Quality Level Prediction of Image Compression using Block-wise Confidence-aware CNN

Kyuwon Kim
q1.kim@samsung.com
Chulju Yang
chulju.yang@samsung.com

Advanced Multimedia Lab.
Samsung Electronics
Suwon, Republic of Korea

Abstract

Almost all images are compressed to be transferred or stored, exhibiting various visual artifacts. For this reason, identifying the compression quality level with only visual cues is the starting point to enhance the image quality. This paper introduces a compression quality prediction method, named Q1Net, which yields a single quality level with over 99%-accuracy in a matter of milliseconds on mobile devices regardless of the image resolution. This real-time and high-accurate performance is attributed to the observation that most image compression methods are based upon transform coding on small blocks of different characteristics. To separately investigate and exploit the distinct visual deformations induced by one transform coding, our method measures the compression quality level on various image patches containing a basic coding block and its neighboring pixels. Our approach then elaborately selects promising candidate patches that can indicate the compression quality reliably through CNN-based statistical confidence estimation. In order to make a final decision, the proposed method fuses the prediction results from a selected number of input patches, which makes it scalable and operable on mobile devices with varying computational capabilities. According to the extensive experiments on the DIV2K dataset and an off-the-shelf smartphone, our block-wise confidence-aware Q1Net achieves better performance in compression quality prediction than other well-known CNN-based methods in terms of speed and accuracy.

1 Introduction

Many online service providers, such as social media platforms and instant messaging services, present numerous images to users. As such, they should reduce cloud storage costs by storing image data in as little space as possible. Also, in order for users to view images without delay, the media transmission speed should be increased. One obvious solution to kill two birds with one stone is to apply transform-based image compression techniques [4, 6, 15, 16, 17, 18, 22]. Table 1 shows various JPEG [19] quality levels adopted by dif-

Table 1: JPEG quality levels used by different services

	Google Photos	Facebook	Pinterest	Kakao Talk [27]
Level	60–85	88–92	80	90, 95

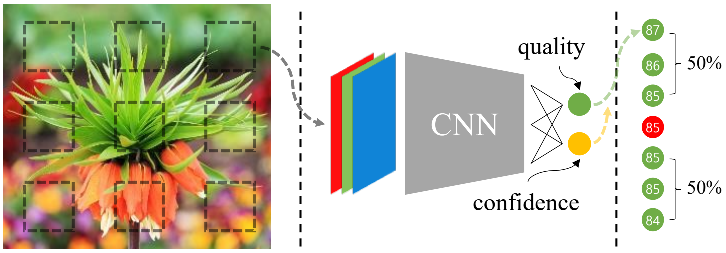


Figure 1: Illustrative toy example of predicting the compression quality level using block-wise confidence-aware CNN, coined Q1Net. In this simplified example, the given compressed image is assumed to be small enough that only nine input patches can be examined across the image. Each of the input patches contains a compression coding block. To leverage the GPU parallelism, the patches are uniformly sampled and then fed to Q1Net for batch prediction, yielding nine pairs of quality and confidence. Among the nine pairs, only seven high-confidence qualities are harnessed to calculate the mean value (85) as the final quality level in this example.

ferent services to reduce storage and speed up the transmission. The values in Table 1 were measured using the proposed method in this paper. The level ranges from 0 to 100, with a higher number indicating better quality in terms of the quantization table adopted by the libjpeg-turbo library [2]. The quantization table controls the size and visual quality of the compressed image.

Unfortunately, the benefits of image compression do not come for free. Its quantization process inevitably causes various degradations such as blocking and ringing artifacts [24], even to high-definition photos captured with modern cameras. Arguably, the very first step to enhance the image quality is to determine the compression quality level reliably. Suppose a system is well aware of the compression quality of an image. In that case, it can decide whether the artifacts should be reduced or not, avoiding as many heavy tasks of image-to-image translation as possible. When an enhancement task is found to be required, the predicted quality level again can serve as a significant hint that allows the following method of artifact reduction to pinpoint and eliminate the prevalent patterns of artifacts particularly occurring at that level. The better an image is understood, the more accurately it can be enhanced.

