Perceptually Optimizing Deep Image Compression

Li-Heng Chen  
The University of Texas at Austin  
lhchen@utexas.edu

Christos G. Bampis  
Netflix Inc.  
christosb@netflix.com

Zhi Li  
Netflix Inc.  
zli@netflix.com

Andrey Norkin  
Netflix Inc.  
anorkin@netflix.com

Alan C. Bovik  
The University of Texas at Austin  
bovik@ece.utexas.edu

ABSTRACT
Mean squared error (MSE) and $\ell_p$ norms have largely dominated the measurement of loss in neural networks due to their simplicity and analytical properties. However, when used to assess visual information loss, these simple norms are not highly consistent with human perception. Here, we propose a different proxy approach to optimize image analysis networks against quantitative perceptual models. Specifically, we construct a proxy network, which mimics the perceptual model while serving as a loss layer of the network. We experimentally demonstrate how this optimization framework can be applied to train an end-to-end optimized image compression network. By building on top of a modern deep image compression model, we are able to demonstrate an averaged bitrate reduction of 28.7% over MSE optimization, given a specified perceptual quality (VMAF) level.

KEYWORDS
convolutional neural networks; deep image compression; perceptual optimization; perceptual image quality

Li-Heng Chen, Christos G. Bampis, Zhi Li, Andrey Norkin, and Alan C. Bovik. 2020. Perceptually Optimizing Deep Image Compression.

1 INTRODUCTION
Deep neural networks have made rapid advances on diverse multimedia tasks [6, 28, 33], especially the image transformation problems including denoising [9], super-resolution [24], frame interpolation [29], and so on. Specifically speaking, a generative network is learned to reconstruct high-quality output images from degraded input images under a supervised manner. A loss function is defined to measure the fidelity between the output and a ground-truth image. For example, the denoising task aims to reconstruct a noise-free image from a noisy image, and Convolutional Neural Networks (CNNs) have been shown to provide good noisy-to-pristine mapping functions. However, despite the tremendous amount of research being applied on deep learning image transformation problems, the loss functions used to guide model training has been underexamined and largely limited to the $\ell_p$ norm family. The structural similarity quality index (SSIM) [41] has also been adopted as loss functions for several image reconstruction tasks [39, 49], owing to their perceptual relevance and good analytic properties, such as differentiability.

As a long-standing research problem, predicting picture quality with high-quality reference pictures has achieved remarkable success. Numerous full-reference (FR) perceptual models have been proposed and proven to surpass MSE-based measurements. Examples include other SSIM-type methods [34, 42, 43], VIF [36], VSNR [10], MAD [25], FSIM [47], and VSI [46]. Moreover, learning based quality predictors such as Video Multimethod Assessment Fusion (VMAF) [27], a successful open-sourced example developed by Netflix, has been powerful tools to optimize tremendous volumes of internet video traffic. Unfortunately, most of the advanced, high-performance image quality indexes have never been adopted as loss functions for end-to-end optimization networks, because they are generally non-differentiable and functionally complex.

Recently, lossy image compression models have been realized using deep neural network architectures. This may be viewed as a special case of generative networks, where the input image is equal to the ground-truth image. Unlike the conventional image codecs standards, which rely on "handcrafted" functional blocks such as transform matrix or in-loop filters, the parameters of learned image compression are optimized in an end-to-end manner. Most of these have employed deep auto-encoders. For example, Ballé et al. [4] proposed a general infrastructure for optimizing image compression where bitrate is estimated and considered during training. In [5], this model is improved by incorporating a network, scale hyperprior, into the compression framework. The authors use the additional network to estimate the standard deviation of the quantized coefficients to further improve coding efficiency. Later, Minnen
Adversarial example. \[ \text{trained image classification network have the capability to capture} \]

experimental studies suggest that the features extracted from a well-

\[ \text{objective evaluation against several other perceptual models} \]

\[ \text{results, as validated by conducting a human subjective study} \]

optimization. Two recent studies adopted structural similarity func-

\[ \text{their convexity properties} \]

