Task-Aware Quantization Network for JPEG Image Compression

Jinyoung Choi[1] and Bohyung Han[1]

Dept. of ECE & ASRI, Seoul National University, Korea
{jin0.choi,bhhan}@snu.ac.kr

Abstract. We propose to learn a deep neural network for JPEG image compression, which predicts image-specific optimized quantization tables fully compatible with the standard JPEG encoder and decoder. Moreover, our approach provides the capability to learn task-specific quantization tables in a principled way by adjusting the objective function of the network. The main challenge to realize this idea is that there exist non-differentiable components in the encoder such as run-length encoding and Huffman coding and it is not straightforward to predict the probability distribution of the quantized image representations. We address these issues by learning a differentiable loss function that approximates bitrates using simple network blocks—two MLPs and an LSTM. We evaluate the proposed algorithm using multiple task-specific losses—two for semantic image understanding and another two for conventional image compression—and demonstrate the effectiveness of our approach to the individual tasks.

Keywords: JPEG image compression, adaptive quantization, bitrate approximation.

1 Introduction

Image compression is a classical task to reduce the file size of an input image while minimizing the loss of visual quality. This task has two categories—lossy and lossless compression. Lossless compression algorithms preserve the contents of input images perfectly even after compression, but their compression rates are typically low. On the other hand, lossy compression techniques allow the degradation of the original images by quantization and reduce the file size significantly compared to lossless counterparts. Note that the rate-distortion trade-off [6] in lossy compression characterizes the relationship between image file size and its visual quality.

JPEG [30] is the most widely accepted image compression standard. The encoder and decoder of JPEG have several building blocks for the subtasks including transform coding, chroma subsampling, quantization and entropy coding, where each component provides options for further optimization and customization. JPEG2000 [34] and BPG have been proposed after JPEG to improve the performance of image compression, yet they are not as accepted universally as

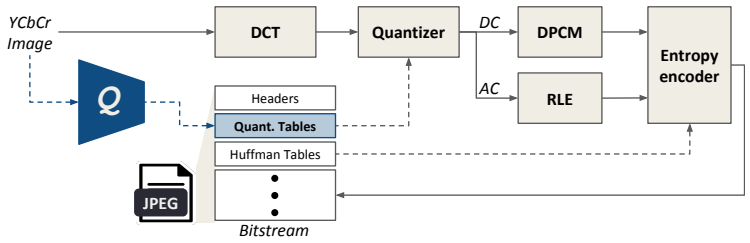

Fig. 1. The application example of the proposed approach incorporated into general JPEG encoding pipeline. Every componts remain the same apart from that the recommendable quantization table is inferred by the pretrained quantization network (Q) before encoded. This quantization table can be specified as coding parameters of JPEG encoder and reused at decoding process from the header file.

JPEG that benefits from high compatibility and low complexity. Despite the popularity of JPEG for decades, its standard configuration is rather suboptimal.

Recently, there have been huge advances in learned image compression using deep neural networks. One of the most common approaches is to use autoencoders, where the intermediate bottleneck representation is used as the compressed code for an input image. Some approaches have reached to outperform the traditional compression algorithms by introducing novel deep neural network architectures with non-linear operations [1, 4, 5, 17, 21, 23, 26, 28, 36, 37]. Although learned image compression algorithms are more flexible to integrate diverse objective functions to improve image quality and extend to the images in other domains, their applicability is still limited because they require dedicated decoders trained with encoders jointly and high-performance hardware.

To address the limitations, we propose a novel framework for JPEG image compression based on deep neural networks. Fig. 1 illustrates the JPEG image compression pipeline accompanying with the proposed quantization network. Our method incorporates a learning-based approach into the JPEG compression standard and estimates the data-driven quantization tables perfectly compatible with the off-the-shelf JPEG encoder and decoder. Moreover, contrary to the conventional image compression methods that focus only on low-level image quality measures, our framework also allows to optimize performance of arbitrary high-level image understanding tasks, *e.g.*, image classification and image captioning, by introducing proper loss functions. As the proposed framework requires bitrate loss calculations which involve non-differentiable coding algorithms, we instead devise a deep neural network model that directly estimates the code length given JPEG DCT representations.

The main contributions of the proposed approach are summarized as follows:

- We propose a novel task-aware JPEG image compression framework based on deep neural networks, which optimizes for performance of high-level target tasks, *e.g.*, image recognition, in addition to visual quality of a decoded image.

- The proposed approach learns a quantization network to estimate image-specific quantization tables fully compatible with the standard JPEG Codec.
- To simulate the JPEG's coding algorithm, we design a code length prediction network of JPEG DCT representations, which empowers differentiable learning and accurate inference on a bitrate.

The rest of the paper is organized as follows. Section 2 and 3 briefly discuss the related work and the background of JPEG, respectively. The proposed approach is presented in Section 4. Section 5 describes the modeling of bitrate prediction in detail. The experimental results with analysis are in Section 6.

2 Related Work

The optimal quantization table may differ image by image and also depend on bitrate constraints and image quality standards. Since finding an explicit solution of the optimal quantization table under the exact distribution of images is infeasible, various techniques with different error metrics have been studied to find the optimal quantization table.

A rate-distortion (RD) optimization method based on an MSE measure uses the statistics of DCT coefficients to compute bitrate and distortion for each frequency [31] and identifies the optimal quantization table. While the distortion metric based on MSE makes the algorithm efficient, the models driven by human visual system (HVS) [3, 16, 41] often show empirically better performance. Note that the default JPEG quantization tables are also derived by exploiting HVS properties in an image-agnostic manner. Heuristics such as genetic algorithm and simulated annealing are also employed to search for improved quantization tables in [13, 29]. Especially, Hopkins *et al.* [13] propose new baseline tables for a subset of chosen quality factors by perturbing standard tables via simulated annealing with respect to FSIM [43] measure.

