\tilde{y}_i|\boldsymbol{\psi}^{(i)})]$ and normalizing after each update step.

One drawback of this approach is that fully-factorized prior is unable to capture correlations, and has been addressed by introducing a latent variable under which the prior becomes independent [5]. To achieve this, a hyperprior $\tilde{\mathbf{z}}$ is introduced and $\hat{y}_i$ is modeled as a zero-mean Normal random variable with a standard deviation specified by another neural network h_s: $p(\tilde{\mathbf{y}}|\tilde{\mathbf{z}}) = \prod_i \left(\mathcal{N}(0, h_s(\tilde{\mathbf{z}})^2) * \mathcal{U}(-\frac{1}{2}, \frac{1}{2})\right)(\tilde{y}_i)$. Then, the role that $\mathbf{y}_i$ had in fully-factorized prior model is now transferred to $\hat{\mathbf{z}}$, which takes exactly the same factorizing form. Then, writing out the approximate posterior by incorporating the new latent variable we get: $q(\tilde{\mathbf{y}}, \tilde{\mathbf{z}}|\mathbf{x}) = q(\tilde{\mathbf{y}}|\mathbf{x})q(\tilde{\mathbf{z}}|\mathbf{x}, \tilde{\mathbf{y}})$, but here we assume that $\tilde{\mathbf{z}}$ is only dependent on $\tilde{\mathbf{y}}$, that is, it is sufficient to estimate the spatial distribution of the standard deviations: $q(\tilde{\mathbf{y}}, \tilde{\mathbf{z}}|\mathbf{x}) = q(\tilde{\mathbf{y}}|\mathbf{x})q(\tilde{\mathbf{z}}|\tilde{\mathbf{y}})$, which yields:

$$\begin{aligned} q(\tilde{\mathbf{y}}, \tilde{\mathbf{z}}|\mathbf{x}) = \prod_i \mathcal{U}\left(\tilde{y}_i|y_i - \frac{1}{2}, y_i + \frac{1}{2}\right) \\ \cdot \prod_j \mathcal{U}\left(\tilde{z}_j|z_j - \frac{1}{2}, z_j + \frac{1}{2}\right) \end{aligned} \tag{3}$$

where we can assume factorization because $q(\tilde{\mathbf{y}}|\mathbf{x})$, $q(\tilde{\mathbf{z}})$ and $q(\tilde{\mathbf{y}}|\tilde{\mathbf{z}})$ are assumed to factorize. Finally, carrying out variational approximation of the true posterior similarly to (2), yields the minimization objective:

$$\begin{aligned}\mathbb{E}_{\boldsymbol{x}\sim p_x} D_{\mathrm{KL}}\left[q \| p_{\tilde{\boldsymbol{y}},\tilde{\boldsymbol{z}}|\boldsymbol{x}}\right] = \mathbb{E}_{\boldsymbol{x}\sim p_{\boldsymbol{x}}}\mathbb{E}_{\tilde{\boldsymbol{y}},\tilde{\boldsymbol{z}}\sim q}\left[-\log p_{\boldsymbol{x}|\tilde{\boldsymbol{y}}}(\boldsymbol{x} \mid \tilde{\boldsymbol{y}}) \right. \\ \left. - \log p_{\tilde{\boldsymbol{y}}|\tilde{\boldsymbol{z}}}(\tilde{\boldsymbol{y}} \mid \tilde{\boldsymbol{z}}) - \log p_{\tilde{\boldsymbol{z}}}(\tilde{\boldsymbol{z}})\right] + \text{const}\end{aligned} \tag{4}$$

where the $q(\cdot, \cdot|\cdot)$ is joint factorized variational posterior over $\tilde{\mathbf{y}}, \tilde{\mathbf{z}}$ given the input $\mathbf{x}$. The first term in the loss is

[1] Compression code available at https://github.com/liujiaheng/compression

the likelihood that encapsulates the distortion (which can be simply a mean squared error on pixels), the second and third terms can be seen as the cross entropy losses encoding $\tilde{y}$ and $\tilde{z}$, respectively.

Differently from fully-factorized prior case, the parameters $\boldsymbol{\psi}$ still need to be found, but only for $\hat{\mathbf{z}}$, while the arithmetic encoding (AE) can query the prior distribution directly: $p_{\tilde{y}_i}(\hat{y}_i|\sigma_i) = \int_{\hat{y}_i-\frac{1}{2}}^{\hat{y}_i+\frac{1}{2}} \mathcal{N}(y|0,\hat{\sigma}_i)\,\mathrm{d}y$. This method only requires additional encoding of the hyperprior $\hat{\mathbf{z}}$, that can be sent as side information.