SSIM have been widely adopted because of the simple analytical

\[ \text{measurements made by a perceptual image quality model using a} \]

\[ \text{in deep image compression network. In order to address the afore-

\[ \text{domain databases are generally quite different from distortions} \]

\[ \text{the conventional distortion types in public} \]

\[ \text{network may produce a poor interpolation result. Additionally, as} \]

\[ \text{of the network to be an interpolation problem, whereby the neural} \]

\[ \text{true VMAF score and the proxy VMAF score predicted by the network} \]

\[ \text{proxy network was trained to mimic the VMAF algorithm. However,} \]

\[ \text{an “adversarial” example generated by the deep compression net-

\[ \text{its loss layer is the output of a} \]

\[ \text{deep compression network often generates “adversarial” examples when} \]

\[ \text{the perceptual loss is defined as} \]

\[ \mathcal{L}_{\text{perceptual}} = \sum_{i} \frac{1}{N_i} |\phi_i(x) - \phi_i(\hat{x})|^2, \]  

where \( \phi_i \) denotes the output feature map of the \( t \)-th layer with \( N_i \) elements of a pre-trained network \( \phi \). In practice, the loss computed from the high-level features extracted from a pre-trained VGG classification network [37], also called VGG loss, has been commonly adopted for diverse computer vision tasks. The VGG loss has been applied to such diverse tasks as style transfer [17, 20], superresolution [7, 20, 26, 35], and image inpainting [44]. Despite its ubiquity, this “unreasonable” perceptual loss is notorious for creating unpleasant artifacts [20]. Most importantly, this edge-sharpening loss function is incapable of optimizing a network toward a specific quality model.

3 PROPOSED PERCEPTUAL OPTIMIZATION FRAMEWORK

Learning a successful CNN model depends highly on the size of the training set. Luckily, training a proxy network on an existing model does not require human-labeled subjective quality scores such as mean opinion scores (MOS), which is often the greatest obstacle to learning DNN-based IQA models [18, 22, 45]. Ground truth scores for training the proxy network are easily obtained, given the availability of pristine and distorted patches. To start with, we created a simple network trained by existing datasets [31] comprising numerous reference and distorted patches. To start with, we created a simple network trained by existing datasets [31] comprising numerous reference and distorted patches. Also, the corresponding metric scores were calculated as the ground-truth for training. The proxy network is first learned to predict the metric score given a pristine patch and a distorted patch. Next, the trained proxy network is inserted into the loss layer of the deep compression network with the goal of maximizing the proxy score. Unfortunately, severe complication can arise when applying this straightforward methodology.

We discovered that the deep compression network often generates “adversarial” examples when its loss layer is the output of a pre-trained network having fixed parameters. Figure 2 shows such an “adversarial” example generated by the deep compression network using a proxy network as its loss function. In this example, the proxy network was trained to mimic the VMAF algorithm. However, comparing Fig. 2(a) with Fig. 2(b), it is apparent that the true VMAF score and the proxy VMAF score predicted by the network \( f_p \) are very different. This can be understood by considering the training of the network to be an interpolation problem, whereby the neural networks maps a test image to an accurate quality score. However, when the input is too different from the training set, the proxy network may produce a poor interpolation result. Additionally, as pointed out in [12], the conventional distortion types in public domain databases are generally quite different from distortions created by a deep neural networks. In this regard, training a proxy network on previously created databases might be suboptimal for this problem.

3.1 Alternating Learning Framework

In order to tackle the aspect, our approach to training an image compression model in a perceptually optimized way is depicted
in Fig. 3. The idea is to simply use the adversarial examples along with their objective quality scores as additional training data of the proxy network. The proxy network is then updated, enabling it to predict proxy quality much more accurately. This framework involves optimizing an image compression network \( f_c \), and a proxy network of an IQA model \( f_p \). In each training iteration, the two networks are alternately updated as follows:

**Deep Compression Network.** To integrate the proxy network \( f_p \) into the update of \( f_c \) given a mini-batch \( x \), the model parameters of \( f_p \) are fixed during training. In order to minimize perceptual distortion, the output of \( f_p \) becomes part of the objective in the optimization of \( f_c \):

\[
 f_p(x, \hat{x}) = f_p(x, f_c(x)) .
\]  

By back-propagating through the forward model, the loss derivative is used to drive \( f_c \).