Recently, the errors given by deep neural networks have arisen as new metrics for image quality assessment. Dodge and Karam [7] studied how image quality affects classification accuracy, where JPEG compression is employed as an image quality degradation factor. Liu *et al.* [24] suggested a heuristic image compression framework that optimizes performance of a target task, by analyzing the importance of each frequency and the statistics for better quantization.

3 JPEG Compression

This section presents the standard procedure in the JPEG encoder and decoder. Before processing described in Section 3.1, color components of an image are transformed to YCbCr color space and chrominace channels are optionally subsampled. Then the image is split into 8×8 blocks for each channel. Encoding and decoding procedures are performed on each 8×8 block independently.

3.1 Encoder

Discrete Cosine Transform (DCT) A frequency domain representation including DCT has the capability to capture spatial redundancy and represent raw data with a small number of coefficients. JPEG encoder transforms the pixels in an 8×8 block by DCT [2] (2D DCT Type 2) to a coefficient matrix in the same size, denoted by $\mathbf{f} = (f_{i,j})$. The coefficient matrix is arranged in the order of DCT bases, where the first coefficient $f_{0,0}$ represents the DC component, the average value of all pixel data within the block, while the rest of 63 coefficients are the AC components. Note that JPEG DCT coefficients take integers in the range of $[-2^{10}, 2^{10})$.

Quantization The elements in the DCT coefficient matrix are divided by the corresponding entries in a quantization matrix $\mathbf{q} = (q_{i,j})$ preset in the bitstream and rounded up to the nearest integers. The quantized DCT is represented by $\hat{\mathbf{f}} = (\hat{f}_{i,j})$, where $\hat{f}_{i,j} = \lceil f_{i,j}/q_{i,j} \rceil$. The performance of compression is determined mostly in this step.

Symbol Coding The quantized DCT coefficients are reordered in a zigzag manner, forming a vector, and then coded into intermediate symbols. The reshaped vector is denoted by $\hat{\mathbf{f}} = (\hat{f}_0, \hat{f}_1, \ldots, \hat{f}_{63})$, where $\hat{f}_0$ is the quantized DC coefficient. From this point, DC and AC coefficients are treated separately as follows. First, the encoder applies differential pulse code modulation (DPCM) to DC coefficients of a whole image in the raster-scan order. In other words, by letting $\hat{f}_0^j$ be the DC coefficient in the j^{th} block, the encoded symbol δ_j is given by $\delta_j = \hat{f}_0^j - \hat{f}_0^{j-1}$ for $j \neq 0$ while $\delta_0 = \hat{f}_0^0$. Second, it performs run-length encoding (RLE) using AC coefficients, which scans the elements in $\hat{\mathbf{f}}_{-0} = (\hat{f}_1, \ldots, \hat{f}_{63})$ and assigns a tuple symbol (r_i, s_i) to each of nonzero coefficients, $\hat{f}_i (\neq 0)$. Note that r_i and s_i denote the run-length of zero that immediately precedes $\hat{f}_i$ and the number of bits needed to encode $\hat{f}_i$, respectively. There is an exception that symbols are generated for $\hat{f}_i = 0$; when total length of consecutive zeros exceedes 16 before encountering a nonzero, every succession of 16 zeros are coded into $(r_i, s_i) = (15, 0)$. If the remaining coefficients are all zeros, we mark a special symbol, $(r_i, s_i) = (0, 0)$, which denotes end-of-block (EOB). Note that the EOB symbol is not necessary when $\hat{f}_{63} \neq 0$ since it is redundant to clarify the case. Finally, a sequence of generated symbols for $\hat{\mathbf{f}}_{-0}$ after RLE looks like

$$(r_{i_1}, s_{i_1})\hat{f}_{i_1}; (r_{i_2}, s_{i_2})\hat{f}_{i_2}; \cdots; (r_{i_K}, s_{i_K})\hat{f}_{i_K}$$

where $\{i_k\}_{k=1}^K$ are the selected indices in an increasing order for which symbols are generated.

Entropy Coding Each symbol is entropy-coded by the Huffman coding or the arithmetic coding. Our work adopts the Huffman coding, the most common method, based on the Huffman tables for DC and AC specified in the bitstream.

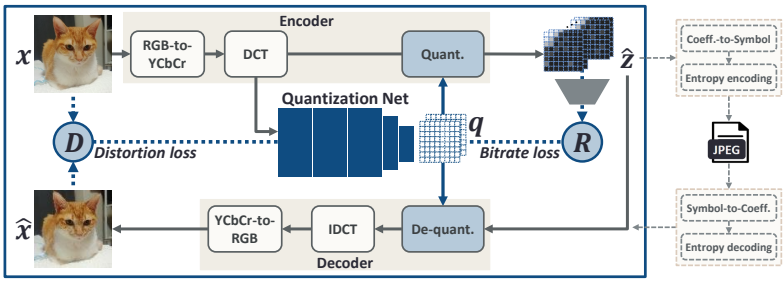

Fig. 2. The overall training framework of our method. The blue dashed lines are for the loss calculation whose gradients update our quantization network. The blue bold arrows indicate that the output of the quantization network is used as a quantization table for encoding/decoding processes. Bitrate loss is drawn from the pretrained network (colored grey). The grey dashed operations on the right-hand side are not involved in the training procedure.

3.2 Decoder

The decoding procedure is given by the inverse operations of the encoder. Note that, as quantization step involves rounding operations, the inverse quantization, which is given by an element-wise product of the quantized DCT coefficient and the quantization table, leads to inevitable loss of information within the original image.