Despite its importance, there have not been many studies on quality level prediction of image compression. Chances are that the compression quality level is thought to be easily or naturally revealed during the decompression phase. However, an essential requirement for sensible image quality prediction is to utilize only image pixels, excluding any metadata such as quantization tables. In many practical situations, a file header containing original metadata is not available due to crop operations, format conversions (to PNG or BMP), or any other modifications. Uchida *et al.* [28] proposed a CNN-based estimation method that generates a JPEG quality level map for fake image detection. Recently, Kim *et al.* [11] proposed an integrated method of compression artifacts reduction in which a pixel-wise quality level map controls latter artifact removal networks. The previous studies share a similar insight with ours that compression quality prediction is essential to deal with a variety of alterations to compressed images in the real world. However, rather than resorting to the quality level map by Uchida *et al.* [28] and Kim *et al.* [11], our method focuses on identifying a representative and reliable compression quality level for an efficient image enhancement. In fact, the same quantization rule is applied to every block of an image for JPEG compression. Even though visual artifacts vary over blocks due to each block's characteristics, each block basically undergoes the same amount of quantization deformation.

To overcome the loss of prediction accuracy caused by spatially varying artifacts, our method exploits the concept of confidence. In a worst-case scenario, if a particular image region contains only low-frequency components not affected by different quantization schemes at a different quality level, the training and prediction of a machine learning model become unreliable. Unfortunately, previous studies [11, 28] do not actively incorporate strategies to cope with these challenges. In this paper, we design our method of compression quality prediction to be highly selective so that it can go with strongly informative and reliable blocks. Naturally, it enables precise image analysis while dramatically reducing the amount of computation. Nowadays, images with a resolution of more than 12 megapixels are readily available on smartphones, so making a quality level map by applying CNN to all image blocks would require excessive time and energy. If all pixels in an image are degraded by the same mechanisms, yielding a single representative number that explains the reason for degradation would be more efficient. To this end, our method predicts a single compression quality level by an end-to-end training of a model that not only inspects the compression quality of uniformly sampled blocks but also estimates their confidence scores. Especially, we establish a formula for statistically approximating the prediction confidence, applying this to the loss function and the output layer of CNN. Many studies [5, 7, 8, 11, 14] have shown that an inverse CNN model of frequency quantization can be trained to reasonably pick and remove distinctly noticeable artifacts exhibiting on a specific compression quality level. This means that Q1Net can play an essential role in compression artifacts removal by providing accurate quality information to the existing methods.

Our contributions can be summarized as follows:

1. We present a quick and accurate block-wise prediction method of compression quality level that is able to account for the nature of block transform coding. The patch size for the CNN input is set to factor in a coding block necessarily and sufficiently. Generally, the smaller a receptive field is, the faster a CNN inference is.

2. In order to increase the accuracy when consolidating the prediction results for each block, our CNN architecture, called Q1Net, is constructed to additionally compute the confidence indicating which prediction results are more trustworthy. The proposed architecture is trained in an end-to-end fashion.

3. Our method can adapt to different devices with different computing capabilities. This is because the number of blocks being sampled can be scaled according to the device's computational power.

Fig. 1 shows the overall inference flow of the proposed approach.

2 Proposed Method

2.1 Motivation of Confidence Estimation

Transform-based image encoders perform the transform and quantization block-wise, with a global quality parameter controlling how much to compress an image. Although unlike JPEG, some video encoding technologies adopt the adaptive block quantization scheme, the quality variation still centers around the global target quality. Thus, theoretically, investigating just one coding block and its neighboring boundaries will give some insight into the compression quality of the entire image. This task of block investigation can be naturally

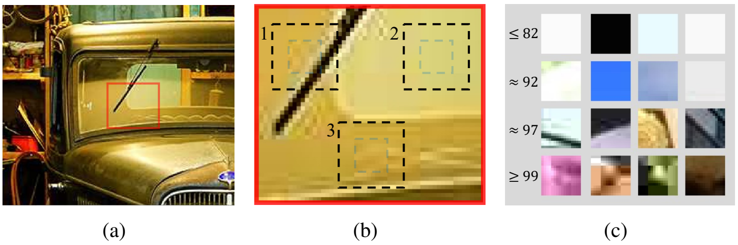


Figure 2: Illustrative example to show the motivation of this work: (a) Compressed image with the quality level 50. (b) Zoomed-in view of (a). There are three black-dotted 16×16 pixel rectangles, representing candidate patches being fed to CNN. This input patch is merely an extended block of an inner-center, blue-dotted rectangle representing the 8×8 JPEG coding block. This size enlargement is to consider boundary parts between neighboring blocks. Among b1, b2, and b3, which image patch stands out in terms of the possibility to tell the true compression quality? Our research answers this block selection question using confidence estimation. (c) Real examples of input patches and their confidence scores measured by Q1Net. The lower the row, the higher the confidence.