**Proxy IQA Network.** Given a mini-batch pair \( x \) and \( \hat{x} \) collected from the most recent update of the compression network, the quality scores \( M(x, \hat{x}) \) are calculated. The network \( f_p \) is updated to optimally fit \( M \) given the input \( (x, \hat{x}) \). Note that the compression network \( f_c \) is not needed in this part of the training. As may be seen, \( f_p \) is incorporated into the training of the compression network. However, it is important to understand that it is not present during the inference phase.

By applying the proposed alternating training, the proxy network is capable of spontaneously adapting to newly generated adversarial patches. In addition, exotic artifacts created by deep image compression can be "seen" by the proxy IQA network: the patches reconstructed by the compression network are directly used to update the proxy network, hence the aforementioned problem becomes immediately resolved.

### 3.2 Network Architecture

The proxy IQA network \( f_p \) takes a reference patch \( x \) and a distorted patch \( \hat{x} \) as input, where both have \( W \times H \) pixels. They are then concatenated into a 6-channel signal, where a \( W \times H \times 6 \) raw input \( \{x, \hat{x}\} \) is fed into the network and reduced to a predicted quality score. As depicted in Fig. 3, the network \( f_p \) may be as simple as a shallow CNN consisting of three stages of convolution, ReLU nonlinearity, and subsampling. The spatial size is reduced by a factor of 2 after each stage via \( 2 \times 2 \) max pooling layers. The size of convolution kernels are fixed to \( 5 \times 5 \) for all stages, while the number of filters at the first stage is set to 16 and is increased by a factor of 2 for each subsequent stage. Finally, 64 feature maps with size \( W/16 \times H/16 \) are flattened and fed to a fully connected layer which yields the output. The parameterization of each layer is detailed in the figure.

The image compression network comprises an analysis transform \( (g_a) \) at the encoder side, and a synthesis transform \( (g_s) \) at the decoder side. Both transform units are implemented as consecutive layers of convolution-down(up) sampling-activation. Instead of the commonly used ReLU, a generalized divisive normalization (GDN) transform is adopted as the activation function [3]. It is similar to the local gain control behavior in human visual system, where visual signals are normalized by their rectified neighbors. Lastly, the functional block "Q/EC" denotes quantization and entropy coding. In this work, we build on two different deep image compression models [4, 5].

### 3.3 Loss Functions

As illustrated in Fig. 3, let \( x, y, \hat{y}, \) and \( \hat{x} \) be the source batch, latent presentation, quantized latent presentation, and reconstructed
where has a similar notion as rate-distortion optimization (RDO) in Table 1: Comparison of conventional codecs and optimized deep image compression: average change of BD-rate expressed as percentage, using three different IQA models to train the compression network. The baseline of comparison is the MSE-optimized BLS model [4]. Smaller or negative values indicate better coding efficiency.

| Image Dataset | Kodak | Tecnick | NFLX Billboard |
|---------------|-------|---------|----------------|
|                | PSNR  | SSIM    | VMAF           | PSNR  | SSIM    | VMAF           | PSNR  | SSIM    | VMAF           |
| JPEG          | 113.99| 129.49  | 149.86 | 78.36 | 119.33 | 218.04 | 171.59 | 89.73 | 102.28 | 143.99 | 168.20 | 89.95 |
| JPEG2000      | -11.51| 6.25    | -1.02  | -33.39| -13.06 | -1.55  | -8.41  | -34.25| -27.81 | -1.43  | -3.93  | -38.98 |
| HEVC_444      | -26.35| -6.32   | -6.12  | -28.23| -28.32 | -8.97  | -11.07 | -27.65| -49.43 | -17.12 | -16.06 | -35.03 |
| HEVC_420      | -27.33| -25.98  | -24.97 | -42.18| -19.48 | -28.97 | -33.95 | -46.67| -37.63 | -35.41 | -33.88 | -50.91 |
| BLS SSIM_p    | 15.89 | -21.31  | -19.25 | 7.19  | 8.67   | -10.79 | 16.11  | 8.68  | 16.79  | -19.01 | -17.73 | 9.75   |
| BLS MSIM_p    | 11.67 | -11.58  | -21.77 | -0.17 | 4.47   | -17.40 | -23.50 | 0.19  | 12.28  | -11.59 | -23.53 | 4.34   |
| BLS VMAF_p    | 5.23  | -6.53   | -7.78  | -23.35| 6.23   | -8.45  | -5.97  | -23.78| 7.00   | -4.35  | -5.43  | -21.97 |
| BM3HJ MSE [5] | -21.46| -10.94  | -10.17 | -25.78| -26.03 | -20.22 | -16.71 | -33.75| -36.64 | -21.21 | -21.08 | -38.01 |
| BM3HJ VMAF_p  | -15.90| -13.57  | -13.17 | -47.11| -19.64 | -23.14 | -16.73 | -53.18| -29.96 | -18.87 | -19.29 | -56.06 |

