Super Resolution for QR Code Images

Takahiro Shindo
School of FSE
Waseda University
Tokyo, Japan
taka_s0265@ruri.waseda.jp

Taiju Watanabe
School of FSE
Waseda University
Tokyo, Japan
lvpurin@fuji.waseda.jp

Remina Yano
Graduate School of FSE
Waseda University
Tokyo, Japan
yano.remina@toki.waseda.jp

Marika Arimoto
Graduate School of FSE
Waseda University
Tokyo, Japan
m.arimoto@akane.waseda.jp

Miho Takahashi
Graduate School of FSE
Waseda University
Tokyo, Japan
miho.takahashi@akane.waseda.jp

Hiroshi Watanabe
Graduate School of FSE
Waseda University
Tokyo, Japan
hiroshi.watanabe@waseda.jp

2022 IEEE 11th Global Conference on Consumer Electronics (GCCE) | 978-1-6654-9232-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/GCCE56475.2022.10014154

***Abstract*—In this paper, we propose an image denoising and a super resolution method for converting unreadable low resolution QR code images into readable high resolution ones. We propose an image denoising and a super resolution method using a simple CNN-based model. Image denoising using CNN-based image generation models can be applied to various noises by training model with different levels of noise. On the other hand, denoising using the conventional image processing method can only be applied to the specific type of noise. Therefore, image denoising using CNN is considered to be superior to image processing methods in terms of generalization performance. We further implement super resolution method to convert into high resolution images. We propose QRCNN and QRGAN, simple image denoising and super resolution models for QR code images. QRCNN and QRGAN are based on SRResNet and SRGAN, respectively. However, they both have simpler structure like SRCNN. Given QR code images for the dataset, we can reduce the computational complexity and memory usage by modifying model structure.**

***Keywords*—QRCNN, QRGAN, SRResNet, SRGAN, SRCNN**

I. Introduction

Nowadays, there are a lot of opportunities for people to scan QR codes due to the rise of QR code payments and site guidance. However, we sometimes encounter scanned QR codes images that are printed blurry or taken from a distance. Our proposed methods, QRCNN and QRGAN can convert unreadable low resolution QR code images into readable high resolution ones. QRGAN is a GAN-based [1] model based on QRCNN. In order to reduce computational complexity, these models have simple model structures and only allow grayscale images for the dataset. QRCNN and QRGAN are simpler model of SRResNet and SRGAN [2], respectively. In contrast to SRResNet and SRGAN, our proposed methods can reduce memory usage by not using pretrained VGG [3] for loss calculation. Super resolution model, SRCNN [4] is widely known to have very simple model structure whose number of layers and parameters are similar to our proposed methods. We compare the performance of our methods with SRCNN and show superiority to SRCNN.

II. Related Works

A. SRCNN

The most common super resolution method based on CNN is SRCNN. This model consists of only three convolution layers and two activation functions (ReLU [5]). In SRCNN, low resolution input images are upscaled to the desired image size using bicubic interpolation [6] before input to the network. Therefore, the size of the image does not change in the neural network. The loss function of SRCNN is given by

$$l^{SR} = l^{SR}_{MSE}. \tag{1}$$

Only mean squared error (MSE) of the ground truth and the generated image is used for loss function.

B. SRGAN

The most common super resolution method based on GANs with generators and discriminators is SRGAN. The generator attempts to produce an image that is close to the ground truth. The discriminator attempts to distinguish between the ground truth and the image produced by the generator. SRGAN utilizes this principle to improve super resolution tasks. Contrary to SRCNN, SRGAN enables super resolution by upscaling the size of the image in the middle of the neural network. Therefore, the input image is smaller than the output image. The generator consists of deep layers with 16 residual blocks and skip connections. Each residual block contains two convolutional networks. The discriminator consists of 8 convolutional networks. The loss function of SRGAN is given by

$$l^{SR} = \underbrace{l^{SR}_{X}}_{\text{content loss}} + \underbrace{10^{-3} l^{SR}}_{\text{adversarial loss}}, \tag{2}$$

where the function consists of a content loss and an adversarial loss. Content loss is calculated by MSE or obtained from feature maps derived from VGG. Adversarial loss, on the other hand, results from the generator and the discriminator competing each other.