4 Quantization Network

We discuss how to predict a quantization table for task-specific image compression using the proposed quantization network.

4.1 Framework

Let $\mathcal{X}$ and $\mathcal{Z}$ be a set of training images and a set of quantized representations, respectively. Each image $\mathbf{x} \in \mathcal{X}$ is mapped to $\hat{\mathbf{z}} \in \mathcal{Z}$ by a JPEG encoder, *i.e.*, $\hat{\mathbf{z}} = \text{Enc}(\mathbf{x}, \mathbf{q})$, where $\mathbf{q}$ denotes a set of coding parameters. *i.e.*, a quantization table. The bitrate $R(\cdot)$ of $\hat{\mathbf{z}}$, which is proportional to the number of bits required to express $\hat{\mathbf{z}}$, is predicted by an approximation model and expected to be under a target value. On the other hand, a JPEG decoder (Dec) reconstructs an image by $\hat{\mathbf{x}} = \text{Dec}(\hat{\mathbf{z}}, \mathbf{q})$ while incurring some distortion due to its lossy properties. Note that we can bypass symbol coding and entropy coding functions, *e.g.*, the Huffman encoding and decoding for training because they are performed outside the pipeline of the network. The objective of the proposed compression algorithm is to minimize distortion between an input and output pair given a constraint $R \leq R_{\text{target}}$ by identifying the optimal $\mathbf{q}$ for each $\mathbf{x}$. This is reformulated as an

unconstrained optimization problem by introducing the Lagrange multiplier λ that depends on the distortion metric $D(\cdot,\cdot)$ and the regulairzer R_{target} as

$$\min_{\mathbf{q}} D(\hat{\mathbf{x}}, \mathbf{x}) + \lambda R(\hat{\mathbf{z}}). \tag{1}$$

The optimal $\mathbf{q}$ is given by learning a quantization network $Q(\cdot)$ that directly estimates a quantization matrix from DCT coefficients $\mathbf{F}(=(\mathbf{f}^j))$, *i.e.*, $\mathbf{q} = Q(\mathbf{F})$. Then, for given n training images, $\{\mathbf{x}^i\}_{i=1}^n$, the training loss for $Q(\cdot)$ becomes

$$\mathcal{L}_Q = \sum_{i=1}^{n} D(\text{Dec}(\hat{\mathbf{z}}^i, \mathbf{q}^i), \mathbf{x}^i) + \lambda R(\text{Enc}(\mathbf{x}^i, \mathbf{q}^i)), \tag{2}$$

where $\mathbf{z}^i = \text{Enc}(\mathbf{x}^i, \mathbf{q}^i)$ and $\mathbf{q}^i = Q(\mathbf{F}^i)$. Fig. 2 illustrates the overall scheme of our algorithm, where the distortion loss can be given by perceptual image quality or task-specific error.

Although we make a mathematical formulation of the objective function in (2), there exist multiple non-differentiable components in the loss. In particular, it is not straightforward to define a differentiable bitrate loss function or its surrogate. To handle this challenge, we design a deep neural network for estimating a proper bitrate in (2) that is trainable via the standard error backpropagation. We present the details of our model for bitrate approximation in Section 5. Before that, we discuss how to make JPEG encoding and decoding procedure differentiable.

Differentiable JPEG Encoder and Decoder The original JPEG encoder and decoder contain non-differentiable components such as rounding operations during DCT transformation and quantization, which hampers training via the standard backpropagation. This issue is resolved by adopting a differentiable alternative introduced in [36], which simply replaces the derivative of a rounding function with an identity. This technique essentially changes only the derivative of the non-differentiable function without modifying the function itself. In other words, rounding functions are performed as normal in the forward pass while the backpropagation simply bypasses them.

4.2 Network Architecture

The quantization network of our framework takes DCT coefficients as an input and outputs two quantization tables, one for luminance and the other for chrominance, per image. Our base architecture consists of several convolutional layers and fully-connected (FC) layers. The role expected from the convolutional layers is to convey useful information in spatial and frequency domains while FC layers are to predict the appropriate quantization levels considering the absolute values of input coefficients and relative importance within an entire image for each frequency. Learning importance map for image compression appears frequently in recent deep learning based approaches [17,21,23,26]. They typically intend to

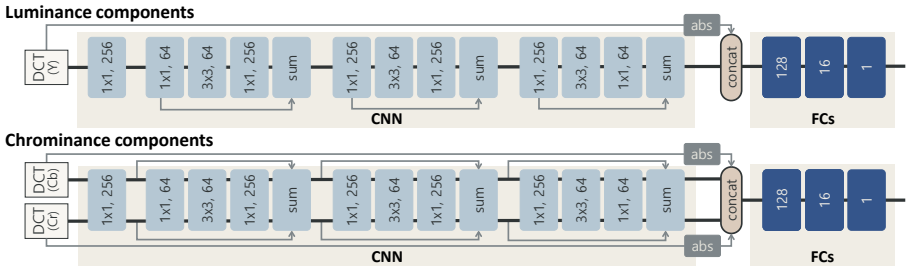

Fig. 3. The network architecture of the quantization network. The notation $K \times K, C$ in convolutional layers denotes that $K \times K$ kernels and C filters are applied. The concat operation before FC layers means the channel-wise concatenation. Batch normalization followed by ReLU activation comes right after each convolutional layer, though omitted in the figure for simplicity. The numbers in FC layers refer to the size of output sample.

allocate different number of bits depending on the complexity and the context of an image. We also adopt the same concept to control the quantization coefficients for each frequency using the features from a convolutional neural network (CNN).