batch, respectively. The model parameters in the analysis and synthesis transforms are collectively denoted by \( \theta = (\theta_\theta, \theta_\phi) \). The proxy network \( f_p \) has model parameters \( \phi \). Given a perceptual metric \( M \), the goal is to optimize the full set of parameters \( \theta, \phi \), such that the learned image codec can generate a reconstructed image \( \hat{x} \) that has a high perceptual quality score \( M(x, \hat{x}) \). Furthermore, the rate should be as small as possible.

Generally, learned image compression network is optimized by minimizing the objective function defined by

\[
L(\theta) = \lambda L_d(\theta) + L_r, \tag{3}
\]

which has a similar notion as rate-distortion optimization (RDO) in conventional codecs. Under this scheme, \( L_d \) is the residual between the source patch and the reconstructed patch mapped by \( d(.) \)

\[
L_d(\theta) = d(x - \hat{x}), \tag{4}
\]

where \( d(.) \) is a distance function such as mean square error or mean absolute error. On the other hand, \( L_r \) is the rate loss representing the bit consumption of an encode \( \hat{y} \). We followed the original work in [4], where the rate loss is defined by

\[
L_r(\theta) = -\log_2 p(y). \tag{5}
\]

The term \( p(y) \) denotes the entropy model. This entropy term is minimized when the actual marginal distribution and \( \hat{y} \) have the same distribution. During training, the latent presentation \( y \) is quantized to \( \hat{y} \) by adding i.i.d uniform noise \( U \left( -\frac{1}{2}, \frac{1}{2} \right) \). Then, \( \hat{y} \) is used to estimate the rate via (5). Unlike the training phase, normal rounding-based quantization is applied to quantize \( y \). Then, entropy coders such as variable length coding or arithmetic coding can be used to losslessly encode the discrete-valued data into the bitstream during the inference.

Rather than just minimizing an \( \ell_p \) norm between \( x \) and \( \hat{x} \), we introduce a loss term \( L_p \). This proxy loss \( L_p \) is defined to maximize the output of proxy network \( f_p \), denoted by \( M \), with fixed network parameters \( \phi \):

\[
L_p(\theta; \phi) = M_{\text{max}}(x, \hat{x}). \tag{6}
\]

Here \( M_{\text{max}} \) denotes the upper bound of the model \( M \), which is a constant to the loss function. Finally, the total loss function for optimizing the compression network is the weighted combination of the losses:

\[
L(\theta; \phi) = \lambda \left[ \alpha L_p + (1 - \alpha) L_d \right] + L_r, \tag{7}
\]

where \( \lambda \) controls the trade-off between bitrate and distortion of the encoded bitstream, and \( \alpha \) weights \( L_p \) against \( L_d \). Here, the term \( L_d \) plays a different role as a regularization term. Since the proxy network is updated at each step, the loss function \( L_p \) is also changed. The pixel loss serves to stabilize the training process. In our model, we empirically set \( \alpha = 1.54e^{-3} \) and \( M_{\text{max}} = 100 \) when optimizing for VMAF.

The proxy network \( f_p \) aims to mimic an image quality model \( M \). While updating \( f_p \), we define a metric loss \( L_m \) to attain this objective given two image batches \( x \) and \( \hat{x} \):

\[
L_m(\phi; \hat{x}) = \left[ M(x, \hat{x}) - M(x, \hat{x}) \right]^2. \tag{8}
\]

Note that \( \hat{x} \) is a constant, since it is obtained from the reconstructed patches generated during the most recent update of the compression network.

4 EXPERIMENTS

The following subsections thoroughly describe the experimental setup. We also present the quantitative evaluation and subjective comparison.