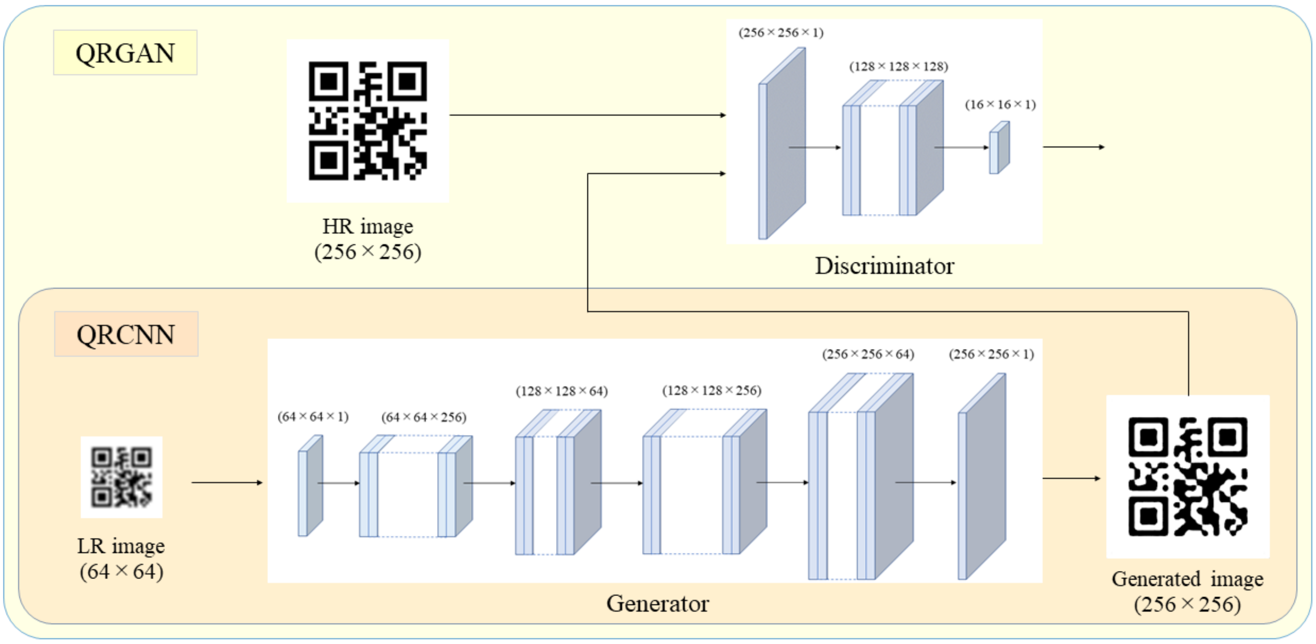

Fig. 1. Model structure of QRCNN and QRGAN

C. SRResNet

This model has similar deep structure to SRGAN, but does not have the discriminator. It is a model which we deprive the adversarial components of SRGAN. Similar to SRGAN, the generator attempts to produce an image that is close to the ground truth. Due to the lack of a discriminator, the loss function of SRResNet is given by

$$l^{SR} = \underbrace{l^{SR}_{X}}_{\text{content loss}} , \tag{3}$$

where the adversarial loss is not included. Same as SRGAN, content loss is calculated by MSE or obtained from feature maps derived from VGG.

III. Proposed Method

A. Model Structure

The model structure of both QRCNN and QRGAN are simple and the number of parameters of the generator used in these models is 176,449. Fig. 1 shows the model structure of QRCNN and QRGAN. Similar to SRResNet and SRGAN, our proposed methods enable super resolution by upscaling the input image in the middle of the network. The generator has three convolutional layers, two pixel shuffle layers and two activations (LeakyReLU). The discriminator of QRGAN has two convolutional layers, a batch normalization layer and an activation function (LeakyReLU). Contrary to SRResNet and SRGAN, proposed models are applied to grayscale images and does not use residual blocks or skip connections.

