Learning from the CNN-based Compressed Domain

Zhenzhen Wang[1,2] zwang033@e.ntu.edu.sg
Minghai Qin[1] qinminghai@gmail.com
Yen-Kuang Chen[1] y.k.chen@ieee.org

Computing Technology Lab, DAMO Academy[1]
Nanyang Technological University[2]

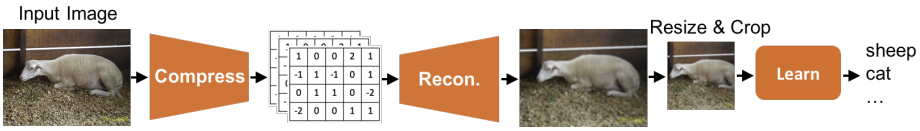
Abstract

Images are transmitted or stored in their compressed form and most of the AI tasks are performed from the reconstructed domain. Convolutional neural network (CNN)-based image compression and reconstruction is growing rapidly and it achieves or surpasses the state-of-the-art heuristic image compression methods, such as JPEG or BPG. A major limitation of the application of the CNN-based image compression is on the computation complexity during compression and reconstruction. Therefore, learning from the compressed domain is desirable to avoid the computation and latency caused by reconstruction. In this paper, we show that learning from the compressed domain can achieve comparative or even better accuracy than from the reconstructed domain. At a high compression rate of 0.098 bpp, for example, the proposed compression-learning system has over 3% absolute accuracy boost over the traditional compression-reconstruction-learning flow. The improvement is achieved by optimizing the compression-learning system targeting original-sized instead of standardized (e.g., 224x224) images, which is crucial in practice since real-world images into the system have different sizes. We also propose an efficient model-free entropy estimation method and a criterion to learn from a selected subset of features in the compressed domain to further reduce the transmission and computation cost without accuracy degradation.

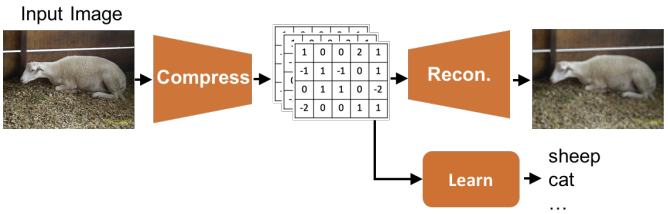
1. Introduction

Convolutional neural networks (CNNs) have revolutionized many visual-based understanding tasks, such as image classification [28, 31], object detection [12] and semantic segmentation [16]. Recently, the applications of CNNs are extended to non-visual understanding tasks, in particular, to the image compression task. That is, transforming an image from the color space domain (e.g., RGB domain) to a compressed representation. The purpose of image compression is to remove intrinsic redundancy in the image and thus be able to use much smaller number of bits to represent it, which is beneficial both in transmission and in data storage. Therefore, almost all images exist in the compressed domain. CNN-based compression methods [3, 6, 25, 5, 20, 24] have achieved promising performance and can compete with or even outperform the traditional heuristic methods such as JPEG [29], JPEG2000 [23], and BPG [4]. To perform computer vision (CV) tasks from the compressed domain, the compressed representation (a sequence of compressed features in Fig. 1) is reconstructed to RGB images first and then normalized to a matched size to be fed into CNNs for the CV tasks (see Fig. 1(a)). However, both of the CNN-based compression and reconstruction networks are computationally intensive because they are performed over original images of high resolutions. For example, a state-of-the-art four-layer CNN-based reconstruction network [6] requires over 900 GFLOPS to reconstruct a 1080p (1920×1080 pixels) image while a ResNet-50 [14] can perform an inference from the reconstructed and standardized image (224×224) with only 4 GFLOPS. Therefore, learning from the compressed domain without image reconstruction is crucial in reducing the computation complexity and the latency of the CV system.

Several works [8, 11, 32] were proposed to learn from the traditional discrete cosine transform (DCT) domain which is used in JPEG compression. In this work, we would like to explore the compressed domain defined by CNN-based compression networks. It is shown in [27] that learning from the compressed domain is feasible but the accuracy from the compressed domain drops significantly. Another limitation of existing works is that input size to the compression network is fixed, e.g., 224×224, during both training and inference. The real-world images to be compressed, however, are usually with variable resolutions. For example, the image height and width in ImageNet [9] validation dataset range from fifties to thousands. The lack of



(a) Learning from the reconstructed domain.



(b) Learning from the compressed domain.

Figure 1. Comparison of the conventional system (a) and the proposed system (b).

optimization for variable-size images results in inferior accuracy and thus limits the application of learning from the compressed domain.