4.1 Experimental Setup

Implementation Details. We used the TensorFlow framework (version 1.12) to implement the proposed method. The Adam solver [23] were used to optimize both the proxy network and the deep compression network, with parameters \( (\beta_1, \beta_2) = (0.9, 0.999) \) and a weight decay of 0.01. We set the initial learning rates for both \( f_p \) and \( M \) aims to mimic an image quality model \( M \). With \( L_p \), we define a metric loss \( L_m \) to attain this objective given two image patches \( x \) and \( \hat{x} \):

\[
L_m(\phi; \hat{x}) = \left[ M(x, \hat{x}) - M(x, \hat{x}) \right]^2. \tag{8}
\]

Note that \( \hat{x} \) is a constant, since it is obtained from the reconstructed patches generated during the most recent update of the compression network.
We also used a subset of the Tecnick dataset [2] containing 768 × 128 × k kodim10 compression algorithms on the image Figure 4: Rate-Distortion (RD) curves for different image problems. test images were included in the training sets, to avoid overfitting resolutions and contents. It should be noted that none of these images of resolution 1200 used to evaluate image compression algorithms and IQA models. We utilized the Kodak dataset of 24 very high quality uncompressed 768 × 512 images. This publicly available image set is commonly used to evaluate image compression algorithms and IQA models. We also used a subset of the Tecnick dataset [2] containing 100 images of resolution 1200 × 1200, and 223 billboard images collected from the Netflix library [38], yielding images having more diverse resolutions and contents. It should be noted that none of these test images were included in the training sets, to avoid overfitting problems.

**Evaluation Setup.** As is the common practice in the field of video coding, we measured the objective coding efficiency of each image codec using the Bjøntegaard-Delta bitrate (BD-rate) [16], which quantifies average differences in bitrate at the same distortion level relative to another reference encoder. To calculate BD-rate, we encoded the images at eight different bitrates, ranging from 0.05 bpp (bit per pixel) to 2 bpp. In all the experiments conducted, we denote the image compression model [4] as the BLS model. The performances of all of the codecs were compared to the same baseline – the MSE-optimized BLS model. A negative number of BD-rate means the bitrate was reduced as compared with the baseline. The input image formats used were YUV444 for JPEG and JPEG2000, and both YUV420/444 for intra-coded HEVC, respectively. Lastly, the distortion levels that were used for BD-rate calculation were quantified using PSNR, SSIM, MS-SSIM (also represented by MSIM in the table), and VMAF.

### Table 1 tabulates the benchmark study on the aforementioned three datasets. Each cell shows the BD-rate relative to the BLS baseline, with respect to different quality models. We denote an optimized compression model for a given IQA model M using (7) and (8) by $M^p$. In addition to the BLS model, we also deployed the proposed VMAF$_p$ optimization framework on a more sophisticated deep compression model [5] (BMSHJ) to test its generality. We report the BD-rate changes obtained, averaged over all the images in each dataset. These results show that our optimization approach is able to successfully optimize a deep image compression model over different IQA algorithms. Indeed, significant BD-rate reductions were obtained in many cases. An interesting observation can be made that, unlike using other IQA models used as targets of the proposed optimization, VMAF$_p$ optimization delivers coding gain with respect to all of the BD-rate measurements, except the PSNR BD-rate. This suggests VMAF being a good optimization target.

As a basic test, we subjectively compare results yielding similar bitrates but different objective quality scores. Figure 5 shows a visual comparison under extreme compression (around 0.05 bpp). Obviously, the VMAF$_p$-optimized model significantly outperformed the MSE-optimized baseline model, delivering performance comparable to HEVC and JPEG2000 with respect to VMAF score and subjective quality. We also plot the corresponding VMAF Rate-distortion (RD) curve, a common tool for comparing different encoders, in Fig. 4. We observe that the proposed optimization scheme generally leads to a compression gain in VMAF. At high bitrates, the VMAF$_p$-optimized model yielded comparable VMAF scores as the baseline MSE-optimized model, while consuming significant fewer bits. In this particular example, roughly 16% of the bits can be reduced without suffering perceptual quality.

#### Figure 4: Rate-Distortion (RD) curves for different image compression algorithms on the image kodim10, measured with VMAF.