B. Loss Function

Like SRResNet, loss function of QRCNN has only content loss. Similar to SRGAN, loss function of QRGAN consists of content loss and adversarial loss. SRResNet and SRGAN use feature maps of VGG for content loss. However, our proposed methods use mean squared difference of pixels between the generated image and the ground truth image for the content loss. The loss function of QRCNN is given by

$$l^{SR} = \underbrace{l^{SR}_{MSE}}_{\text{content loss}} . \tag{4}$$

The loss function of QRGAN is given by

$$l^{SR} = \underbrace{l^{SR}_{MSE}}_{\text{content loss}} + \underbrace{10^{-3} l^{SR}}_{\text{adversarial loss}} . \tag{5}$$

IV. Experiment

A. Dataset

We use 1100 images from QR code image dataset [?], which is available on Kaggle. These QR code images contain linked numbers. When we scan these QR codes correctly, we can obtain these numbers. We use 1000 images for training and 100 images for evaluation. Since the size of the original image varies, we normalized them to a single size 256×256 [pixels] using bicubic method.

B. Training Details

As for training, we use 1000 QR code images. To simulate small QR code images, we convert training images to 64×64 [pixels] using bicubic method and apply Gaussian blur or add Gaussian noise. We use these resized images for the input of the generator of QRCNN and QRGAN. As for the input of the discriminator of QRGAN, the output image (256×256 [pixels]) generated by the generator of QRGAN and the ground truth image (256×256 [pixels]) are used. The number of epochs is just 3 and 10 for Gaussian blur and Gaussian noise, respectively, so the learning process is very short.

C. Evaluation Method

We use 100 images for evaluation. Similar to the training process, we convert testing images to 64×64 [pixels] using bicubic and apply Gaussian blur or add Gaussian noise. We use these as the input images for the pretrained generator of QRCNN and QRGAN and simulate whether the output images are readable. In other words, we simulate whether the linked number is obtained by scanning the output image. For this simulation, we use python module, pyzbar. Pyzbar can scan QR code images and determine what is linked with these QR codes.

D. Comparison

We compare the performance of our proposed methods and SRCNN. To ensure the fairness of this comparison, the number of input channels of SRCNN is fixed to one. SRCNN has three convolutional layers. The number of channels and the kernel size of these convolutional layers inherit the values recommended in the paper of SRCNN. The number of output channels of these convolutional layers are 128, 64 and 1, while the number of the kernel size of these layers correspond to 9, 5 and 5. The number of parameters of SRCNN is 216,961, which is larger than our proposed method. We train SRCNN similar to QRCNN and QRGAN. In SRCNN, low resolution input images are upscaled to the desired image size using bicubic interpolation before input to the network.

E. Evaluation

1) Gaussian blur: To evaluate the performance of QRCNN and QRGAN, we first prepare blurred low resolution QR code images. We use Gaussian blur for this purpose. To control blurriness of the input images, we change the parameters of Gaussian blur used in training and evaluation. The parameters are standard deviation and kernel size. We use these prepared QR codes images for the input of SRCNN, QRCNN and QRGAN. The result of the training process is shown in Fig. 2. (a) refers to the input image that Gaussian blur is applied, (b) is the output image of SRCNN, (c) is the output image of QRCNN, (d) is the output image of QRGAN and (e) is the ground truth. All output images are readable. Table I-V shows the result of the readability (%) corresponding to different parameter sets. The blur values refer to the number of QR code images which are already readable even before applying QRCNN or QRGAN. Other values refer to the number of readable QR code images which are generated by SRCNN, QRCNN and QRGAN. For example, when the standard deviation is 1.10 and the kernel size is 3, blurred images will be completely readable by applying QRCNN or QRGAN. From this simulation, some unreadable QR code images with Gaussian blur can be converted into readable ones by SRCNN, QRCNN and QRGAN. Moreover, our proposed methods show better performance than SRCNN even though the number of parameters is less.