In this work, the compressor follows the following sequence of processes in the testing phase: The encoder part passes the input image $\mathbf{x}$ through $g_a(\cdot)$, to get the latents $\mathbf{y}$ with spatially varying standard deviations. These latents are then passed through $h_a(\cdot)$, to get a "summary" $\mathbf{z}$ of the distribution of standard deviations. Now, $\mathbf{z}$ is quantized (via quantizer $Q(\cdot)$) to get $\hat{\mathbf{z}}$, which is then compressed using AE, and transmitted as side information. Then, the encoder uses this quantized side information $\hat{\mathbf{z}}$ to estimate $\hat{\boldsymbol{\sigma}}$, which is the spatial distribution of standard deviations, and finally the encoded representation $\hat{\mathbf{y}}$ is produced by using both $\hat{\mathbf{y}}$ and $\boldsymbol{\sigma}$.

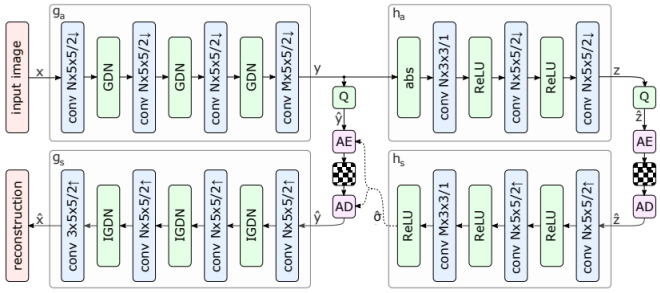

Figure 2. Network architecture of the hyperprior model taken from [5]. Q represents quantization operation. AE and AD are the arithmetic encoding and decoding operations respectively.

Now, the arithmetic decoding (AD) part of the decoder first recovers $\hat{\mathbf{z}}$ from the compressed side information signal. It then passes the recovered $\hat{\mathbf{z}}$ through $h_s(\cdot)$ to obtain $\hat{\boldsymbol{\sigma}}$, which, together with the latent $\hat{\mathbf{y}}$, is then used by the decoder to recover the quantized latent $\hat{\mathbf{y}}$. Then, it passes the recovered $\hat{\mathbf{y}}$ through $g_s(\cdot)$ to get the reconstructed image $\hat{\mathbf{x}}$. The overall loss function for this compression method is given by (3). The flow of the method together with the architecture used is illustrated in Figure 2.

3.2. Incorporating VQA

Now we connect the VQA model[2] to the reconstructed image and perform the VQA task as previously discussed in section 2. We use the VQA model loss function from [14] along with the compression loss function (3) to train our model end-to-end. The final objective becomes:

$$\mathcal{L} = \mathcal{L}_{\text{rate}} + \tau\mathcal{L}_{\text{dist}} + \mathcal{L}_{\text{VQA}} \tag{5}$$

Where $\mathcal{L}_{\text{dist}}$ is the distortion loss implemented as mean squared error loss, $\mathcal{L}_{\text{rate}}$ is the entropy losses over the latent and hyper-latent, $\mathcal{L}_{\text{VQA}}$ is the VQA classification loss over the answer classes, and τ is the weight we use for $\mathcal{L}_{\text{dist}}$. Due to space constraints, we briefly discussed the methodology, however, we encourage the reader to go through the original works [5, 14] for more elaborate derivations and explanations.

4. Experiments and Results

We conduct various experiments by analyzing both qualitative and quantitative results on hyper-parameters like latent size and weight of distortion loss, τ. We also report the VQA results of other classical compressors like JPEG. First, we overview the implementation details.

4.1. Implementation Details

Dataset: We use the VQA-V1 [1] dataset containing around 120,000 images for training and validation, which are accompanied by close to 400,000 questions. The top 3000 most frequent answers are taken as labels.

Pre-processing: Since the VQA-V1 dataset images vary in size and the pre-trained VQA model was trained on fixed-size inputs, we resize the images with respect to the smaller dimension and take center 128×128 crop.