**Source** | **BLS-MSE** | **JPEG2000** | **BLS-VMAF$_p$** | **HEVC-444**
--- | --- | --- | --- | ---

| bpp | VMAF | PSNR / SSIM | VMAF | PSNR / SSIM |
|-----|-----|------------|-----|------------|
| 0.052 | 27.82 | 26.12 / 0.817 | 0.050 | 36.39 / 0.806 |
| 0.058 | 37.06 / 0.840 | 0.054 | 39.67 / 0.843 |

**Figure 5: Visualization of decoded pictures (kodim10) from different compression models. Each cropped to 128 × 128 patch for display purpose.**

**Training Setup.** We used a subset of the 6507 processed images from the ImageNet database [15] as training data. As described in [4], small amounts of uniform noise were added to the images. The images were then down-sampled by random factors to reduce compression artifacts and high-frequency noise, and randomly cropped to a size of 256 × 256. In each mini-batch, we randomly sampled 8 image patches from the subset. We then cropped the images to 128 × 128 patches resulting in an 8 × 128 × 128 × 3 tensor.

**Evaluation Datasets.** To evaluate various image codecs, we used a subset of the 6507 processed images from the ImageNet database [15] as training data. As described in [4], small amounts of uniform noise were added to the images. The images were then down-sampled by random factors to reduce compression artifacts and high-frequency noise, and randomly cropped to a size of 256 × 256. In each mini-batch, we randomly sampled 8 image patches from the subset. We then cropped the images to 128 × 128 patches resulting in an 8 × 128 × 128 × 3 tensor.

**Evaluation Datasets.** To evaluate various image codecs, we used a subset of the 6507 processed images from the ImageNet database [15] as training data. As described in [4], small amounts of uniform noise were added to the images. The images were then down-sampled by random factors to reduce compression artifacts and high-frequency noise, and randomly cropped to a size of 256 × 256. In each mini-batch, we randomly sampled 8 image patches from the subset. We then cropped the images to 128 × 128 patches resulting in an 8 × 128 × 128 × 3 tensor.

**Evaluation Datasets.** To evaluate various image codecs, we used a subset of the 6507 processed images from the ImageNet database [15] as training data. As described in [4], small amounts of uniform noise were added to the images. The images were then down-sampled by random factors to reduce compression artifacts and high-frequency noise, and randomly cropped to a size of 256 × 256. In each mini-batch, we randomly sampled 8 image patches from the subset. We then cropped the images to 128 × 128 patches resulting in an 8 × 128 × 128 × 3 tensor.
To further understand the behavior of the proposed alternating training, we compared true VMAF scores and proxy VMAF scores (predicted by the proxy network) of the reconstructed batch. The two scores are plotted in mean values (lines) and one standard deviations (shadows).

4.3 Study of Alternating Training

To further understand the behavior of the proposed alternating training, we compared true VMAF scores against proxy VMAF scores in Fig. 6. All of the scores were calculated on the reconstructed patches produced during training. Figure 6(a) shows that the proxy VMAF scores quickly approached 100, whereas much lower true VMAF scores were assigned to the patches produced by the compression model. This directly reflects the problem we have mentioned (Sec. 3) when a pre-trained proxy network is applied. However, when the reconstructed patches were fed into the proxy network along with their objective quality scores, the proxy network is updated straightaway to predict proxy quality much more accurately. As shown in Fig. 6(b), the true and proxy scores become highly consistent early in the training process.

4.4 Computational Cost

It is critical for a learned image compression model to have comparable execution time to other codecs. We compiled the source code of standard codecs, in order to be able to compare them on the same computer with a 2.10GHz CPU and 4 GTX-1080TI GPUs. The results were then calculated by averaging the runtime over all 24 Kodak images under different bitrate settings. The encoding and decoding times of the various compared codecs are reported in Table 2. It may be observed that the time complexity of the MSE-optimized and VMAFp-optimized models are nearly identical, as they deploy the same network architecture in application. Overall, the runtime of learned image compression models are acceptable and can be further reduced if performed on a GPU. It is worth noting that the BLS models achieves the fastest decoding speed when a GPU is available. It should also be noted that the decoding time of HEVC was estimated from the reference software HM16.9, which might be slow.