The internal flow of our quantization network is depicted in Fig. 3. Before being passed to the network, a DCT coefficient matrix obtained from an image is reshaped to a 3D tensor whose channels represent individual DCT frequency; this process is similar to a reverse operation of the sub-pixel convolution (pixel shuffle) [33]. We expect that the salient regions in the informative channels have higher activations. Fig. 4 visualizes the output of the convolutional layers in the quantization network, which learns attention-like representations implicitly. The attention-like features are then augmented with the original DCT representations after applying the absolute value function. Finally, the FC layers produce quantization values for each of 64 DCT frequency bases, which forms a 8×8 quantization table. We construct two independent quantization tables, one for luminance and the other for chrominance. Note that the two channels in the chrominance channels share convolutional parameters and are eventually merged before the FC layers. Fig. 3 demonstrates the architectural difference between the two channels. We borrow a bottleneck skip connection block from ResNet50 [11], with only minor changes in order to match the channel sizes. Our model assumes no chroma subsampling (4:4:4 mode), which means that luminance and chrominance channels have the same resolution; it is possible to apply various ratios of chroma subsampling, which is handled merely by adding convolutional layers that reduce the spatial dimensions of chrominance components.

There are several prior works that learn deep neural networks using DCT coefficients in recent years [10, 25, 38]. Gueguen *et al.* [10] empirically show that DCT coefficients from JPEG are compatible with the standard convolutional neural network architectures designed for image classification tasks in RGB space. Although we have a different goal from those works, they give us an

Fig. 4. Visualization of the implicit attention learned by convolutional layers with classification loss. We use the learned representation of luminance components in Fig. 3 to generate the spatial attention map. Each feature map is min-max normalized to be unit-sized, and then averaged channel-wisely. The original RGB image is overlayed with the upsampled attention map.

intuition that we can take advantage of the standard convolutional layers for DCT coefficient inputs to learn the features in the frequency domain.

5 Approximation of Bitrate Measure

Predicting a code length given by the Huffman coding algorithm from DCT coefficients is complicated task. This is mainly because the length of AC symbols from RLE are irregular and the behavior of encoder thus becomes difficult to analyze. Motivated by this fact, we develop an approximate model of the bitrate measure $R(\cdot)$ using a deep neural network instead of an explicit analytic function. Since $R(\cdot)$ is exactly proportional to the length of bitstream that results from a sequence of discrete algorithms in JPEG entropy encoder, we design our network model to predict the final code length given the quantized vector considering the mechanism of JPEG encoder.

Several existing works [14, 22, 27] have studied on the approximation capabilities of a neural network theoretically by assuming certain criteria. Recently, advance of deep learning models based on various structures and components empower deep neural networks to break through complicated tasks. This direction is feasible to our case because the input of the bitrate measure R is bounded and a large number of synthetic examples can be sampled on our own.

Since JPEG operates the same processes over 8×8 basic blocks, we reduce our problem defined on a fixed dimension of domain, an 8×8 matrix. If an image is divided into n_B blocks, the bitrate measure is redefined as $R(\hat{\mathbf{z}}) = \sum_{j=1}^{n_B} R(\hat{\mathbf{z}}_j)$. The approximation of $R(\hat{\mathbf{z}}_j)$ is realized through several networks to embrace the encoding rules more accurately. The constituent networks are an intermediate symbol predictor for AC coefficients (S_{ac}) and a code length predictors for DC and AC coefficients (H_{dc} and H_{ac}, respectively). The predicted code length, which is proportional to our bitrate loss of an input image, is given by combining the three components together as

$$R(\hat{\mathbf{z}}) = \sum_{j=1}^{n_B} \mathrm{H}_{dc}(\delta_j) + \mathrm{H}_{ac}(\mathrm{S}_{ac}(\hat{\mathbf{f}}^j_{-0})), \tag{3}$$

where δ_j is a DC symbol and $\hat{\mathbf{f}}^j_{-0}$ is an AC vector from $\hat{\mathbf{z}}_j$. The followings are design procedures and implementation details of the three networks.

5.1 Symbol Prediction from RLE Model

Symbol coding process of JPEG encoding involves counting numbers in a serial order and several exception rules described in Section 3.1. We adopt an RNN model for the network S_{ac}, which mimics encoding rule during scanning a sequence and outputs a logit of symbol labels. The task can be formulated as a fully supervised classification using randomly generated inputs.

Our model learns the run-length size r_{i_k} for input coefficient f_{i_k} in a symbol, $(r_{i_k}, s_{i_k})\hat{f}_{i_k}$. For the coefficients with no corresponding symbol, *i.e.*, $i \notin \{i_1, \ldots, i_K\}$, we assign a dummy symbol r_{NIL} to facilitate training. In this correspond, an input AC coefficient sequence derives a symbol sequence $\mathbf{r} = (r_1, \ldots, r_{63})$ of the same dimension. Since the number of possible symbols are confined to 16 (integers in $[0, 15]$, see Section 3.1), the network learns to generate the logit representing the probability of symbols given a sequence of AC coefficients. We choose a bidirectional LSTM as our RNN model and let $g : \mathbb{R} \mapsto \mathbb{R}^{16}$ be a function that produces symbol probability vector. Then, our predicted symbol $\hat{r}_i$ is given by the result of the following classification task:

$$\ell^* = \arg\max g(\mathrm{BiLSTM}(f_i)), \tag{4}$$

i.e., $\hat{r}_i = \ell^*$. Now we can train this model by generating a training set, $\{(\hat{\mathbf{f}}_{-0}, \mathbf{r})\}$, from a random sequence in $\mathcal{Z}$. We adopt a single-layer bidirectional LSTM with 16 hidden states. However, since (4) is non-differentiable, we approximate $\hat{r}_i$ to $\tilde{r}_i$ using a Gumbel-softmax function as proposed in [15], which results in the final predicted symbols $\tilde{\mathbf{r}} = (\tilde{r}_1, \ldots, \tilde{r}_{63})$, inputs for the next regression step.