recast as a regression problem. However, a typical regression model has the drawback of having to answer no matter what the input is. In other words, a traditional regression task does not consider whether inputs have distinctive features for reliable outputs or not. Here is an example: Fig. 2 shows an enlarged image compressed with the quality level 50. Among image patches of (b1), (b2), and (b3) in Fig. 2, which one is least likely to report the true compression quality? Here, an image patch represents an extended block of an 8×8 JPEG coding block. The uncompressed version of the solid color patch shown in Fig. 2(b2) is more likely composed of similar color pixels containing mainly the direct current (DC) component in the discrete cosine transform (DCT) domain. Therefore, it would generate similar compressed block outputs for even other compression quality levels, confusing the regression network. Results from this kind of input should be appropriately excluded. This is where confidence estimation for quality prediction comes into play.

Using prior knowledge of image compression and edge detection, one might conjecture that having sufficient image gradient is one of the criteria to become a viable input candidate. However, mere edge strength calculation cannot explain all the complicated cases when it comes to computing the confidence of quality predictions. A great deal of time and effort will also be required for researchers to manually come up with other types of criteria to identify a block as eligible or ineligible for quality prediction. Therefore, it is necessary to pick and choose blocks that can reliably indicate the compression quality through machine learning and statistical analysis. In this way, we can increase the possibility that confidence can be measured based on even color and edge pattern as well as edge strength.

2.2 Confidence-aware CNN

For the CNN training, the ground-truth label of the compression quality can be easily provided by compressing raw images with a certain level. However, there is no ground-truth confidence score for quality prediction. This section discusses how to practically estimate the confidence of predicted quality levels for each candidate patch.

Assuming that a trained CNN model $G(\cdot)$ can accurately predict the compression quality $\hat{q} = G(x)$ of a given input patch x, the true confidence observation c_i of the given block x_i

can be approximated as follows:

$$c_i = m - |q_i - \hat{q}_i|, \tag{1}$$

where $\hat{q}_i$ is the predicted quality of a given i^{th} input x_i and q_i is the true quality, and m is the maximum quality level. For example, m is set to 100 for libjpeg-turbo [2] and 51 for the FFmpeg [1] constant rate factor.

Therefore, using the quality prediction network $G(\cdot)$ and the loss function (1), we can train another confidence estimator,

$$\hat{c} = H(x), \tag{2}$$

where $\hat{c}$ is the estimated confidence for the input x. When L_1 loss is employed for training of $H(\cdot)$, the network will end up estimating a median number z that has the smallest average deviation from the confidence measurements for each type of the candidates:

$$\underset{z}{\operatorname{argmin}}\, \mathbb{E}_c\{|z - c|\}. \tag{3}$$

However, this method of using two networks, $G(\cdot)$ and $H(\cdot)$, leads to inefficiency in that it requires two learning stages and two inferences. Since the two tasks of quality prediction and confidence estimation are expected to share many common features, we come up with more efficient and viable $F(\cdot)$ that can output both quality and confidence in one network as follows:

$$\{\hat{q},\, \hat{c}\} = \{F(x)_1, F(x)_2\} = F(x), \tag{4}$$

where $F(x)_1$ and $F(x)_2$ are the first and second output units of $F(x)$, respectively. As a result, the following loss function is adopted:

$$\underset{\theta}{\operatorname{argmin}}\, \mathbb{E}_{(x,q)}\{|F_\theta(x)_1 - q| + \lambda\, |F_\theta(x)_2 - c|\}, \tag{5}$$

where F_θ is a confidence-aware CNN with a parameter set θ. We experimentally set the parameter λ, which reflects the relative importance of confidence as opposed to quality, to 0.5. However, we found that other values of λ (such as 0.25 or 1) still achieve comparable results, as indicated in Table 3.