Motivated by reducing computations of a practical system for image compression, reconstruction, and learning, we present an end-to-end solution. It consists of a compression network that accepts the original-size input image and convert it into the compressed domain, a reconstruction network that outputs a decompressed image of its original size, and learning networks that read variable-size compressed features directly from the compressed domain. Fig. 1 shows such an exemplar system. Instead of reconstructing images of different sizes and then spatially resizing them into the standard size for learning (see Fig. 1(a)), we manage to learn directly from the compressed-domain features of different sizes (see Fig. 1(b)). Note that in both cases, the size of the compressed features depends on the size of the input images, which can vary in a large dynamic range.

The development of CNN architectures and their software implementations enable the input to a CNN to be flexible. For example, ResNet [14] and MobileNetV2 [21] use the adaptive average pooling layer to convert a feature map of any size into one single number for the last output layer. Thus they can algorithmically classify images of any size. However, conventionally trained CNNs have inferior accuracy when the input image is not normalized to the matched size. For example, the accuracy of a ResNet-50 trained with input size 224×224 drops from 75.7% to 72% when the original-size images (from ImageNet validation dataset) are fed into it. The accuracy degradation is mainly due to the inconsistency between the fixed-size training and variable-size inference. In order to solve this, we train the compression-learning system accepting variable-size images and observe significant accuracy improvement, which is comparative or even better than from the reconstructed domain.

In addition to optimizing the learning network, we also propose a method to simplify the training of the compression-reconstruction networks. One of the challenges in training them is on minimizing the loss function that is called rate-distortion optimization (RDO). Existing works [3, 6] utilized sophisticated neural networks models to estimate the entropy in RDO and it needs extra efforts to train. Observing that compressed feature maps usually approximate Gaussian distributions, we propose to directly use differential entropy [7] as an approximation. Due to its close form, the differential entropy is differentiable with respect to the weights and allows the gradient of the loss function to be flowed back into the compression network during training.

Observing that features in the compressed domain exhibit large variation in the amount of information they contain, we propose to learn only from a selected subset of compressed features to further reduce the transmission and computation cost. We have shown that using only one eighth or less number of the original compressed features, we can still maintain or even improve the learning accuracy.

We summarize our contributions as follows.

- We show that learning directly from the compressed domain can achieve comparative or even better image classification accuracy than from the reconstructed RGB domain while avoiding large amount of computation in the reconstruction network. Experiments show that this improvement (over reconstruction) is more than 1% and 2% for modest and large compression ratio, respectively. As far as we know, our work is the first clear demonstration of superior accuracy of learning from CNN-based domain over the reconstructed domain.

- We introduce a model-free method to estimate the discrete entropy of the compressed features such that this

estimation is differentiable with respect to the weights in the compression network, which can be trained more efficiently than using complicated entropy estimation networks [3, 6]. The correlation between the proposed estimation and the true entropy is close to 99%.

- We propose a criterion to learn from a selected subset of compressed features instead of all features to further reduce the transmission and computation cost. The resultant accuracy with 12.5% features is similar to or even better than learning from all features.
- Compared to the state-of-the-art results in [27], our proposed compression-reconstruction/learning system with a fixed-size input (224×224) improves (PSNR, accuracy) by $(3.31$ dB$, 6.83\%)$ for image classification and $(3.31$ dB$, 4.74\%)$ for semantic segmentation, respectively, where our proposed compression system has almost the same set of bpps as [27].

2. Related Work

CNN-based compression. In recent years, a number of CNN-based image compression networks have been proposed. [25, 26] used RNN-based network to compress images with variable bit-rate. [17] improved the performance by a better initial hidden state and [3, 6] outperformed existing methods by using shallow networks, Generative adversarial network (GAN) [13] has also been exploited. The discriminator usually works as the quality metric to distinguish reconstructed and original images [22], and a generator is often used as a decoder to recover images from the compressed domain [1].

In [30], segmentation maps were used to direct bit allocation, as the region of interest (ROI) denoting by these maps usually attracts more human visual attention than the background. [19] proposed an importance map instead of using the segmentation map. Following the similar idea, [2] treated the segmentation maps as extra channels and the reconstruction network can thus have more information for better reconstruction quality. Our work differentiates from all mentioned works in that we also perform learning tasks (e.g., classification, segmentation) from the compressed domain while they only do image reconstruction. Also, the saliency map can also used as the criteria for bit allocation [20].

Learning from the compressed domain. Several works perform learning tasks in the frequency domain. JPEG, the most widely-used compression format, uses DCT to convert the RGB domain into the frequency domain. [11, 32] took the DCT coefficients as the input to CNNs for image classification, object detection and instance segmentation. Our method differs from these works in that the compression is accomplished by a CNN-based network, instead of a DCT transformation.

The most related work to ours is [27], which also used the compressed features derived from CNN-based compression network. Our work differs from [27] in several ways. Most importantly, the input size to the compression network in [27] is fixed (i.e., 224) while our compression-learning system is optimized and aims to accept the original-size image. This differentiation is critical as a) real-world images to be compressed are in various sizes; b) it provides large accuracy improvement, e.g., the classification accuracy of ImageNet validation set is improve from 55.18% to 70.37% at 0.098 bpp. Other works (e.g., [15]) will sacrifice 8% to 18% accuracy (at a modest compression ratio) and thus not feasible to real-world applications.