5.2 Regression Model for Final Code Length

We train two MLP models for DC and AC to predict the final code length based on the Huffman tables. We convert each element of DC symbols and AC vectors into the logarithmic scale, *i.e.*, f_i to $\log_2(|f_i|+1)$, since the lengths of codewords are clustered tightly in that scale. This adjustment makes our training more efficient and accurate by providing training examples more focused to frequently observed input patterns.

The networks are trained in a supervised manner based on the Huber loss using randomly generated vectors whose elements are sampled from $[-10, 10]$. Modifying the input scale in (3), the approximate code length of an image becomes

$$R(\hat{\mathbf{z}}) = \sum_{j=1}^{n_B} \{\mathrm{H}_{dc}(\log_2(|\delta_j| + 1)) + \sum_{i=1}^{63} \mathrm{H}_{ac}(\log_2(|f_i^j| + 1), \tilde{r_i}^j)\}. \tag{5}$$

Note that all three models in (3) are trained jointly. Our final regression model achieves 5.30% ± 1.60 on SMAPE (symmetric mean absolute percentage error) [9] when tested over 5,000 images sampled from the ImageNet validation set, where each image is JPEG compressed with random quality factor. SMAPE

is a common evaluation metric for regression problem that provides a reasonable judgment on relative errors.

In theory, one can further optimize a Huffman coding by using custom Huffman tables or adaptive Huffman coding algorithms, which dynamically updates code tables according to the change of distributions of input symbols [19]. Lakhani [20] reports that variants of the standard compression algorithm have limited benefits considering the additional cost in space and time. In this work, we pursue the method that keeps the standard codec intact and choose to follow the standard Huffman coding. Meanwhile, the proposed method that approximates the bitrate is still applicable to any extensions of Huffman coding.

6 Experiments

Dataset We train the quantization network on the ImageNet [32] dataset, which has 1,000 class labels. We constructed a test set for classification by sampling 5,000 images in 1,000 classes from the ImageNet validation set. We also used Caltech101 [8] for cross-dataset evaluation, where 406 randomly selected images are set aside for a test set. Flickr8k [12] and Kodak PhotoCD[1] datasets are additionally used for various assessments of our method.

Training We use the Adam [18] optimizer to learn models in all experiments. The learning rate starts from 0.001 and decreases by 0.1 for every 2k iterations. Each model is trained until the objective function converges.

Comparison The baseline dataset is generated by varying the JPEG quality factor scaled from 1 to 100 with the standard tables for each image. In all cases, including ours, 4:4:4 chroma subsampling mode is applied to test and evaluation. We compare two compressed images given by our algorithm and the standard JPEG by identifying the image in the baseline dataset with the (almost) same file size as ours. For each rate constraint parameter, we measure average bitrate and distortion (or performance) on test dataset to obtain a single point on the rate-distortion curve. To compute bitrate, we compute the actual bpp (bits per pixel) by applying the symbol coding and Huffman coding to the quantized representations.

Image Classification Accuracy For classification task, we employed Inception-v3 [35] for distortion metric. We adopted the pretrained model available in PyTorch. Fig. 5(a) presents that our quantization improves classification accuracy consistently, especially with aggressive compression rates. In our evaluation setting, the classification accuracy drops dramatically from around 0.4bpp and decent classification accuracy is achieved at higher bitrate. It is therefore natural that the proposed algorithm has relatively low gain on high bitrate range.

[1] http://r0k.us/graphics/kodak/

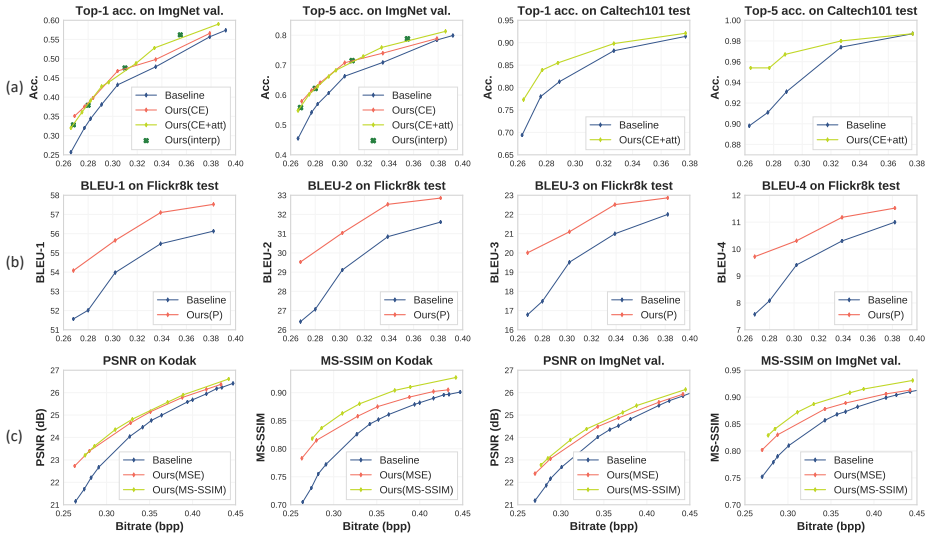

Fig. 5. Comparion by rate-distortion(/performance) curves. (a) Experiments with cross entropy loss of Inception-v3. Ours(CE+att) have the same architecture as ours(CE) except the attention module at the end of CNN. Classification accuracies(top-1/top-5) on original datasets are 0.76/0.93 for ImageNet and 0.96/0.99 for Caltech101. Ours(interp) are interpolated results from the models of ours(CE+att). (b) Experiments with Inception-v3 perceptual loss (ours(P)) for image captioning. Original test set records 61, 36, 26 and 14 on BLEU-1 to BLEU-4 scores. (c) Experiments with MSE loss and MS-SSIM loss.