The problem with this one-stage network learning is that the proposed CNN model $F(\cdot)$ is immature at the initial stage of training. Consequently, the quality prediction ability of $F(\cdot)$ is also unstable, resulting in erratic confidence measurements according to (1). One strategy to solve this problem is to train $G(\cdot)$ first and then acquire $H(\cdot)$. We empirically found that, however, even without the two-stage CNN learning, (5) converges to an optimum in the end, establishing true end-to-end learning. Fig. 2(c) presents examples of estimated confidence scores for some compressed input patches in the DIV2K dataset [3]. Interestingly, Q1Net is not explicitly trained to assign higher confidence to the patches with stronger gradients, but it naturally infers this fact through (1). The overall design of the architecture is illustrated in Fig. 3, where the numbers of filter channels and output units are determined by the parameters k and α, respectively. Note that pooling layers are not used in our architecture since the receptive field is small, as discussed in the following section.

2.3 Scalable and Real-time Block-based Approach

A realistic and practical approach should support not only the latest devices but also those with limited computational capabilities. Q1Net is designed to accept a variable number of

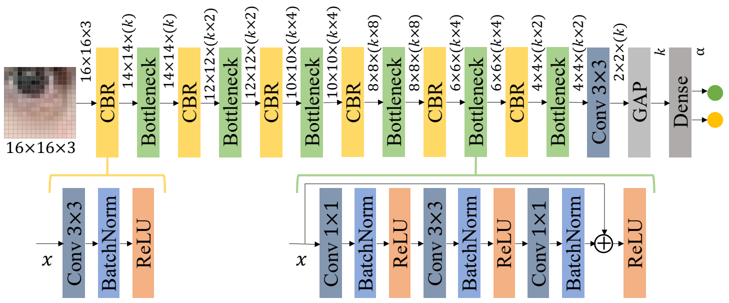


Figure 3: Network architecture of the proposed confidence-aware Q1Net: GAP, Conv, and Dense mean global average pooling, convolution layer, and fully connected layer, respectively. The Bottleneck block is basically a residual block [10]. The output shapes of each layer are specified in the format of 'height × width × (number of channels)'. In the baseline network architecture, the parameter k that controls the number of channels is 8, and the number of output units α is 2.

blocks and adapt to devices with different computing power. That is, by default, a total of 256 (16×16) blocks are uniformly sampled. However, on low-end mobile devices, a smaller number of blocks can be extracted from the image to make a balance between accuracy and computational cost.

As shown in the example inputs of Fig. 2(b), Fig. 2(c), and Fig. 3, our method tries to adopt the minimal size of the receptive field. In the case of JPEG, the patch size is set to 16×16, which is necessary and sufficient for taking into account a basic 8×8 coding block and the boundaries with its surrounding neighbors. Due to the resulting low computational complexity, it is possible to maximize GPU parallelism by batch prediction of multiple input patches at once. The confidence threshold τ, which decides whether a prediction result of each block will contribute to the final calculation, is obtained by grid search on a validation dataset. The final quality level is determined by calculating the median from the selected prediction results.

3 Experimental Results

3.1 Experimental Settings

We compare Q1Net with MobileNetV2 [21], EfficientNet [26], ShuffleNet [31], LCNN [13, 29], JQE [28], and Q-EST [11]. For EfficientNet, the most compact model B0 is used. For ShuffleNet, we set the bottleneck ratio to 1 : 2 and the number of groups to 8 [31]. LCNN is an enhanced version of the Light CNN architecture by Lavrentyeva *et al.* [29]. The comparison methods of JQE and Q-EST are implemented to evaluate the quality map-based methods by Uchida *et al.* and Kim *et al.*, respectively, discussed in Section 1. The other experimental settings are the following:

Datasets and compression method. All methods are trained and evaluated on 700 training, 100 validation, and 100 test images from 900 high-resolution (HR) images of the DIV2K dataset [3]. The DIV2K dataset is chosen for this research because it retains rich details and no compression artifacts stored in the PNG lossless image format [3]. We compressed each image with libjpeg-turbo [2] [1] by changing the compression quality level from 1 to 100.

[1]Experimental results using H.264 [20] and FFmpeg [1] are presented in the supplementary material.

Accordingly for evaluation, a total of 10,000 compressed images are iterated over by each method.