3. Method

3.1. Problem Statement

A complete image analysis system consists of modules for image capture, image compression, image reconstruction, and image learning. Throughout the paper, we denote the originally captured image by $x \in \mathcal{X}$ where $\mathcal{X}$ is the RGB domain. The image compression module is a function $z = CompNet(x)$ that transforms the image in $\mathcal{X}$ to the compressed domain $\mathcal{Z}$. z is then entropy encoded for transmission or storage. The image reconstruction module is a function $\hat{x} = RecNet(z)$ that transforms the compressed representation $z \in \mathcal{Z}$ to the reconstructed image $\hat{x}$ in the reconstructed RGB domain $\hat{\mathcal{X}}$. In this paper, $CompNet(x)$ is composed of a CNN-based compression network and a quantizer, and $RecNet(z)$ is a CNN-based reconstruction network. $CompNet(x)$ and $RecNet(z)$ are jointly trained to minimize the difference between x and $\hat{x}$ and to minimize the number of bits to represent z. The reconstructed image is meant to be presented for human to understand. The image learning module is a function $LearnNet()$ that outputs the desired learning result for the original image x. Almost all conventional machine learning tasks are performed from the RGB domain, which means the input domain of $LearnNet(\hat{x})$ has to be the reconstructed domain $\hat{\mathcal{X}}$ and thus the image reconstruction is required before learning. However, most automatic systems do not aim for human to understand, but for machine to learn. In this paper, we define $LearnNet(z)$ over $\mathcal{Z}$ such that the learning accuracy of $LearnNet(z)$ is as good as $LearnNet(\hat{x})$. This converts the reconstruction task and the learning task from sequential to parallel, effectively removing the latency in performing $RecNet(z)$. Another potential benefit of learning directly from the compressed domain is that we can avoid information loss in $RecNet(z)$ as reconstruction and learning might have different need in processing information from z.

As presented, the system in this paper is a multi-task system where compressed features can be reconstructed for hu-

man to see and be learned for machine to understand. For this purpose, the input x to this system is a real-world image with various resolutions. For example, the height/width of images in the ImageNet validation set ranges from 55 to 5005 pixels. Since $CompNet(x)$ is an all-CNN network that downsamples x by a fixed scaling factor, the compressed feature z has various resolutions as well. We observe that resizing compressed features to a fixed size in the compressed domain catastrophically hurts the learning accuracy. Consequently, $LearnNet(z)$ should be able to accept variable-size input z without hurting the learning accuracy. This is fundamentally different from learning after reconstruction because the reconstructed image can be spatially re-scaled to a proper size for CNNs. For example, 224×224 has become a standard input size for classification networks on ImageNet dataset.

3.2. Network Architectures

In this section, we present the detailed network architecture for the proposed compression network, reconstruction network, and learning networks. We will explore two typical learning tasks, namely, image classification and semantic segmentation, and we will denote the corresponding $LearnNet(z)$ by $ClsNet(z)$ and $SegNet(z)$, respectively.

Compression network and reconstruction network. We use the state-of-the-art compression/reconstruction network in [6] as $CompNet(x)$ and $RecNet(z)$. The only difference is that we do not have their entropy estimation model since we propose a model-free estimation method (see Section 3.3).

Learning networks $LearnNet(z)$**.** We manage to apply network structures similar to ResNet-50 and MobileNetV2 since they contain some useful and thus widely-used building blocks such as residual blocks and depth-wise separable convolutions. Assume the original image x has dimensions $w \times h \times 3$ in the RGB domain, then the compressed features have dimensions $w/16 \times h/16 \times C$ where C is the number of channels in the compressed features. Since it has smaller w and h but larger C than the RGB domain image, we skip the input layer (stride-2 convolution and max-pooling if existed) of a conventional CNN architecture, e.g., ResNet-50 or MobileNetV2. We adjust the number of channels in the residual block to match the number of compressed channels C such that existing CNN architectures can accept the compressed-domain features as input.

The widely-used and -accepted datasets such as ImageNet and PASCAL VOC 2012 [10] contain images with much smaller resolutions than the real-world images. Therefore, the resolution of compressed features is usually very small. For example, images of width less than 300 are quite common in ImageNet and they have width less than 18 in the compressed domain, which is too small for existing deep CNNs to make an inference. In order to perform learning tasks on compressed features for these dataset, we propose to use a deconvolutional layer and a pixel shuffle layer to upscale the image by two and then followed by a 1×1 convolutional layer to adjust the number of channels to match the existing ResNet-50 or MobileNetV2 architecture. For the same reason, we skip the first residual block of the ResNet-50 or the first two inverted residual blocks of the MobileNetV2 to reduce the number of downsizing operations in the CNNs. It also helps to reduce the computations (GFLOPS) during inference. We call the corresponding learning networks ResNet-41 and MobileNetV2-45 to indicate their number of layers. The ResNet-41 is also served as the backbone for the segmentation networks. The detailed learning network architecture is shown in Table 1 and we use the same notations as they are in [14, 21].