In effect, our quantization network alleviates the failure of the classifier at low bitrates. Further, the results on Caltech101 imply the promising cross-dataset generalization performance as well.

Quality of Image Captioning To demonstrate the applicability of our method to various tasks, we run an experiment for image captioning, which requires more advanced semantic understanding than image classification. We choose Inception-v3 perceptual loss as a distortion metric. We train the image captioning model introduced in [39] on Flickr8k dataset using Inception-v3 feature extractor. BLEU scores are computed on generated captions as presented in Fig. 5(b). Our quantization method seems to benefit high-level tasks by preserving more semantic information in the images.

Percieved Quality We also trained the proposed quantization network using conventional distortion metrics such as pixel-wise MSE loss and MS-SSIM [40] loss. We evaluate the results on ImageNet and Kodak test dataset in terms of PSNR and MS-SSIM index because they are widely used as an approximation

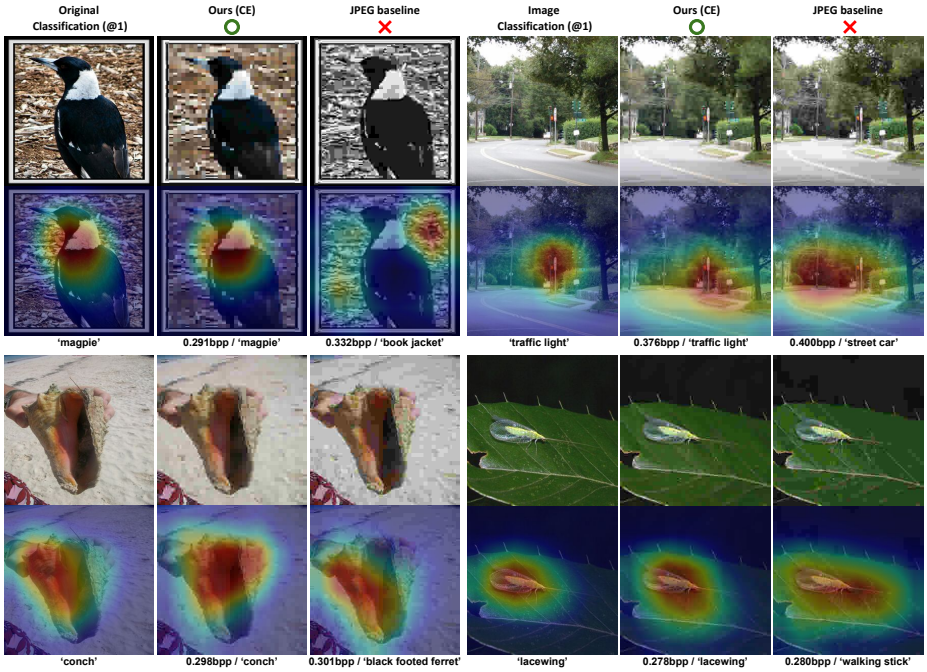

Fig. 6. Examples of classification results on experiment with Inception-v3 loss on ImageNet. Bitrates and predicted classes are shown below the images with CAM visualizations.

to human perception of visual quality. Fig. 5(c) shows that our methods surpass the baseline on both metrics, especially on MS-SSIM.

Effect of Attention Module To examine whether the architecture of our quantization network can further improve the compression performance, we added a simple attention module at the end of the original convolution layers in Fig. 3. Our attention module is identical to CBAM [42] except that we omit max-pooled features and spatial attention is drawn ahead of channel attention. Fig. 5(a) illustrates that our quantization network benefits from the attention module, which implies the potential improvement of our method by introducing better attention mechanisms or CNN architectures.

Interpolation using Multiple Models We present that the interpolation of quantization tables from multiple quantization networks is valid for the bitrate control. As shown in ImageNet plots in Fig. 5(a), interpolated results based on two models still surpass the baseline and also aligned well to the curve given by our algorithm without interpolation. This implies that we can realize an arbitrary bitrate by learning several landmark quantization networks and interpolating the resulting quantization tables for JPEG encoding.