Training settings. Image patches with the size of 16×16, 224×224, and 256×256 pixels are randomly sampled from the training datasets. Since there is already a great deal of small patches in an image of DIV2K, data augmentation is not adopted. The proposed CNN is implemented using TensorFlow on a Titan Xp GPU. We trained our models by Adam optimizer [12] with the default TensorFlow parameters of $\beta_1 = 0,9$, $\beta_2 = 0.999, \varepsilon = 10^{-7}$. The mini-batch size is set to a large number of 2048 thanks to the small number of model parameters as indicated in Table 2. The initial learning rate is set to 10^{-3}. The confidence threshold τ is set to 96 by grid search on 100 validation images split from the training dataset.

Evaluation environment and metrics. The inference latency is measured in TensorFlow Lite on the Qualcomm Adreno 660 Mobile GPU of Samsung Galaxy S21. The prediction performance is evaluated by MAE (Mean Absolute Error) and SDE (Standard Deviation of Error).

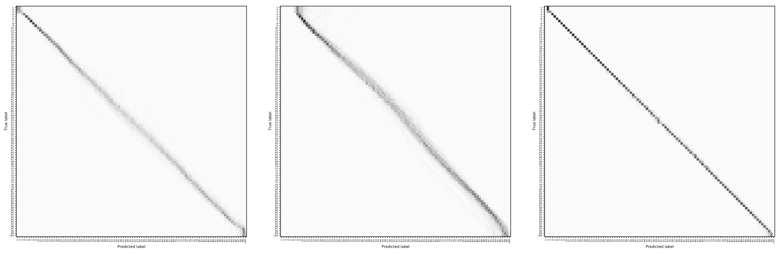
3.2 Results and Analysis

For a fair comparison, all evaluated methods are set up to process a similar amount of image pixels. The input sizes of all methods are 256×256 except for MobileNetV2 and ShuffleNet, whose input sizes are 224×224. Q1Net also extracts 16×16 patches of 16×16 pixels, investigating the similar input space with comparison methods. For evaluating the conventional networks, input patches are extracted from the central region of images according to popular belief that the center location often holds rich visual information. Indeed, the DIV2K test dataset (0801.png to 0900.png) retains rich visual textures in the central part.

Table 2: Performance comparison on DIV2K [1] with the input size of about 256×256

Method	MobileNetV2 [21]	EfficientNet [26]	ShuffleNet [31]	LCNN [13, 29]	JQE [28]	JQE-like [28]	Q-EST [11]	**Q1Net** (Baseline)
MAE	1.16	2.24	2.04	1.76	4.06	1.60	1.22	**0.40**
SDE	1.43	2.29	2.29	1.61	2.46	1.49	1.34	**0.39**
Time	7 ms	12 ms	11 ms	12 ms	3.2 s	**6 ms**	**6 ms**	10 ms
#Params	2.32 M	4.13 M	3.71 M	1.47 M	188 K	217 K	213 K	**125 K**

Performance comparison on DIV2K dataset. Table 2 compares each method's MAE, SDE, latency, and model size, summarizing the main experimental result with about 256×256 input pixels. In this task of predicting the quality level from 1 to 100, MobileNetV2 achieves better prediction performance with an MAE of 1.16 with only 7 ms processing time compared to other well-known real-time networks. JQE is a quality map-generating model trained and made publicly available by Uchida *et al.* [28]. This kind of quality map-based method predicts the compression level by averaging the entire quality map. However, we observed that the accuracy is reduced due to potential outlier blocks, as indicated in the evaluation result of JQE. In addition, JQE is much slower than other methods due to the cost of constructing a map of 256×256 via convolution operations. Therefore, rather than comparing map-based algorithms by Uchida *et al.* [28] and Kim *et al.* [11] as they are, we built JQE-like and Q-EST regression networks, respectively, mainly using dilated and standard convolution layers following the design principles of each paper [11, 28]. JQE-like and Q-EST show comparable performance to MobileNetV2 with a small number of parameters through max-pooling and average-pooling [30]. Compared with other methods, the baseline of Q1Net ($\alpha = 2$, $k = 8$, $\lambda = 0.5$) shows state-of-the-art performance in terms of



(a) MobileNetV2 (b) JQE (c) Q1Net (baseline)

Figure 4: Comparison of confusion matrices [25] on 10,000 compressed images of 100 categories (quality levels): the confusion matrix (c) obtained by Q1Net presents higher diagonal values than the matrices obtained by MobileNetV2 [21] and JQE [28]. It is recommended to zoom in this figure in an electronic copy of this paper.

MAE and SDE. What is more, this is achieved with fewer model parameters. Q1Net's latency is slightly inferior to MobileNetV2, JQE-like, and Q-EST, but it can be further improved by variably changing the number of input patches as suggested in Table 4.