Table 1. Learning network architectures.

(a) ResNet-41 as the learning network

layer name	input size	#block, stride
up-scale	$\frac{w}{16} \times \frac{h}{16}$	stride 0.5
conv3_x	$\frac{w}{8} \times \frac{h}{8}$	$\times 4$, stride 1
conv4_x	$\frac{w}{8} \times \frac{h}{8}$	$\times 6$, stride 2
conv5_x	$\frac{w}{16} \times \frac{h}{16}$	$\times 3$, stride 2
avg_pool	$\frac{w}{32} \times \frac{h}{32}$	-
conv2d	1×1	-

(b) MobileNetV2-45 as the learning network

operator	input size	t	c	n	s
up-scale	$\frac{w}{16} \times \frac{h}{16}$	-	320	1	0.5
bottleneck	$\frac{w}{8} \times \frac{h}{8}$	6	32	3	1
bottleneck	$\frac{w}{8} \times \frac{h}{8}$	6	64	4	2
bottleneck	$\frac{w}{16} \times \frac{h}{16}$	6	96	3	1
bottleneck	$\frac{w}{16} \times \frac{h}{16}$	6	96	3	1
bottleneck	$\frac{w}{32} \times \frac{h}{32}$	6	320	1	1
conv2d, 1×1	$\frac{w}{32} \times \frac{h}{32}$	-	1280	1	1
avg_pool, 7×7	$\frac{w}{32} \times \frac{h}{32}$	-	-	1	-
conv2d, 1×1	1×1	-	1000	1	1

It should be noted that for real-world images with higher resolution, the upscaling layers can be omitted and the complete set of residual blocks can be used for inference from the compressed domain.

3.3. Loss Function

The loss function of the proposed multi-task system can be decomposed into several parts as follows,

$$\mathcal{L}([\theta_{comp}, \theta_{res}, \theta_{cls}, \theta_{seg}, \dots]) = \mathcal{L}_{CompNet} + \beta_1 \mathcal{L}_{RecNet} + \beta_2 \mathcal{L}_{ClsNet} + \beta_3 \mathcal{L}_{SegNet} + \cdots \quad (1)$$

where $\theta_{comp}, \theta_{res}, \theta_{cls}, \theta_{seg}, \dots$ are sets of trainable weights for the compression network $CompNet(x)$, recon-

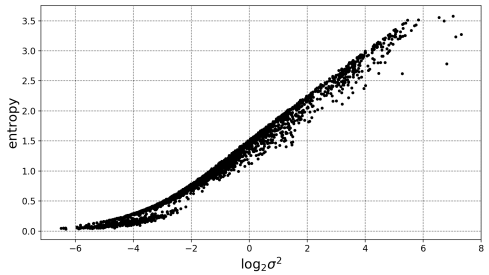


Figure 2. Correlation of the discrete entropy and $\log\sigma^2$ of the compressed features.

struction network $RecNet(z)$ and the learning networks such as $ClsNet(z)$ or $SegNet(z)$. And $\beta_1, \beta_2, \beta_3 \in [0, 1]$ are coefficient of each component. Define $\mathcal{L}_{RecNet}$ as the mean-squared-error (MSE) between x and $\hat{x}$, i.e.,

$$\mathcal{L}_{RecNet} = \|x - \hat{x}\|^2 = \|x - RecNet(CompNet(x))\|^2 .$$

Define $\mathcal{L}_{ClsNet}$ and $\mathcal{L}_{SegNet}$ as the conventional loss of the learning tasks from the RGB domain, e.g., cross-entropy for classification and pixel-wise cross-entropy for segmentation. These losses are all differentiable with respect to the weights. On the other hand, $\mathcal{L}_{CompNet}$ contains trainable weights θ_{comp} and it is usually estimated by the discrete entropy of the compressed features z. Theoretically, the entropy H of a discrete random variable z is defined as $\mathcal{L}_{CompNet} \approx H(z) = \mathbb{E}[-\log_2 P(z)] = \sum_z -P(z)\log_2 P(z)$, where $P(z)$ is the discrete probability of z. Obviously $H(z)$ is not differentiable with respect to z, and thus not differentiable to θ_{comp}. It was proposed in [3, 6] to use a neural network to estimate $H(z)$ from z, which brings more complexity during the training.

