Self Texture Transfer Networks for Low Bitrate Image Compression

Shoma Iwai, Tomo Miyazaki, Yoshihiro Sugaya, Shinichiro Omachi
Department of Communications Engineering, Graduate School of Engineering,
Tohoku University
{shoiwai, tomo, sugaya, machi}@iic.ecei.tohoku.ac.jp

Abstract

Lossy image compression causes a loss of texture, especially at low bitrate. To mitigate this problem, we propose a novel image compression method that utilizes a reference-based image super-resolution model. We use two image compression models and a self texture transfer model. The image compression models encode and decode a whole input image and selected reference patches. The reference patches are small but compressed with high quality. The self texture transfer model transfers the texture of reference patches into similar regions in the compressed image. The experimental results show that our method can reconstruct accurate texture by transferring the texture of reference patches.

1. Introduction

Recently, many machine learning-based image compression methods have been proposed. Some methods outperform human-crafted compression methods, such as BPG [4] and VVC. Most of them have an encoder, decoder, and entropy estimator. They are trained to optimize the rate-distortion trade-off in an end-to-end manner.

To improve the rate-distortion performance, some works have proposed effective entropy estimators. Balle *et al*. [3] have proposed hyperprior network. Minnen *et al*. [17] have introduced PixelCNN [20] based auto-regressive context model. Guo *et al*. [7] have proposed 3-D global context model. Other works investigate architecture of autoencoder. Chen *et al*. [5] adopt a non-local attention module in the encoder and decoder. Akutsu *et al*. [2] use RAM block [11] for feature extraction.

Though many methods use mean square error (MSE) or MS-SSIM for training criteria, these losses lead to blurry results. To overcome this problem, some methods [18, 1, 16, 8] adopt GAN framework [6]. GAN-based method can generate photo-realistic images. Rippel and Bourdev [18] introduced adversarial training. Agustsson *et al*. [1] achieved to reconstruct perceptually high-quality images even at extremely low bit rate. Mentzer *et al*. [16] have proposed a high-fidelity GAN-based image compression model. Iwai *et al*. [8] proposed two-stage training and network interpolation to avoid unstable training and undesirable noise.

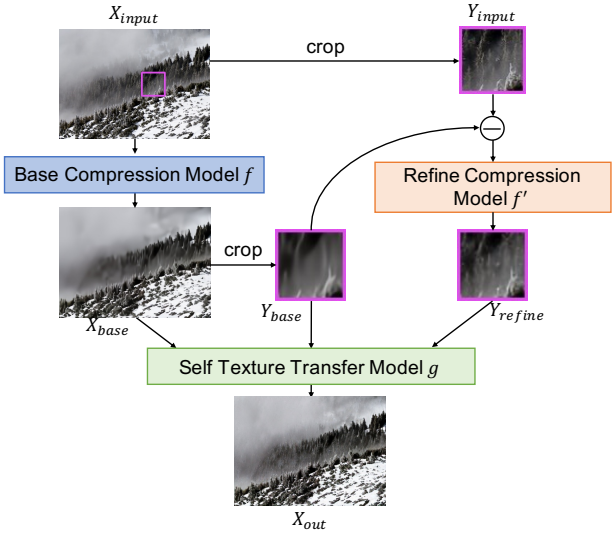

Figure 1. Our method overview.

Although GAN-based method can reconstruct sharp images, it is difficult to recover accurate texture at low bitrate. To compensate for the loss of texture, we propose a self texture transfer network. It is inspired by the state-of-the-art reference-based image super-resolution method [22]. We spend additional bits to reconstruct small parts of the image with high quality and transfer their texture to similar regions in the image. To do this, we use three networks: a base compression network, refine compression network, and self texture transfer network. First, we compress a whole image by the base compression network. Second, we crop patches from the image and compress them by the refine compression network. These patches have richer texture, and we use them as reference patches. Then, we transfer the texture of them into the compressed image by self texture transfer

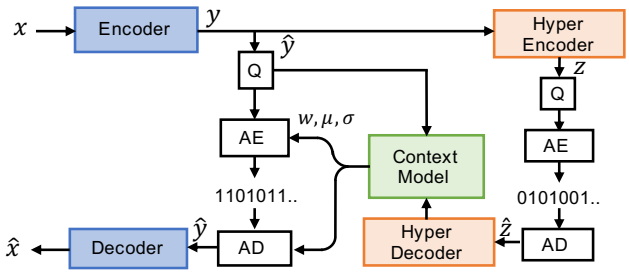

Figure 2. Compression model

network.