Table 2: Average processing speed in milliseconds for different compression models. Model loading time for deep compression is excluded.

| Compression model | Encode time (ms) | Decode time (ms) |
|-------------------|-----------------|-----------------|
| JPEG              | CPU 43.02       | Decode time    |
| JPEG2000          | CPU 10.80       | HEVC CPU 4578.57 | 89.88 |
| BLS MSE           | CPU 251.01      | GPU 231.62      |
| BLS VMAFp         | CPU 246.57      | GPU 119.02      |
| BMSHJ MSE         | CPU 351.23      | GPU 378.68      |
| BMSHJ VMAFp       | CPU 367.71      | GPU 380.04      |
| BMSHJ GPU 308.53  | 341.96          |

5 CONCLUSION

In this work, we focus on designing the loss function for deep image compression. In particular, we have presented a framework for perceptually optimizing a generative network and a proxy image quality assessment network. When integrated into deep image compression models, our method allows end-to-end training and can provide compression gains with respect to different IQA metrics. We believe that the idea behind the proposed training framework is general. With proper modifications of the framework parameters or the architecture of the proxy network, the approach has the potential to improve on a wide variety of image restoration problems with weak MSE based ways of optimization.

ACKNOWLEDGMENTS

This research is supported by Netflix. The authors thank the Texas Advanced Computing Center (TACC) at The University of Texas at Austin for providing HPC resources that have contributed to the research results reported within this paper. URL: http://www.tacc.utexas.edu

REFERENCES

[1] E. Agustsson, M. Tschannen, F. Mentzer, R. Timofte, and L.V. Gool. 2019. Generative adversarial networks for extreme learned image compression. In Proc. IEEE Int. Conf. Comput. Vision. 221–231.
[2] N. Asuni and A. Giachetti. 2014. TESTIMAGES: A Large-scale Archive for Testing Visual Devices and Basic Image Processing Algorithms. In Proc. Eurographics Italian Chapter Conference. 63–70.
[3] J. Ballé. 2018. Efficient Nonlinear Transforms for Lossy Image Compression. In Proc. IEEE Picture Coding Symp. 248–252.
[4] J. Ballé, V. Laparra, and E.P. Simoncelli. 2017. End-to-end Optimized Image Compression. In Proc. Int. Conf. Learn. Represent. 1–27.
[5] J. Ballé, D. Minnen, S. Singh, S.J. Hwang, and N. Johnston. 2018. Variational image compression with a scale hyperprior. In Proc. Int. Conf. Learn. Represent. 1–23.
[6] S. Bose, D. Maniry, K.-R. Muller, T. Wiegand, and W. Samek. 2018. Deep Neural Networks for No-Reference and Full-Reference Image Quality Assessment. IEEE Trans. Image Processing 27, 1 (2018), 206–219.
Perceptually Optimizing Deep Image Compression

[7] J. Bruna, P. Sprechmann, and Y. LeCun. 2016. Super-Resolution with Deep Convolutional Sufficient Statistics. In Proc. Int. Conf. Learn. Represent. 1–17.

[8] D. Brunet, E.R. Vescay, and Z. Wang. 2012. On the Mathematical Properties of the Structural Similarity Index. IEEE Trans. Image Processing 21, 4 (2012), 1488–1499.

[9] H.C. Burger, C.J. Schuler, and S. Harmeling. 2012. Image denoising: Can plain neural networks compete with BM3D? In Proc. IEEE Conf. Comput. Vision Pattern Recog. 2392–2399.

[10] D.M. Chandler and S.S. Hemami. 2007. VSNR: A Wavelet-Based Visual Signal-to-Noise Ratio for Natural Images. IEEE Trans. Image Processing 16, 9 (2007), 2284–2298.

[11] S.S. Chananappaya, A.C. Bovik, and R.W. Heath. 2008. Rate Bounds on SSIM Index of Quantized Images. IEEE Trans. Image Processing 17, 9 (2008), 1624–1639.

[12] Z. Cheng, P. Akyazi, H. Sun, J. Katto, and T. Ebrahimi. 2019. Perceptual Quality Study on Deep Learning Based Image Compression. In Proc. IEEE Int. Conf. Image Proc. IEEE.

[13] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto. 2019. Energy Compaction-Based Image Compression Using Convolutional AutoEncoder. In IEEE Transactions on Multimedia (2019), 1–1.

[14] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto. 2019. Learning Image and Video Compression through Spatial-Temporal Energy Compaction. In Proc. IEEE Conf. Comput. Vision Pattern Recog.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and F.-F. Li. 2009. ImageNet: A large-scale hierarchical image database. In Proc. IEEE Conf. Comput. Vision Pattern Recog. 248–255.