We propose an efficient yet effective method to avoid adding more weights into the training procedure. It is observed that values in the compressed features usually follow a Gaussian-like distribution and we can use a continuous Gaussian distribution to estimate the entropy of its quantized values. For a continuous Gaussian random variable $Z \sim \mathcal{N}(\mu, \sigma^2)$, the differential entropy $h(Z)$ can be expressed in a closed form

$$h(Z) = \frac{1}{2}\log 2\pi e\sigma^2 = \frac{1}{2}\log 2\pi e + \frac{1}{2}\log\sigma^2.$$

According to [7][Theorem 8.3.1], the differential entropy $h(Z)$ and its discrete entropy (after quantization) $H(z)$ satisfies

$$H(z) + \log\Delta \to h(Z), \text{as } \Delta \to 0,$$

where Δ is the quantization step. Since Δ and the term $\frac{1}{2}\log 2\pi e$ are both constant values in this paper, we can approximate the loss function of the compression network by

$$\mathcal{L}_{CompNet} \approx \log\sigma^2,$$

where σ^2 is the variance of the values in the compressed features $z = (z_1, \ldots, z_N)$ and N is the number of elements in z. It can be expressed as $\sigma^2 = \frac{1}{N}\sum(z_i - \mu_z)^2$. Therefore, $\mathcal{L}_{CompNet}$ is differentiable with respect to θ_{comp} in our method. Intuitively, the number of bits needed to represent a random variable should be positively correlated with the variance of the random variable. To justify the approximation quantitatively, Fig. 2 shows the correlation between the true discrete entropy and the approximation by $\log\sigma^2$ for the compressed features in Kodak dataset [18]. It is observed that their linear correlation is close to 99%.

3.4. Training and inferencing the compression, reconstruction, and learning networks

3.4.1 Training flow

In this paper, we first train the compression network $CompNet(x)$ and the reconstruction network $RecNet(z)$ together by setting $\beta_2, \beta_3, \ldots$ to be zeros. By adjusting β_1, we can obtain different rate-distortion tradeoffs. To be more specific, the training images are randomly resized and cropped to 256×256 to train θ_{comp} in $CompNet(x)$ and θ_{res} in $RecNet(z)$. θ_{comp} is then fixed during the training of the learning networks $LearnNet(z)$. We did not jointly optimize $CompNet(x)$ and $LearnNet(z)$ for two reasons. First, image compression and reconstruction is the basic task and $CompNet(x)$ is preferred not to be changed each time a task (e.g., classification or segmentation) is added. Second, joint training will possibly affect the reconstruction quality and thus it is not a fair comparison even if the learning accuracy from the compressed domain is as good as that from the reconstructed domain.

3.4.2 Inferencing the original image without resizing

As mentioned in Section 3.1, the proposed multi-task system is required to accept images for different sizes. For an image x of dimension $w \times h \times 3$, the reconstructed image $\hat{x}$ is obtained by

$$\hat{x} = RecNet(CompNet(x)),$$

and the learning result (e.g., the classification label of the original image x) is obtained by

$$\text{result} = LearnNet(CompNet(x)).$$

Some widely-used CNN architectures such as ResNet-50 and MobileNetV2 support input with variable w and h to pass through all the convolutional layers and use an adaptive average pooling layer to standardize the size of feature maps. Therefore, it is technically legitimate to inference an original image in the proposed compression-learning network without any modification. However, the learning accuracy drops significantly as the CNNs are conventionally trained with a fixed input size.

3.4.3 Training the compression-learning network with variable-size input

In order to improve the accuracy of the learning network with variable-size inputs, we propose a training method to improve the robustness of $LearnNet(z)$ against variable-size z. Ideally, it is better to resize each image to different sizes during training to maximally improve the robustness. However, current training framework restricts tensors in one mini-batch to be the same shape. Therefore, we make a tradeoff by randomly resizing the input images to different sizes across mini-batches and keeping them the same size within a mini-batch. In this paper, we resize and crop the training images to a random size between 224 and 448. These two numbers are chosen empirically due to the observation that most of the image sizes in ImageNet are located inside this interval. This method can be generalized to larger image dataset (if exists) by randomly resizing them to a larger range.

3.4.4 Learning from a subset of compressed feature maps

Observing that all 256 feature maps in the compressed domain exhibit large variations in the amount of information they carry, we propose to learn from a subset of the compressed domain where feature maps with richest information are selected for the learning tasks. We define the criterion of the information richness to be the variance of the all signals in each feature map, since it is closely related to the entropy of a feature map (see Section 3.3). In practice, we calculate the average variance of 256 feature maps in the ImageNet training set and select 32 and 16 static feature maps to separately train two learning networks, respectively.

4. Experimental results

4.1. Experiment settings

We evaluate the compression-reconstruction/learning networks on some widely used image datasets. First, $CompNet(x)$ and $RecNet(z)$ are trained and validated on ImageNet. Then $CompNet(x)$ is fixed while $ClsNet(z)$ is trained and validated on ImageNet and $SegNet(z)$ is trained and validated on PASCAL VOC 2012.