TABLE I
RESULTS OF THE READABILITY OF GAUSSIAN BLUR (KERNEL SIZE 3)

Method	standard deviation					
	1.1	1.2	1.3	1.4	1.5	1.6
Blur	26	18	22	19	15	18
SRCNN	100	100	100	100	98	97
QRCNN	100	100	100	100	100	100
QRGAN	100	100	100	100	100	100

TABLE II
RESULTS OF THE READABILITY OF GAUSSIAN BLUR (KERNEL SIZE 5)

Method	standard deviation					
	1.1	1.2	1.3	1.4	1.5	1.6
Blur	4	0	0	0	0	0
SRCNN	100	86	66	35	24	6
QRCNN	100	100	75	56	53	55
QRGAN	100	100	74	59	50	55

TABLE III
RESULTS OF THE READABILITY OF GAUSSIAN BLUR (KERNEL SIZE 7)

Method	standard deviation					
	1.1	1.2	1.3	1.4	1.5	1.6
Blur	3	0	0	0	0	0
SRCNN	100	100	99	76	57	27
QRCNN	100	100	100	100	85	49
QRGAN	100	100	100	100	88	50

TABLE IV
RESULTS OF THE READABILITY OF GAUSSIAN BLUR (KERNEL SIZE 9)

Method	standard deviation					
	1.1	1.2	1.3	1.4	1.5	1.6
Blur	4	0	0	0	0	0
SRCNN	100	100	96	75	65	52
QRCNN	100	100	100	100	82	74
QRGAN	100	100	100	100	82	74

TABLE V
RESULTS OF THE READABILITY OF GAUSSIAN BLUR (KERNEL SIZE 11)

Method	standard deviation					
	1.1	1.2	1.3	1.4	1.5	1.6
Blur	4	0	0	0	0	0
SRCNN	100	100	95	75	65	51
QRCNN	100	100	100	100	76	74
QRGAN	100	100	100	100	77	74

TABLE VI
RESULTS OF THE READABILITY OF GAUSSIAN NOISE

Method	standard deviation					
	0.7	0.8	0.9	1.0	1.1	1.2
Noise	55	39	12	0	0	0
SRCNN	100	100	94	80	65	48
QRCNN	100	100	100	95	81	66
QRGAN	100	100	100	100	80	73

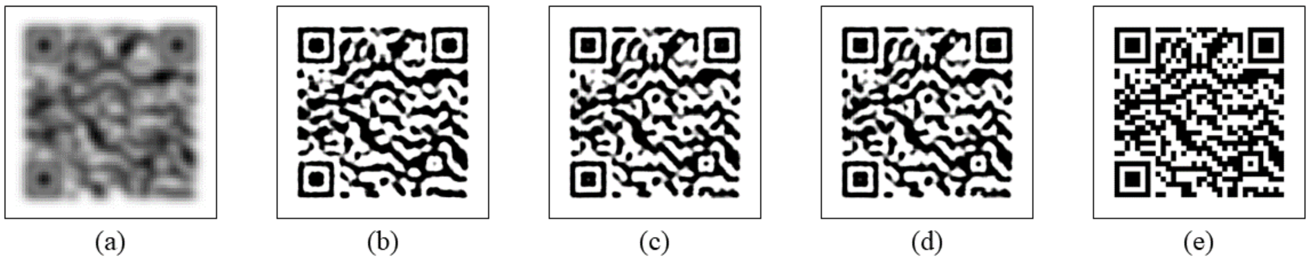

Fig. 2. Results of training for blurred images : (a) Input with kernel size 7 and standard deviation 1.4 (height and width are expanded by 4 to align with others); (b) Output of SRCNN; (c) Output of QRCNN; (d) Output of QRGAN; (e) Ground truth.