Comparison of confusion matrices. To verify the performance of Q1Net as a classifier, we rounded off the prediction results to derive a confusion matrix [25] for 100 categories and compared it with the results of MobileNetV2 and JQE. In the confusion matrix of Fig. 4, the diagonal elements represent the number of correct classifications, while off-diagonals are incorrect predictions. The diagonal line of Fig. 4(c) generated by Q1Net is thicker than Fig. 4(a) and 4(b), indicating higher prediction accuracy.

3.3 Ablation Study: Contribution of Each Innovation in Q1Net

We conduct an ablation study to demonstrate the impact of each design choice of Q1Net. The results with 16×16 input patches are shown in Table 3.

Table 3: Ablation study of block-wise CNN with different settings on DIV2K [3]

Method	Setting						Result			
	α	λ	k	Loss	BB	Confidence	MAE	SDE	Time	#Params
Q1-Regressor	1		8	MAE	✓		1.28	5.79	9 ms	125 K
Q1-Classifier	100		8	CCE	✓		1.68	7.69	10 ms	125 K
Q1-LambdaQ	2	0.25	8	(5)	✓	✓	0.39	0.38	10 ms	125 K
Q1-Lambda1	2	1.0	8	(5)	✓	✓	0.36	0.39	10 ms	125 K
Q1-Light	2	0.5	6	(5)	✓	✓	0.44	0.44	7 ms	71 K
Q1-Heavy	2	0.5	12	(5)	✓	✓	**0.26**	**0.29**	19 ms	280 K
Q1-No-BB	2	0.5	8	(5)		✓	0.87	0.89	**5 ms**	**49 K**
Q1-Sobel	1		8	MAE	✓	✓	0.46	0.50	14 ms	125 K
Q1-Softmax	100		8	CCE	✓	✓	0.52	0.67	10 ms	125 K
Baseline	2	0.5	8	(5)	✓	✓	0.40	0.39	10 ms	125 K

We investigated the impact of confidence-aware loss (5) by replacing it with different loss functions. A simple regression model, Q1-Regressor, trained with the single output unit achieves comparable performance to MobileNetV2 [21] and Q-EST [11] in terms of MAE. However, its high SDE of 5.79 indicates that its prediction model is unreliable for certain images and suffers from outliers. We also train a standard CNN classification model, Q1-Classifier, using softmax activation and categorical cross-entropy (CCE) loss with 100 classes. Q1-Classifier achieves an MAE of 1.68 and an SDE of 7.69. Considering its higher SDE, the classification model suffers from weaker outliers than the regression model in our experiments. Unlike MAE loss, CCE loss does not consider the amount of error for the wrong classification during training. For example, the classification model treats two different predictions with the error of 1 and 99 just as the same incorrect classification. Thus, the classification model was hardly trained in our experiments, converging to a local minimum. We derived the CCE result in Table 3 by leveraging transfer learning and fine-tuning that incrementally increases the network's output units (α), extending it from easy binary classification to complex 100-class classification. On the other hand, other models with MAE loss and (5) were trained without multi-stage transfer learning because the error proportional to the wrong amount can be directly propagated backward during training.

In Table 3, except for Q1-Regressor and Q1-Classifier, all others are results to which the concept of confidence is applied. Note that once confidence measurements are introduced, MAE and SDE drop dramatically below 1.0. When λ in (5) is changed from 0.5 (baseline) to 0.25 or 1, the performance remains similar. Since the confidence-aware CNN F_θ is trained with plenty of raw and compressed image pairs in our experiments, both terms in (5) are eventually minimized regardless of λ. We also measure the performance variation according to the model size. As expected, the larger k, the lower the MAE and SDE. When k is 12, the SDE drops to 0.26. Q1-No-BB is a model in which all BottleNeck blocks (BBs) [10] are removed from the baseline model in Fig. 3. It is observed that BBs reduce MAE and SDE to less than 50%. Q1-Sobel is basically the same model as Q1-Regressor, but it additionally measures the confidence of input patches by Sobel operator-based edge detection [23] to eliminate unreliable predictions. That is, Q1-Sobel does not use the input patches with weaker gradients to calculate the final quality level. Since edge detection can readily determine the existence of high-frequency components, Q1-Sobel is highly effective in eliminating outliers, achieving comparable performance to the baseline of Q1Net. However, Q1-Sobel takes 4 ms more than the baseline due to the additional convolution operations on the CPU. Furthermore, using edge-detection is similar to extracting hand-made features, hardly generalizing to other image analysis tasks. Q1-Softmax uses softmax scores from the softmax layer of Q1-Classifier to reject outliers. We obtained the lowest MAE and SDE for Q1-Softmax by thresholding the softmax scores using a grid search. Q1-Softmax has the benefit of conveniently measuring confidence. However, it achieves higher MAE and SDE than the baseline of Q1Net, possibly due to the aforementioned difficulties of model training. Also, different from thresholding of softmax probabilities, the second loss term in (5) of Q1Net can be applied to regression problems. Based on our ablation study in this section, we argue that confidence estimation by Q1Net allows for robust predictions of compression quality by reducing bias from outliers.