Training The pre-processed crop is passed through the compressor, we take the reconstruction, standardize its values, and then pass it through a Resnet [9], which produces a 2048-dimensional feature vector. This feature vector is passed through the VQA model and the loss is calculated. However, during training we freeze the Resnet and VQA model weights, but backpropagate the gradients through to the compression model. For the latent and hyper latent sizes, we always have the same size for both: $16 \times 16 \times N$ where N can be $1, 32, 132$ channels. Therefore in the quantitative results we specify N and also indicate if we used VQA loss gradients in training the whole model end-to-end. The training is carried using batch size of 64 and using the hyperparameters specified in [5]. We could not experiment with bigger latent sizes due to computational resource limitations. As such, instead of controlling the trade-off between the rate and distortion through parameter λ, we instead control it through the size of the latent size to expedite the training and keep fixed $\lambda = 4096$.

4.2. Quantitative Results

First we show the comparison between various latent sizes, their compressions and VQA performances with

[2]VQA code available at https://github.com/Cyanogenoid/pytorch-vqa

Model specifications	$\tau = 0$				$\tau = 0.1$				$\tau = 1$				$\tau = 10$			
	BPP	PSNR	Acc	MSE	BPP	PSNR	Acc	MSE	BPP	PSNR	Acc	MSE	BPP	PSNR	Acc	MSE
$N = 1$ [5]	0.004	5.72	41.3	0.268	0.005	6.25	41.2	0.237	0.008	8.38	42.7	0.145	0.004	11.57	41.9	0.070
$N = 32$ [5]	0.4256	-32.01	44.56	6.3e3	0.236	9.07	41.4	0.171	0.258	11.51	46.9	0.071	0.299	17.68	46.9	0.017
$N = 132$ [5]	3.031	-302.5	40.0	1.7e29	1.132	-2.23	38.9	1.677	0.893	12.31	51.5	0.059	0.922	19.02	50.9	0.011

Table 1. Each row represents the results from VQA-supervised $N = 1, 32, 132$ models respectively. The models are trained by taking supervision from $\mathcal{L}_{\text{VQA}}$ along with $\mathcal{L}_{\text{dist}}, \mathcal{L}_{\text{rate}}$.

Model specifications	BPP	PSNR	Acc
(VQA-supervised) $N = 1$ [5]	0.008	8.38	42.7
(VQA-supervised) $N = 32$ [5]	0.258	11.51	46.9
(VQA-supervised) $N = 132$ [5]	0.893	12.31	51.5
(No VQA) $N = 1$ [5]	0.024	16.31	40.0
(No VQA) $N = 32$ [5]	0.579	24.35	51.2
(No VQA) $N = 132$ [5]	1.436	27.31	53.0

Table 2. Comparison of [5] with and without $\mathcal{L}_{\text{VQA}}$ supervision in the end to end training.

varying τ for the $\mathcal{L}_{\text{dist}}$, which is implemented as per-pixel mean squared error loss. We report the corresponding bits per pixel (BPP), PSNR (peak signal-to-noise-ratio), the accuracy of those compressed images on the VQA task, and the reconstruction (MSE) error, which can be seen in Table 1. We observe that with increasing τ the accuracy increases and the best VQA accuracy is achieved at $\tau = 1$. We also see that using latent size 132 results in the best VQA performance (which is unsurprising given the model's capacity), however, we pay a higher cost in BPP. We also observe that for a higher latent size, τ is a very important hyper-parameter; if we set it to a sufficiently lower value the compression, reconstruction, and VQA tasks are done very poorly. On the other hand, with a better τ, models with bigger latent size outperform their counterparts in terms of PSNR, MSE and accuracy. However, the lower MSE value does not correspond to a better reconstruction in this case as we will discuss in section 4.3.

Next we analyze the results of the compression model without $\mathcal{L}_{\text{VQA}}$ supervision against model with $\mathcal{L}_{\text{VQA}}$ in Table 2. The results for models without VQA are obtained by compressing the image is first via [5] and then the VQA accuracy is calculated by performing VQA task on the reconstructed image. We can clearly see that without the VQA supervision the reconstructed images yield better accuracy and PSNR but the BPP metric is costlier. However with a lower latent size ($N = 32$) we get similar results as the best performing model with VQA supervision ($N = 132$), but also we have better BPP and PSNR. However, not only the VQA accuracy is important here, but we also have to look at the actual reconstructions to decide which is a better approach: with VQA or without VQA. We will do that in the next section. It is also important to note that for the lowest latent size, $N = 1$ the model with VQA supervision outperforms it's corresponding competitor because, at that extremely lossy level of compression both the reconstructions (with and without VQA) are bad, but here is where VQA supervision actually helps the model to get a better accuracy.