Hyper-parameter tuning. Four pairs of $(CompNet(x), RecNet(z))$ are trained by setting $\beta_1 = (20, 50, 100, 200)$. We use Adam with initial learning rate 0.0001. The corresponding entropy of the compression networks are $(0.098, 0.30, 0.49, 0.72)$ bpp for ImageNet validation set.

Evaluation. We evaluate $ClsNet(z)$ by top-1 and top-5 accuracy, evaluate $SegNet(z)$ by pixel-wise mean intersection over union (mIoU), evaluate $CompNet(x)$ by bpp, and evaluate $RecNet(z)$ by image quality metric (PSNR).

Table 2. Classification accuracy comparison between learning from the compressed domain and from the reconstructed domain.

bpp	Method	ResNet	MobileNetV2
0.72	Learn-after-recon.	**74.59**	70.36
	Learn-from-recon.(Ours)	74.08	**70.44**
0.49	Learn-after-recon.	**73.93**	69.86
	Learn-from-recon.(Ours)	73.42	**70.08**
0.30	Learn-after-recon.	72.76	68.24
	Learn-from-recon.(Ours)	**73.32**	**69.41**
0.098	Learn-after-recon.	67.08	62.46
	Learn-from-recon.(Ours)	**70.37**	**64.81**

4.2. Learning from the compressed domain for original-size images

Table 2 shows the comparison of classification accuracy of ImageNet between the proposed method (learning from the compressed domain) and the conventional method where a standard ResNet-50 and MobileNetV2 is applied on the reconstructed the image. The input to the proposed compression-learning network is the original-size image in the ImageNet validation set, making it a practical system to learn from the compressed domain. As a comparison, the conventional method reconstructs the image and then resizes it to the standard input size (224×224) to a ResNet-50 or a MobileNetV2.

As stated in Section 3.2, our $ClsNet(z)$ is a ResNet-41 (a subnet of ResNet-50) and MobileNetV2-45 (a subnet of MobileNetV2). We can see from Table 2 that for low compression ratio (e.g., bpp=0.72 or 0.49), the proposed methods can be comparative with the top-1 accuracy of a conventional learn-after-reconstruction method (on average, -0.51% using the ResNet backbone and +0.13% using the MobileNetV2 backbone). On the other hand, for the very high compression ratio (e.g., bpp=0.098), the proposed methods outperform the top-1 accuracy of the conventional methods by +3.29% using a ResNet backbone and +2.35% using a MobileNetV2 backbone, respectively. Note that the proposed methods achieve competitive accuracy and avoid large amount of computation during reconstruction.

4.3. Training with variable-size input

In order to show the advantage of training with variable-size input, we compare the results of the variable-size training with the fixed-size training in Table 3. The fixed size is chosen to be 224 and 448, respectively, corresponding to the minimum and maximum size for the variable-size input training. Table 3 shows the validation accuracy of ImageNet when the images are fed into the compression-learning network with their original size. For $LearnNet(z)$ being a

Table 3. Classification accuracy comparison between fixed-size input training and variable-size input training. All experiments are performed with the original-size images in ImageNet.

bpp	Train Input Size	ResNet-41	MobileNetV2-45
0.72	fixed-224	70.23	68.90
	fixed-448	71.52	69.86
	variable(Ours)	**74.08**	**70.44**
0.49	fixed-224	69.81	65.04
	fixed-448	70.26	66.90
	variable(Ours)	**73.42**	**70.08**
0.30	fixed-224	68.73	64.33
	fixed-448	69.04	65.0
	variable(Ours)	**73.32**	**69.41**
0.098	fixed-224	66.15	59.30
	fixed-448	66.90	60.74
	variable(Ours)	**70.37**	**64.81**
avg.	Ours over fixed-224	4.07	4.29
	Ours over fixed-448	3.36	3.06

ResNet-41, we observe that variable-size input training improves the top-1 accuracy by 4.07% over training with a fixed size 224 and by 3.36% over training with a fixed size 448, respectively. The improvement is averaged over four bpps. Similar top-1 accuracy improvement of 4.29% and 3.06% are respectively observed for $LearnNet(z)$ being a MobileNetV2-45.

Comparing the fixed-224 and fixed-448 from Table 3, it is observed that fixing the input size to be 448 is slightly more robust to the variable-size input inference. Also note that the complexity of the variable-size training lies between the fixed-224 and fixed-448 training while achieving much better accuracy.

4.4. Learning from a subset of compressed feature map

Since variance of a signal usually represents the amount of information it carry, we statically select 32 and 16 feature maps (out of 256 of the compression network introduced in Section 3.2) with largest variance and train two separate ResNet-41 as the learning networks. Note that the number of input channels to the ResNet-41 is adjusted to 32 and 16, respectively. Table 4 shows the classification accuracy of the ImageNet validation set by learning from the subset of the features. Compared with learning with all 256 feature maps, it is observed that we can almost maintain accuracy by using only 1/8 of the feature maps. This shows that the computation and transmission may be further reduced. At high compression ratio (e.g., 0.098 bpp), learning from a subset of features may even outperform learning from all features. The reason might be that those features with low variance have contributing signals to the reconstruction task, but they are noise to the learning task.