2. Proposed Method

2.1. Overview

Fig 1 shows our overall method. First, we compress the input image X_{input} by the base compression model f and obtain a reconstruction $X_{base} = f(X_{input})$. Next, we manually choose the patches Y_{base} from the X_{base}. We crop the same position and obtain the real patches Y_{input} from X_{input}. Then we encode and decode the residual $Y_{input} - Y_{base}$ by the refine compression model and get refine patches $Y_{refine} = f'(Y_{input} - Y_{base})$. Finally, we enhance the texture of X_{base} according to Y_{refine} by self texture transfer model g. The final output is $X_{out} = g(X_{base}, Y_{base}, Y_{refine})$. Hence, a receiver needs the position of reference patches and the latent code of X_{base} and Y_{refine} for decoding.

2.2. Compression Model

Fig 2 shows the architecture of the base and refine compression models. They have the same architecture. The base compression model handles a whole image, and the refine compression model compresses only the selected patches with high quality. Our compression model consists of an encoder, decoder, hyper-encoder, hyper-decoder, and context model. The encoder transforms an input image x into latent code y, and the decoder reconstructs the image from quantized latent code $\hat{y}$. Inspired by Akutsu *et al.* [2], we use RAM block [11] in the encoder and decoder. The rest networks: hyper-encoder, hyper-decoder, and context model, estimate the entropy of $\hat{y}$. The hyper-encoder and hyper-decoder provide side-information for entropy estimation. The context model is an auto-regressive model that estimates the parameters of the distribution of $\hat{y}$. Since we use Gaussian Mixture model for distribution estimation, it estimates three parameters: weight w, mean μ, and standard deviation σ for each position of $\hat{y}$. Then we calculate the probability mass function of $\hat{y}$ and transform it into bitstream by arithmetic coding. Inspired by Guo *et al.* [7], we adopt causal global prediction in the context model.

2.3. Self Texture Transfer Model

Our self texture transfer model is based on the reference-based image super-resolution method [22]. Fig 3 shows the structure of the self texture transfer model. It enhances the quality of X_{base} by using reference patches Y_{refine}. It consists of three modules: a feature extraction module, attention module, and fusion module. First, the feature extraction module extracts deep features from the base image X_{base}. Next, the attention module calculates the similarity between each point of X_{base} and Y_{base}, and searches the most similar point. Then, the fusion module transfers the texture of Y_{refine} into X_{base} according to the similarity. In the following subsections, we will show the details of these modules.

2.3.1 Feature Extraction Module

For feature extraction module, we adopt RFB-ESRGAN [19], which is based on ESRGAN [21]. It took first place on NTIRE 2020 Perceptual Extreme Super-Resolution Challenge [23]. The structure of the feature extraction module is shown in fig 3. It has 16 residual in residual dense blocks (RRDBs) and 8 residual in residual dense receptive field blocks (RRFDBs). Each RRDB contains three residual dense blocks, and each of them has five convolution layers. Likewise, each RRFDB contains three residual dense receptive field blocks, and each of them has five receptive field blocks (RFBs) [15]. The RFB consists of convolution layers which have various kind of kernel size.

To reduce memory usage, we use an intermediate feature map of the base decoder F_{inter} as the input of this module, rather than decoded image X_{base}. Its spatial size is $\frac{H}{4} \times \frac{W}{4}$, where H and W are height and width of an input image X_{input}, respectively.

2.3.2 Attention Module

The attention module calculates two kinds of attention maps: hard attention map T and soft attention map S. First, learnable texture extractor (LTE) extracts three feature maps: query Q, key K, and value V from X_{base}, Y_{base}, and Y_{refine}, respectively.

$$Q = LTE(X_{base}) \tag{1}$$
$$K = LTE(Y_{base}) \tag{2}$$
$$V = LTE(Y_{refine}) \tag{3}$$

The architecture of LTE is the same as VGG-19 [13]. Second, we calculate relevance map R:

$$R = QK^T (R \in \mathbb{R}^{n \times m}, Q \in \mathbb{R}^{n \times d}, K \in \mathbb{R}^{m \times d}), \tag{4}$$