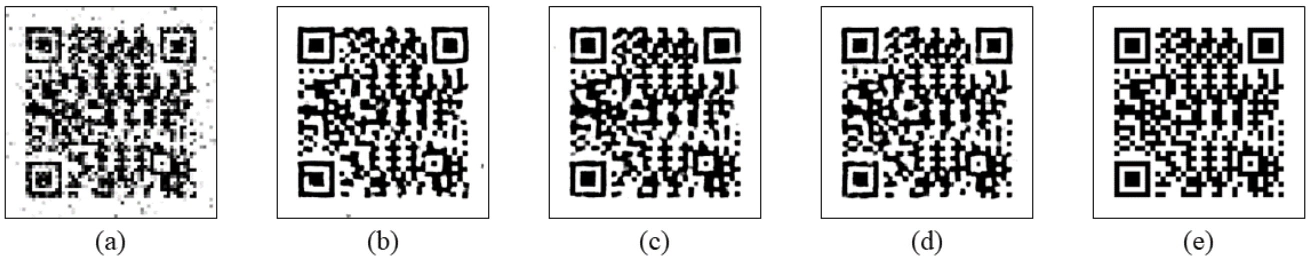

Fig. 3. Results of training for noise images : (a) Input with standard deviation 1.0 (height and width are expanded by 4 to align with others); (b) Output of SRCNN; (c) Output of QRCNN; (d) Output of QRGAN; (e) Ground truth.

2) Gaussian noise: Next, we prepare low resolution QR code images with noise. We use Gaussian noise for this purpose. To control the levels of noise, we change the parameter of Gaussian noise used in training and evaluation. The parameter we change is only standard deviation. We use these prepared QR code images as an input of SRCNN, QRCNN and QRGAN. The result of the training process is shown in Fig. 3. (a) refers to the input image with Gaussian noise, (b) is the output image of SRCNN, (c) is the output image of QRCNN, (d) is the output image of QRGAN and (e) is the ground truth. All output images are readable. Table VI shows the result of the readability (%) corresponding to different parameters. The noise values refer to the number of QR code images which are already readable even before applying QRCNN or QRGAN. Other values refer to the number of readable QR code images which are generated by SQCNN, QRCNN and QRGAN. For example, when the standard deviation is 0.7, images with noise will be completely readable by applying QRCNN or QRGAN. Similar to the first simulation, the result of this simulation also show the superiority of our proposed methods. QRCNN and QRGAN can be applied to images not only with blur but with noise.

V. Conclusion

In this paper, we propose QRCNN and QRGAN. QRCNN and QRGAN can convert unreadable QR code images into readable ones. From the experiments, our proposed methods are effective for QR code images with blur or noise. Moreover, our methods are superior to a typical CNN-based model, SRCNN, in terms of QR code image super resolution. We simplify the model structure by only allowing QR code images for the target of image denoising and super resolution. As a result, both QRCNN and QRGAN are able to reduce the number of parameters and computational complexity. Our model does not incorpolate pretrained model, so the memory usage is also small. Therefore, QRCNN and QRGAN can be implemented without relying on high performance GPUs.

Acknowledgment

The results of this research are based on the "Research and Development of Ultra-Coverage Beyond 5G Wireless Communications and Video Coding Standardization Technology through International Collaboration among Japan, the United States, and Australia" of the "Beyond 5G Research and Development Promotion Project (General Type)" commissioned by the National Institute of Information and Communications Technology (NICT), a research and development project for innovative information and communications technology (No. 05101).

References

[1] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio,"Generative adversarial nets", NIPS, 2014.
[2] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al., "Photo-realistic single image superresolution using a generative adversarial network", CVPR, 2017.
[3] K. Simonyan, and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", ICLR, 2015.
[4] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution", ECCV, 2014.
[5] V. Nair and G.E. Hinton, "Rectified linear units improve restricted boltzmann machines", ICML, 2010.
[6] R. G. Keys, "Cubic convolution interpolation for digital image processing", IEEE Trans. Acoust., Speech, Signal Process., vol. ASSP-29, pp. 1153-1160, Dec. 1981.
[7] Cole Dieckhaus, "QR Codes", Kaggle, 18 Feb. 2020.