Table 4: Performance evaluation of the baseline model at different block numbers

#Blocks	4×4	8×8	12×12	16×16	20×20	24×24
MAE	0.82	0.50	0.42	0.40	0.37	0.36
SDE	0.86	0.47	0.42	0.39	0.38	0.37
Time (ms)	2	4	6	10	15	21

Q1Net evaluation at different block numbers. Table 4 shows that the number of input patches of Q1Net can be adjusted to reduce latency further while maintaining predictive performance. It is important to note that using even 4×4 blocks achieves better performance than other compared methods in terms of accuracy and speed.

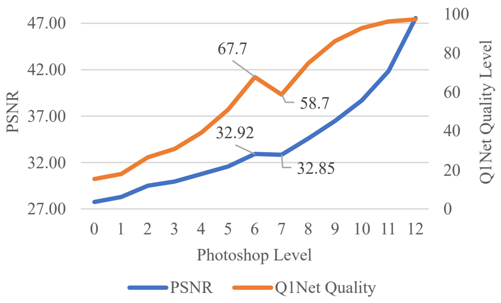


Figure 5: Level prediction of Photoshop-generated JPEGs for 0804.png in DIV2K [3]

3.4 Universal Compression Quality Level

Different compression encoders use different quantization tables and chroma subsampling schemes. Therefore, we investigated how the Q1Net classifier trained using libjpeg-turbo classifies the images compressed by the JPEG encoder of Photoshop at each quality level. Photoshop supports a total of 13 levels of JPEG compression from 0 to 12. For this experiment, images in the DIV2K validation dataset were compressed by each compression quality. As a representative result shown in Fig. 5, the compression quality levels of the two different encoders are found to be correlated to a great extent. However, between Photoshop quality level 6 and level 7, the prediction results of Q1Net are reversed, which contrasts with our expectations. It turns out that Photoshop quality level 7 and above do not perform chroma subsampling, so Photoshop quality level 7 quantizes the luminance more aggressively than level 6 does [9]. Since the human visual system is more sensitive to luminance than chrominance components [20], it makes sense for Q1Net to estimate level 7 as lower quality than level 6. It is also observed in Fig 5 that peak signal-to-noise ratio (PSNR) of a compressed image at level 7 is less than that at level 6. Based on the findings of this experiment, we argue that if the compression level range of an encoder is fine-grained enough to represent most possible variations, then the corresponding level can be found in the compression output of another encoder.

4 Conclusions

The first order of business in true automation of image enhancement is to evaluate the image quality accurately and rapidly. If the presence and type of image degradation are determined, then a system may inform users of the identified issues. Alternatively, it may suggest a new version after enhancing images through the evaluation result. As the old saying goes, *Knowledge is power* so the better we understand an image, the more wisely we can enhance it. This paper introduces Q1Net, which predicts a single compression quality level that clarifies how heavily an image is compressed. To this end, our method evaluates uniformly sampled blocks, whose size is theoretically determined to capture the deformation by a block transformation and quantization stage. Moreover, to effectively eliminate outliers in the prediction results, Q1Net is end-to-end trained with a novel loss function that allows estimating the confidence of each quality prediction.

With various experiments on the DIV2K dataset and a commercial Samsung Galaxy S21 smartphone, the capabilities of Q1Net were demonstrated to be accurate, real-time, explainable, and scalable. We hope that the proposed approach will serve as a solid baseline not only in the task of compression quality prediction but also in other types of quality measurement tasks such as noise level prediction and blurriness detection.