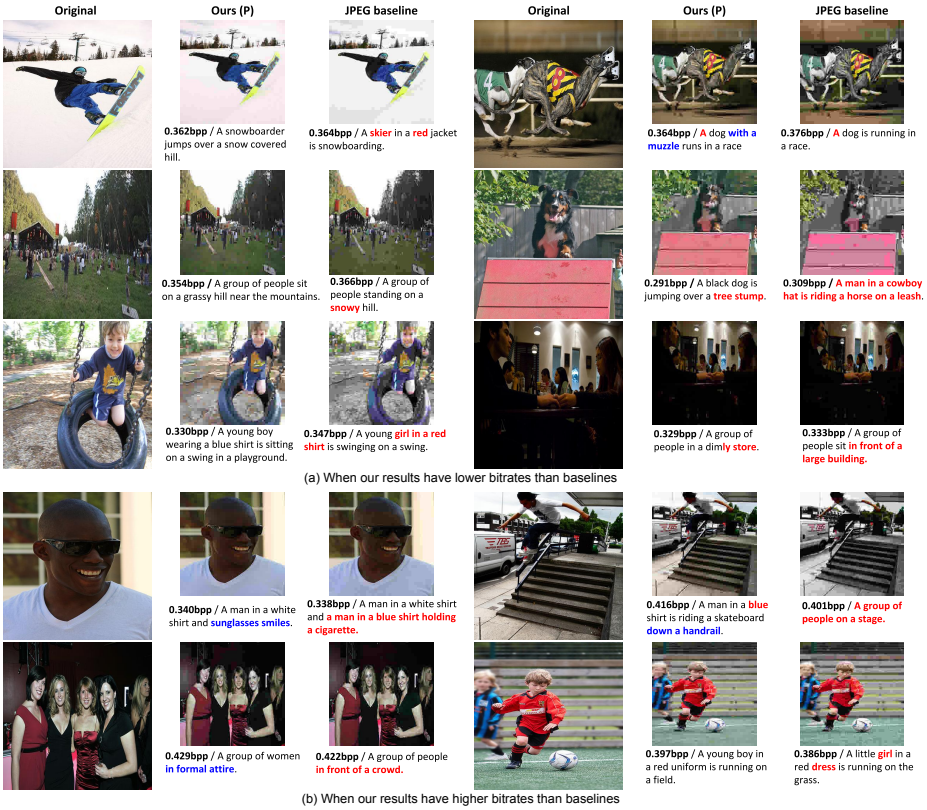

Fig. 7. Comparison on the captions generated by the RNN based caption generator on Flickr8k dataset. (a) Left: Only baseline results commit errors. Right: Both ours and baselines contain mistakes. (b) Ours outperform the baseline results. Errors are colored red and extra descriptions captured only by ours are colored blue.

Qualitative Results Fig. 6 presents qualitative comparisons between the JPEG baseline and our algorithm, where the JPEG baseline predicts wrong labels while ours approach is correct even with lower bitrates. In addition, we visualize class activation mappings (CAM) [44] for top-1 class on each image. The localized regions in the images support the proposed method on the classification task. We provide more examples with CAM localizations in the supplementary document.

Fig. 7 shows the qualitative results of generated captions. We observe that the captions on our compressed images contain fewer mistakes than those on baselines and often grasp detailed information of images missed in the baselines. When our images have a slightly higher bitrate than the baselines, the proposed approach obtains appealing improvements on captions that are hardly seen in the counterpart.

Some qualitative results on Kodak dataset are shown in Fig. 8. Generally, our quantization yields more natural outputs with lower bitrates; color changes are

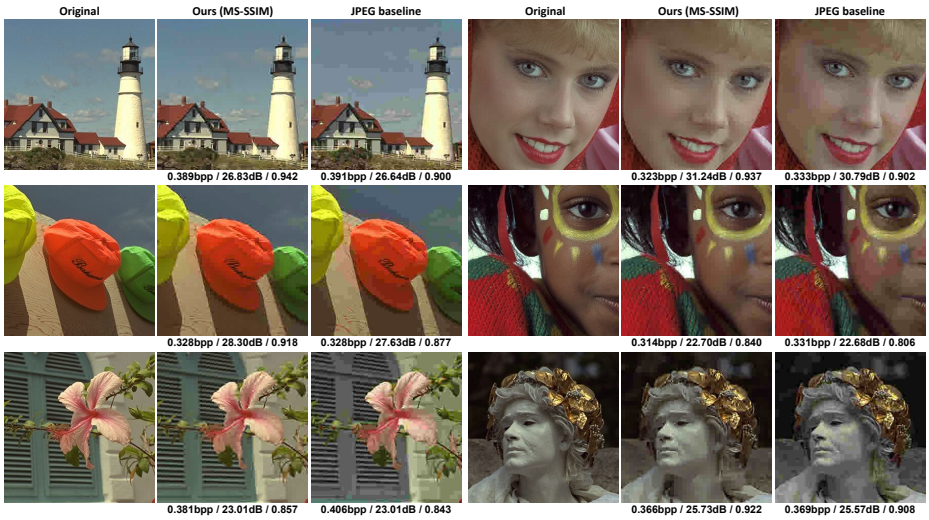

Fig. 8. Comparison on the selectively cropped samples from the Kodak dataset. Bitrate, PSNR and MS-SSIM values are notated below each image.

smoother in monotonous regions and complex structures are less blurred. Unlike the default quantizations, which normally quantize chrominance components more than luminance components, our approach seems to perform more balanced bitrate optimization, which, in turn, produces better visual quality.

7 Conclusion

We proposed a framework for data-driven and task-aware JPEG quantization, which has not been studied intensively. To facilitate our training, we designed a differentiable code length prediction technique given quantized representations based on deep neural networks. The proposed quantization network has been incorporated into the standard JPEG algorithm and improved compression performance substantially. Especially, it is highly effective when the bitrate is very low, where JPEG suffers from severe degradation. The proposed method is indeed practical since it is compatible with any JPEG Codecs with no additional cost for decoding. We believe that our method can be explored further jointly with various options of JPEG compression and for more powerful network architectures.

Acknowledgments This work was partly supported by Kakao and Kakao Brain Corporation, and IITP grant funded by the Korea government (MSIT) (2016-0-00563, 2017-0-01779). We also thank Hyeonwoo Noh for fruitful discussions.