[16] G. Bjøntegaard. 2001. Calculation of average PSNR differences between RD-curves. document VCEG-M33, ITU-T Video Coding Experts Group (VCEG) Thirteenth Meeting (Austin, TX, April 2001).

[17] L.A. Gatys, A.S. Ecker, M. Bethge, A. Hertzmann, and E. Shechtman. 2017. Controlling Perceptual Features in Neural Style Transfer. In Proc. IEEE Conf. Comput. Vision Pattern Recog. 3736–3738.

[18] D. Ghadiyaram and A.C. Bovik. 2016. Massive Online Crowdsourced Study of Subjective and Objective Picture Quality. IEEE Trans. Image Processing 25, 1 (2016), 372–387.

[19] Y.-H. Huang, T.-S. Ou, P.-Y. Su, and H.H. Chen. 2010. Perceptual Rate-Distortion Optimization Using Structural Similarity Index as Quality Metric. IEEE Trans. Circuits Syst. Video Technol. 20, 11 (2010), 1614–1624.

[20] J. Johnson, A. Alahi, and F.-F. Li. 2016. Perceptual Losses for Real-Time Style Transfer and Super-Resolution. In Proc. Eur. Conf. Comput. Vision. Vol. 9906. 694–711.

[21] N. Johnston, D. Vincent, D. Minnen, S. Singh, T. Chinen, S.J. Hwang, J. Johnson, D. Vincent, D. Minnen, M. Covell, S. Singh, T. Chinen, S.J. Hwang, J. Keating, and A. Tejani. 2017. Photo-Realistic Single Image Super-Resolution With Deep Laplacian Pyramid Networks. In Proc. Int. Conf. Learn. Represent.

[22] K.-S. Lu, A. Ortega, D. Mukherjee, and Y. Chen. 2020. Perceptually Inspired Solutions to Data-Driven Image Quality Assessment. In Proc. IEEE Int. Conf. Image Proc.

[23] Z. Li, C.G. Bampis, J. Novak, A. Aaron, K. Swanson, A. Moorthy, and J.D. Cock. 2019. From Patches to Pictures (PaQ-2-PiQ): Mapping the Perceptual Space of Picture Quality To Spectral Space. In Proc. IEEE Int. Conf. Image Proc.

[24] L. Zhang, L. Zhang, X. Mou, and D. Zhang. 2011. FSIM: A Feature Similarity Index for Image Quality Assessment. IEEE Signal Process. Mag. 33, 6 (2017), 139–141.

[25] D.P. Kingma and J. Ba. 2015. Adam: A method for stochastic optimization. In Proc. Int. Conf. Learn. Represent. 1–15.

[26] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang. 2019. Fast and Accurate Image Inpainting Using Multi-scale Neural Patch Synthesis. In Proc. IEEE Conf. Comput. Vision Pattern Recog., Vol. 9906. 4385–4393.

[27] X. Ying, H. Niu, P. Gupta, D. Mahajan, D. Ghadiyaram, and A.C. Bovik. 2020. From Patches to Pictures (PaQ-2-PiQ): Mapping the Perceptual Space of Image Quality. In Proc. IEEE Conf. Comput. Vision Pattern Recog. 23, 10 (2014), 4270–4281.

[28] Z. Li, Y.-L. Liu, Y.-Y. Lin, and Y.-Y. Chuang. 2019. Deep Video Frame Interpolation using Cyclic Frame Generation. In Proc. AAAI. 8794–8802.

[29] K.-S. Lu, A. Ortega, D. Mukherjee, and Y. Chen. 2020. Perceptually inspired weighted MSE optimization using irregularity-aware graph Fourier transform. arXiv:2002.08558

[30] K. Ma, Z. Duannu, Q. Wu, Z. Wang, H. Yong, H. Li, and L. Zhang. 2017. Waterloo Exploration Database: New Challenges for Image Quality Assessment Models. IEEE Trans. Image Processing 26, 2 (2017), 1004–1016.

[31] D. Minnen, J. Ballé, and G.D. Toderici. 2018. Joint Autoregressive and Hierarchical Priors for Learned Image Compression. In Advances in Neural Information Processing Systems 31: 10771–10780.