Table 4. Classification accuracy of statically selected subsets of compressed features.

bpp	# features	Top 1(%)	Top 5(%)
0.72	256	**74.08**	91.67
	32	73.87	**91.93**
	16	71.84	90.47
0.49	256	73.42	**91.51**
	32	**73.44**	91.45
	16	70.66	89.60
0.30	256	**73.32**	**91.23**
	32	72.81	90.70
	16	71.02	89.76
0.098	256	70.37	89.49
	32	**72.93**	**91.0**
	16	71.74	90.38

Table 5. Accuracy comparison between ours and cResNet-39/51 [27] on different compression ratios, using a ResNet architecture.

Network	bpp	Top 1(%)	Top 5(%)	mIoU(%)
cResNet-39	0.64	67.17	87.47	61.85
	0.33	64.14	85.46	60.78
	0.098	54.31	77.65	53.51
cResNet-51	0.64	67.68	87.85	62.86
	0.33	64.78	85.87	61.12
	0.098	55.18	78.20	54.62
ResNet-41 (**Ours**)	0.72	71.71	90.53	68.44
	0.49	71.12	89.93	65.44
	0.30	69.29	88.55	64.50
	0.098	67.72	87.88	62.87

Table 6. Accuracy comparison between ours and [27] on different compression ratios, using a MobilenetV2 architecture.

Network	bpp	Top 1(%)	Top 5(%)
MobileNet-45 (**Ours**)	0.72	69.05	91.74
	0.49	67.42	88.46
	0.30	65.08	86.68
	0.098	59.91	82.90

4.5. Comparison with the existing work

[27] is the state-of-the-art work we are aware of to learn from a CNN-based compressed domain but it only provides compression-reconstruction/learning network with fixed-size input. In this section, we compare our results to [27]. For a fair comparison, we follow the setting in [27], using the fixed input size with spatial resolution 224×224 for classification on ImageNet and the fixed size of 320×320 for semantic segmentation on PASCAL VOC

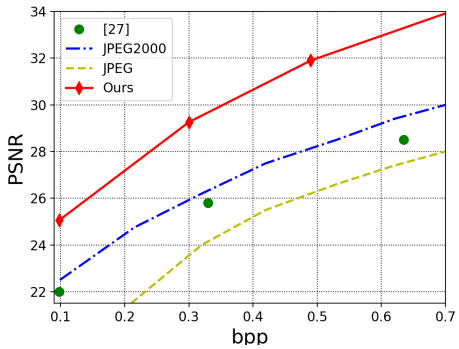


Figure 3. Comparison of reconstructed image quality, ImageNet validation set.

2012. Reconstruction quality is also measured on the ImageNet validation set (as in [27]).

Table 5 compares the (bpp, accuracy) of the classification and segmentation tasks between our proposed compression-learning networks and [27]. Compared to $(0.64, 0.33, 0.098)$ bpps in [27], the compression efficiency of our system is slightly better with $(0.49, 0.30, 0.098)$ bpps. The top-1 classification accuracy is increased from $(67.68\%, 64.78\%, 55.18\%)$ to $(71.12\%, 69.29\%, 67.72\%)$ and the mIoU is increased from $(62.86\%, 61.12\%, 54.62\%)$ to $(65.44\%, 64.50\%, 62.87\%)$. The average accuracy improvement is 6.83% for the image classification and 4.74% for the semantic segmentation. Note that our ResNet-41 is shallower than the ResNet-51 in [27]. Table 6 shows the accuracy of classifications from the compressed domain with a MobileNetV2-45. It can be seen that the accuracy is still improved over [27] by the much simpler MobileNet architecture.

Fig. 3 compares the bpps and reconstruction quality. With the same slightly reduced bpps, the average improvements of PSNR over [27] is 3.31 dB. It is also observed that our proposed compression-reconstruction network uniformly outperforms JPEG and JPEG2000.

4.6. Learning from the compressed domain for large images

In order to demonstrate the proposed methods on larger images in the real-world, we train a ResNet-49 (a ResNet-50 with the first stride-2 convolutional and max pooling layers removed) as the learning network with either 1) fixed-size input (896) and 2) variable-size input (randomly chosen between $[672, 1120]$). Table 7 shows that the proposed system is capable of learning compressed features of large images and outperforms the fixed-size training by 3.23%. This argument can be fully justified when a high resolution (e.g., over 1080p) image dataset is available in the future.