Model specifications	BPP	PSNR	Acc
VQA [14] (original JPEGs)	-	-	53.4
JPEG [23] 1%	0.302	20.60	44.3
JPEG [23] 50%	1.519	29.35	52.1
JPEG [23] 90%	3.649	37.49	53.3
AVIF [6] 1%	0.232	29.47	51.2
AVIF [6] 50%	1.224	32.46	53.0
AVIF [6] 90%	2.604	33.03	52.7
Ours (No VQA) $N = 1$ [5]	0.024	16.31	40.0
Ours (No VQA) $N = 32$ [5]	0.579	24.35	51.2
Ours (No VQA) $N = 132$ [5]	1.436	27.31	53.0

Table 3. Comparison of [5] with JPEG and AVIF compression methods. The quality of the compressed image is shown in % next to JPEG, AVIF.

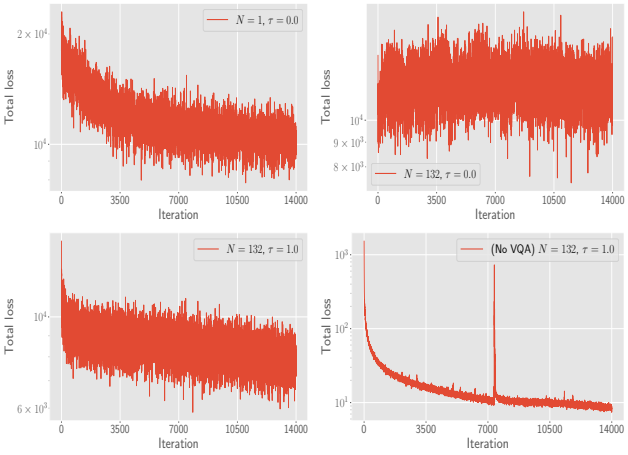

Figure 3. The training loss curves of the total loss over training iterations for various model specifications.

Finally, we move onto the results of the compression model against other classical image compression techniques like JPEG and AVIF, presented in Table 3. We can see that increasing latent size up to $N = 132$ results in a better VQA accuracy and better PSNR, but the BPP is almost tripled than when using latent size 32. Against JPEG, our model performs slightly better in terms of BPP-accuracy trade-off, however, against a relatively new method AVIF, our model is on par with the results, but needs improvement with further hyperparameter tuning.

Figure 4. The VQA task results on [Uncompressed], [(No VQA) $N = 1$], [(VQA-supervised) $N = 1, \tau = 0$], [(No VQA) $N = 132$], and [(VQA-supervised) $N = 132, \tau = 1$] models respectively.

4.3. Qualitative Results

First, we look at the training loss curves for 4 different cases illustrated in Figure 3. For $N = 1, \tau = 0.0$ we see that the total loss decreases with epochs but it oscillates very rapidly, and the same happens for $N = 132, \tau = 1.0$ case, but a much lower training loss is received. However, if we look at the $N = 132, \tau = 0.0$ case, in which the results from 1 were the worst, we can see that the loss never converges and shows the same rapid oscillating behavior in the two previous cases. We suspect this behaviour due to conflicting gradients between the rate loss $\mathcal{L}_{\text{rate}}$ and reconstruction loss $\mathcal{L}_{\text{dist}}$, and the VQA loss $\mathcal{L}_{\text{VQA}}$. This suspicion is given more weight if we look at the case of No VQA supervision at $N = 132$, we can see that the loss steadily decreases relatively smoothly and the spike in between is due to a learning rate change. Therefore, we hypothesize that compression model losses and VQA loss are conflicting with each other.

Now, we take a look at the reconstructed images in Figure 1. As we can see, the models trained with VQA supervision have "trippy" artifacts all over the reconstructed images. In $N = 132, \tau = 0$ case, we see that the model produces no meaningful reconstructions, just random artifacts. This is likely caused due to the domination of VQA supervision which would tune the weights to benefit its task and therefore, the model fails to reconstruct the image.

However, if we look at $N = 132$ with and without VQA supervision, we see very good reconstructions by "No VQA" model, but the model with VQA tries to reconstruct the image meaningfully but also leaves many artifacts even though it has a very low MSE. So it is important to note (as previously pointed out) that low MSE does not necessarily translate to visually acceptable reconstructions. Taking a look at the $N = 1$ case, both "No VQA" and VQA cases produce no meaningful reconstructions due to the size of the latent, however, the model with VQA outperforms its counterpart due to the $\mathcal{L}_{\text{VQA}}$ supervision being more effective on the latent than the reconstruction loss using small latent sizes, as when this is the case, it is difficult to reconstruct the images well in terms of MSE, which seems to correlate with VQA performance the most, but in the extremely lossy regime attaining good MSE values is not an option, so the signal coming from VQA supervision can provide better reconstructions in terms of accuracy.