Acknowledgment

The authors would like to thank Sunok Kim, Hyungsuk Kim, Joonoo Kim, Inho Choi, Tushar Sandhan, Heekuk Lee, and Jongkyu Kim for their thoughtful suggestions.

References

[1] ffmpeg. http://ffmpeg.org/. [Online; accessed 28-May-2021].

[2] libjpeg-turbo. https://libjpeg-turbo.org. [Online; accessed 28-May-2021].

[3] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 126–135, 2017.

[4] Yochai Blau and Tomer Michaeli. Rethinking lossy compression: The rate-distortion-perception tradeoff. In *International Conference on Machine Learning*, pages 675–685. PMLR, 2019.

[5] Lukas Cavigelli, Pascal Hager, and Luca Benini. Cas-cnn: A deep convolutional neural network for image compression artifact suppression. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 752–759. IEEE, 2017.

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

[7] Chao Dong, Yubin Deng, Chen Change Loy, and Xiaoou Tang. Compression artifacts reduction by a deep convolutional network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 576–584, 2015.

[8] Leonardo Galteri, Lorenzo Seidenari, Marco Bertini, and Alberto Del Bimbo. Deep generative adversarial compression artifact removal. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4826–4835, 2017.

[9] Calvin Hass. Jpeg quality and quantization tables for digital cameras photoshop. https://www.impulseadventure.com/photo/jpeg-quantization.html, 2018. [Online; accessed 28-May-2021].

[10] 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.

[11] Yoonsik Kim, Jae Woong Soh, and Nam Ik Cho. Agarnet: adaptively gated jpeg compression artifacts removal network for a wide range quality factor. *IEEE Access*, 8: 20160–20170, 2020.

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

[13] Galina Lavrentyeva, Sergey Novoselov, Andzhukaev Tseren, Marina Volkova, Artem Gorlanov, and Alexandr Kozlov. Stc antispoofing systems for the asvspoof2019 challenge. *arXiv preprint arXiv:1904.05576*, 2019.

[14] Danial Maleki, Soheila Nadalian, Mohammad Mahdi Derakhshani, and Mohammad Amin Sadeghi. Blockcnn: A deep network for artifact removal and image compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 2555–2558, 2018.

[15] Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, and Luc Van Gool. Conditional probability models for deep image compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

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

[17] Yash Patel, Srikar Appalaraju, and R Manmatha. Hierarchical auto-regressive model for image compression incorporating object saliency and a deep perceptual loss. *arXiv preprint arXiv:2002.04988*, 2020.

[18] 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.

[19] William B Pennebaker and Joan L Mitchell. *JPEG: Still image data compression standard*. Springer Science & Business Media, 1992.

[20] Iain E Richardson. *H. 264 and MPEG-4 video compression: video coding for next-generation multimedia*. John Wiley & Sons, 2004.

[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] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[23] Irwin Sobel and Gary Feldman. A 3x3 isotropic gradient operator for image processing. *a talk at the Stanford Artificial Project in*, pages 271–272, 1968.

[24] Sima Sonawane and B.H. Pansambal. Review of artifacts in jpeg compression and reduction. *International Journal of Advanced Research in Electronics and Communication Engineering*, 6, 2017.

[25] Stephen V Stehman. Selecting and interpreting measures of thematic classification accuracy. *Remote sensing of Environment*, 62(1):77–89, 1997.

[26] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019.

[27] Hasan Tinmaz and Jin Hwa Lee. General and instructional perceptions of south korean messenger:'kakaotalk'. *International Journal of Learning and Change*, 12(2):143–168, 2020.

[28] Kazutaka Uchida, Masayuki Tanaka, and Masatoshi Okutomi. Pixelwise jpeg compression detection and quality factor estimation based on convolutional neural network. *Electronic Imaging*, 2019(11):276–1, 2019.

[29] Xiang Wu, Ran He, Zhenan Sun, and Tieniu Tan. A light cnn for deep face representation with noisy labels. *IEEE Transactions on Information Forensics and Security*, 13 (11):2884–2896, 2018.

[30] Dingjun Yu, Hanli Wang, Peiqiu Chen, and Zhihua Wei. Mixed pooling for convolutional neural networks. In *International conference on rough sets and knowledge technology*, pages 364–375. Springer, 2014.

[31] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6848–6856, 2018.