References

1. Agustsson, E., Mentzer, F., Tschannen, M., Cavigelli, L., Timofte, R., Benini, L., Gool, L.V.: Soft-to-hard vector quantization for end-to-end learning compressible representations. In: NeurIPS (2017) 2
2. Ahmed, N., Natarajan, T., Rao, K.R.: Discrete cosine transform. IEEE Transactions on Computers **100**(1), 90–93 (1974) 4
3. Ahumada Jr, A.J., Peterson, H.A.: Luminance-model-based DCT quantization for color image compression. In: SPIE (1992) 3
4. Ballé, J., Laparra, V., Simoncelli, E.P.: End-to-end optimized image compression. In: ICLR (2017) 2
5. Ballé, J., Minnen, D., Singh, S., Hwang, S.J., Johnston, N.: Variational image compression with a scale hyperprior. In: ICLR (2018) 2
6. Davisson, L.D.: Rate-distortion theory and application. Proceedings of the IEEE **60**(7), 800–808 (1972) 1
7. Dodge, S., Karam, L.: Understanding how image quality affects deep neural networks. In: QoMEX (2016) 3
8. Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: CVPRW (2004) 10
9. Flores, B.E.: A pragmatic view of accuracy measurement in forecasting. Omega **14**(2), 93–98 (1986) 9
10. Gueguen, L., Sergeev, A., Kadlec, B., Liu, R., Yosinski, J.: Faster neural networks straight from JPEG. In: NeurIPS (2018) 7
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016) 7
12. Hodosh, M., Young, P., Hockenmaier, J.: Framing image description as a ranking task: Data, models and evaluation metrics. Journal of Artificial Intelligence Research **47**, 853–899 (2013) 10
13. Hopkins, M., Mitzenmacher, M., Wagner-Carena, S.: Simulated annealing for JPEG quantization. In: DCC (2018) 3
14. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural networks **2**(5), 359–366 (1989) 8
15. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. In: ICLR (2017) 9
16. Jayant, N., Johnston, J., Safranek, R.: Signal compression based on models of human perception. Proceedings of the IEEE **81**(10), 1385–1422 (1993) 3
17. Johnston, N., Vincent, D., Minnen, D., Covell, M., Singh, S., Chinen, T., Jin Hwang, S., Shor, J., Toderici, G.: Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks. In: CVPR (2018) 2, 6
18. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015) 10
19. Knuth, D.E.: Dynamic huffman coding. Journal of algorithms **6**(2), 163–180 (1985) 10
20. Lakhani, G.: Optimal huffman coding of dct blocks. IEEE Transactions on circuits and systems for video technology **14**(4), 522–527 (2004) 10
21. Lee, J., Cho, S., Beack, S.K.: Context-adaptive entropy model for end-to-end optimized image compression. In: ICLR (2019) 2, 6

22. Leshno, M., Lin, V.Y., Pinkus, A., Schocken, S.: Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. Neural networks **6**(6), 861–867 (1993) 8
23. Li, M., Zuo, W., Gu, S., Zhao, D., Zhang, D.: Learning convolutional networks for content-weighted image compression. In: CVPR (2018) 2, 6
24. Liu, Z., Liu, T., Wen, W., Jiang, L., Xu, J., Wang, Y., Quan, G.: DeepN-JPEG: A deep neural network favorable JPEG-based image compression framework. In: DAC (2018) 3
25. Lo, S.Y., Hang, H.M.: Exploring semantic segmentation on the DCT representation. In: MMAsia (2019) 7
26. Mentzer, F., Agustsson, E., Tschannen, M., Timofte, R., Van Gool, L.: Conditional probability models for deep image compression. In: CVPR (2018) 2, 6
27. Mhaskar, H.N., Micchelli, C.A.: Approximation by superposition of sigmoidal and radial basis functions. Advances in Applied mathematics **13**(3), 350–373 (1992) 8
28. Minnen, D., Ballé, J., Toderici, G.D.: Joint autoregressive and hierarchical priors for learned image compression. In: NeurIPS (2018) 2
29. Monro, D.M., Sherlock, B.G.: Optimum DCT quantization. In: DCC (1993) 3
30. Pennebaker, W.B., Mitchell, J.L.: JPEG: Still image data compression standard. Springer Science & Business Media (1992) 1
31. Ratnakar, V., Livny, M.: Rd-opt: An efficient algorithm for optimizing DCT quantization tables. In: DCC (1995) 3
32. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115**(3), 211–252 (2015) 10
33. Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A.P., Bishop, R., Rueckert, D., Wang, Z.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: CVPR (2016) 7
34. Skodras, A., Christopoulos, C., Ebrahimi, T.: The JPEG 2000 still image compression standard. IEEE Signal processing magazine **18**(5), 36–58 (2001) 1
35. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: CVPR (2016) 10
36. Theis, L., Shi, W., Cunningham, A., Huszár, F.: Lossy image compression with compressive autoencoders. In: ICLR (2017) 2, 6
37. Toderici, G., Vincent, D., Johnston, N., Jin Hwang, S., Minnen, D., Shor, J., Covell, M.: Full resolution image compression with recurrent neural networks. In: CVPR (2017) 2
38. Verma, V., Agarwal, N., Khanna, N.: DCT-domain deep convolutional neural networks for multiple JPEG compression classification. Signal Processing: Image Communication **67**, 22–33 (2018) 7
39. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: A neural image caption generator. In: CVPR (2015) 11
40. Wang, Z., Simoncelli, E.P., Bovik, A.C.: Multiscale structural similarity for image quality assessment. In: ACSSC (2003) 11
41. Watson, A.B.: Visually optimal DCT quantization matrices for individual images. In: DCC (1993) 3
42. Woo, S., Park, J., Lee, J.Y., Kweon, I.S.: CBAM: Convolutional block attention module. In: ECCV (2018) 12
43. Zhang, L., Zhang, L., Mou, X., Zhang, D.: FSIM: A feature similarity index for image quality assessment. IEEE Transactions on Image Processing **20**(8), 2378–2386 (2011) 3

44. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: CVPR (2016) 13