Finally, we take a look at the VQA task's qualitative performance in Figure 4. The question is asked in the first column with the correct answer indicated next to them. In the following columns, the top-3 answers of the models are dis-

played with the corresponding confidence score. We can see that uncompressed images and (No VQA) $N = 132$ model's images give similar results and generally give better quality answers than other models shown. Also, we can note that these two models have higher confidence in their answers whereas other models are fairly uncertain in their answers. Moreover, the (No VQA) $N = 1$ model is unable to construct colorful images thereby having difficulties when asked about a question related to color, while the model using VQA supervision seems to produce reconstructions in color, although they do not seem to be utilized.

5. Conclusion and Future work

In this work, we have used a lossy compression method [5] and adapted it to an extremely lossy setting to see its effectiveness on the compression quality and on the VQA task. We have analyzed the model with various latent sizes, training with different reconstruction loss weighings, and restricting the supervision from the VQA task. We have seen that for a higher BPP (a higher latent size) the VQA supervision is actually inhibitory to the reconstruction results and also sub-par against "No VQA" supervision. However, if we want extremely lossy compression (e.g. $N = 1$) is required and we care more about the downstream task than the reconstruction, then the downstream task's supervision might be more important.

We have also seen that weighing the reconstruction loss against the VQA loss is to be done very carefully for higher latent size models as they can produce reconstructions that are not even perceivable. Finally, we have seen that the reconstructions of VQA supervised models have "trippy" artifacts all over the images, indicating the misalignment between the reconstruction and VQA losses.

Some future research directions should include ways of removing those artifacts with some back-propagated gradient modification mechanism, which could potentially help us not only get better perceivable reconstructions but also incorporate more knowledge of the downstream task into the compression model. Also, since using the compression in an extremely lossy setting with VQA supervision outperformed its counterparts, it seems promising to explore curriculum learning. Another direction could be using different downstream tasks like visual relationship detection [16], scene graph generation [13] and possibly a multi-task learning pipeline with different downstream tasks.

References

[1] Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Dhruv Batra, and Devi Parikh. Visual question answering dataset v1. visualqa.org, 2016.

[2] Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Dhruv Batra, and Devi Parikh. Vqa: Visual question answering, 2016.

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

[4] J. Ballé, Valero Laparra, and Eero P. Simoncelli. End-to-end optimized image compression. *ArXiv*, abs/1611.01704, 2017.

[5] J. Ballé, David C. Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. *ArXiv*, abs/1802.01436, 2018.

[6] Nabajeet Barman and Maria G. Martini. An evaluation of the next-generation image coding standard avif. In *2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX)*, pages 1–4, 2020.

[7] Nabajeet Barman and Maria G Martini. An evaluation of the next-generation image coding standard avif. In *2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX)*, pages 1–4. IEEE, 2020.

[8] Vivek K Goyal. Theoretical foundations of transform coding. *IEEE Signal Processing Magazine*, 18(5):9–21, 2001.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[10] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[11] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[12] David A Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.

[13] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, D. Shamma, Michael S. Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3668–3678, 2015.

[14] V. Kazemi and A. Elqursh. Show, ask, attend, and answer: A strong baseline for visual question answering. *ArXiv*, abs/1704.03162, 2017.

[15] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014.

[16] Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Visual relationship detection with language priors, 2016.

[17] David Minnen, Johannes Ballé, and George Toderici. Joint autoregressive and hierarchical priors for learned image compression. *arXiv preprint arXiv:1809.02736*, 2018.

[18] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the high efficiency video coding (hevc) standard. *IEEE Transactions on circuits and systems for video technology*, 22(12):1649–1668, 2012.

[19] Gary J Sullivan, Pankaj N Topiwala, and Ajay Luthra. The h. 264/avc advanced video coding standard: Overview and introduction to the fidelity range extensions. In *Applications of Digital Image Processing XXVII*, volume 5558, pages 454–474. International Society for Optics and Photonics, 2004.

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

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[22] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator, 2015.

[23] Gregory K. Wallace. The jpeg still picture compression standard. *Commun. ACM*, 34(4):30–44, Apr. 1991.

[24] Ian H Witten, Radford M Neal, and John G Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540, 1987.