Table 7. Classification accuracy of fixed-size input training and variable-size input training. Validated on the $3\times$ up-scaled images in the ImageNet validation set.

bpp	Train Input Size	Top 1(%)	Top 5(%)
0.49	fixed-896	75.28	92.25
	Ours-variable	**76.26**	**93.04**
0.30	fixed-896	75.47	92.69
	Ours-variable	**76.37**	**93.31**
0.098	fixed-896	75.33	92.80
	Ours-variable	**76.26**	**93.18**

5. Conclusion

In this work, we propose an end-to-end system of learning from the compressed domain. Since the compression network requires variable-size input images, the corresponding compression-learning system has to accept them as well. We introduce a training method that resizes the training samples into different sizes across mini-batches and significantly improves the validation accuracy when the real-world images with a variety of sizes are fed into the compression-learning system. We also present a model-free entropy estimation method by Gaussian approximation. Experiments show that the proposed method (learning from the compressed domain) has competitive or even superior accuracy to learning from the reconstructed images, while it avoids huge amount of computation complexity during reconstruction. We also observe that the proposed system has simultaneously improved the compression ratio (bpp), reconstruction quality (PSNR), and learning accuracy of image classification and segmentation over the state-of-the-art results.

References

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

[2] Mohammad Akbari, Jie Liang, and Jingning Han. Dsslic: Deep semantic segmentation-based layered image compression. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2042–2046. IEEE, 2019.

[3] Johannes Ballé, Valero Laparra, and Eero P Simoncelli. End-to-end optimized image compression. In *International Conference on Learning Representations*, 2017.

[4] Fabrice Bellard. Bpg image format. 2014.

[5] Lahiru D Chamain, Fabien Racapé, Jean Bégaint, Akshay Pushparaja, and Simon Feltman. End-to-end optimized image compression for machines, a study. In *2021 Data Compression Conference (DCC)*, pages 163–172. IEEE, 2021.

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

[7] T. Cover and J. Thomas. *Elements of Information Theory*. Wiley-Interscience New York, NY, USA, 1991.

[8] Benjamin Deguerre, Clément Chatelain, and Gilles Gasso. Fast object detection in compressed jpeg images. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 333–338. IEEE, 2019.

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[10] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[11] Dan Fu and Gabriel Guimaraes. Using compression to speed up image classification in artificial neural networks, 2016.

[12] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] Tianyu He, Simeng Sun, Zongyu Guo, and Zhibo Chen. Beyond coding: Detection-driven image compression with semantically structured bit-stream. In *2019 Picture Coding Symposium (PCS)*, pages 1–5. IEEE, 2019.

[16] Weixiang Hong, Zhenzhen Wang, Ming Yang, and Junsong Yuan. Conditional generative adversarial network for structured domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1335–1344, 2018.

[17] Nick Johnston, Damien Vincent, David Minnen, Michele Covell, Saurabh Singh, Troy Chinen, Sung Jin Hwang, Joel Shor, and George Toderici. Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4385–4393, 2018.

[18] Eastman Kodak. Kodak lossless true color image suite (photocd pcd0992). 1993.

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

[20] Yash Patel, Srikar Appalaraju, and R Manmatha. Saliency driven perceptual image compression. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 227–236, 2021.

[21] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[22] Shibani Santurkar, David Budden, and Nir Shavit. Generative compression. In *2018 Picture Coding Symposium (PCS)*, pages 258–262. IEEE, 2018.

[23] Athanassios Skodras, Charilaos Christopoulos, and Touradj Ebrahimi. The jpeg 2000 still image compression standard. *IEEE Signal processing magazine*, 18(5):36–58, 2001.

[24] Myungseo Song, Jinyoung Choi, and Bohyung Han. Variable-rate deep image compression through spatially-adaptive feature transform. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2380–2389, October 2021.

[25] George Toderici, Sean M O'Malley, Sung Jin Hwang, Damien Vincent, David Minnen, Shumeet Baluja, Michele Covell, and Rahul Sukthankar. Variable rate image compression with recurrent neural networks. In *International Conference on Learning Representations*, 2016.

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

[27] Robert Torfason, Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, and Luc Van Gool. Towards image understanding from deep compression without decoding. 2018.

[28] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015.

[29] Gregory K Wallace. The jpeg still picture compression standard. *IEEE transactions on consumer electronics*, 38(1):xviii–xxxiv, 1992.

[30] Cheng Wang, Yifei Han, and Weidong Wang. An end-to-end deep learning image compression framework based on semantic analysis. *Applied Sciences*, 9(17):3580, 2019.

[31] Zhenzhen Wang, Weixiang Hong, Yap-Peng Tan, and Junsong Yuan. Pruning 3d filters for accelerating 3d convnets. *IEEE Transactions on Multimedia*, 2019.

[32] Kai Xu, Minghai Qin, Fei Sun, Yuhao Wang, Yen-kuang Chen, and Fengbo Ren. Learning in the frequency domain. In *CVPR 2020*, 2020.