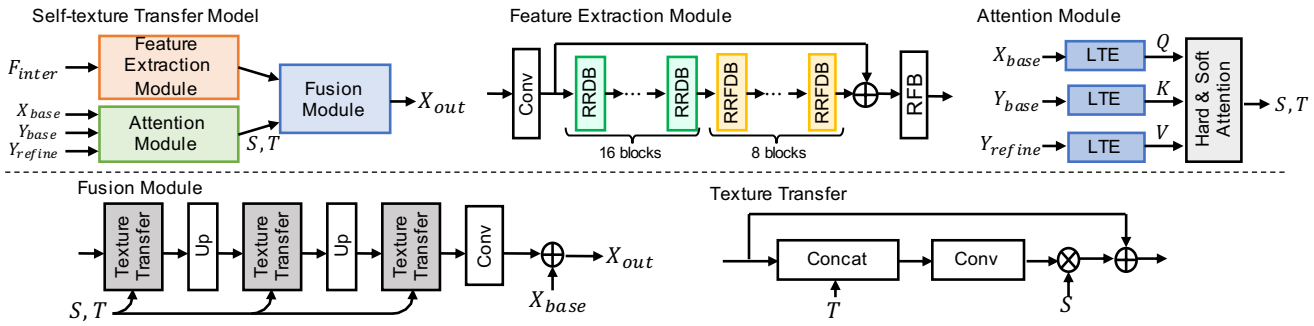

Figure 3. Self texture transfer model

where n, m, and d are $H_Q \times W_Q$, $H_K \times W_K$, and the channels of feature maps, respectively. R represents the similarity between each spatial point in Q and K. For example, $r_{i,j}$ denotes the similarity between query q_i and key k_j. Then, we calculate hard attention map $T = (t_0, \cdots, t_n)$ and soft attention map $S = (s_0, \cdots, s_n)$. t_i represents the feature of V in the most relevant position for query q_i. s_i represents the similarity between q_i and the most similar key. These attention maps are calculated as follows:

$$h_i = \arg\max_j r_{i,j} \tag{5}$$

$$t_i = v_{h_i} \tag{6}$$

$$s_i = \max_j r_{i,j}, \tag{7}$$

where h_i, v_i, and $r_{i,j}$ denote the index of most relevant position, the element of V and R, respectively.

2.3.3 Fusion Module

The fusion module transfers the texture of Y_{refine} into X_{base} by fusing the attention maps and features extracted by the feature extraction module. As shown in fig 3, it has three texture transfer layers, two up-sampling layers, and a final convolution layer. The texture transfer layer fuses hard attention map T and feature map by concatenation and convolution. Soft attention map S is used to avoid inaccurate texture transferring. The operation is as follows:

$$F_{fusion} = F + \text{Conv}(\text{Concat}(F, T)) \odot S, \tag{8}$$

where F and F_{fusion} are an input and output feature map, respectively. Finally, we obtain the output image X_{out} by adding X_{base} to the final convolution layer's output.

2.4. Training Strategy

Our training strategy has three stages. First, we train the base compression model. We use distortion loss and rate loss for training. Second, we train only the refine compression model, so the base compression model is fixed. We use MS-SSIM loss and rate loss in this stage because MS-SSIM optimized model can reconstruct fine texture than MSE optimized model. Finally, we train the self texture transfer model. The loss function in this stage is almost the same as ESRGAN [21], but we use LPIPS [24] instead of VGG perceptual loss [9]. We use RaGAN [10] for adversarial training. The loss function of all stages are as follows.

$$\mathcal{L}_1 = \lambda_d \mathcal{L}_d + \mathcal{L}_{rate} \tag{9}$$

$$\mathcal{L}_2 = \lambda_{MS-SSIM} \mathcal{L}_{MS-SSIM} + \mathcal{L}_{rate} \tag{10}$$

$$\mathcal{L}_3 = \lambda_{l1} \mathcal{L}_{l1} + \lambda_{adv} \mathcal{L}_{adv} + \lambda_{LPIPS} \mathcal{L}_{LPIPS} \tag{11}$$

3. Experimental Settings

For the CLIC2021 image compression track, we trained three different bit-rate models: 0.075, 0.15, and 0.3 bpp in CLIC validation dataset. For the distortion loss in the first stage, we used MSE loss for the 0.075 bpp model, and MS-SSIM loss for the rest model. Though MS-SSIM loss is effective to preserve texture, it tends to fail in preserving structure or contents especially at low bit rate. So we chose MSE loss for the lowest bitrate model. In the first stage training, we adjusted the weight parameter λ_d so that the total bit rate will not exceed the specified bit rate. In the second stage, we trained the refine compression model so that the average bit rate of CLIC validation dataset becomes 0.20, 0.25, and 0.25 bpp for 0.075, 0.15, and 0.3 bpp model, respectively. In the third stage, we trained the self texture transfer model with $\lambda_{l1} = 0.01$, $\lambda_{adv} = 0.005$, and $\lambda_{LPIPS} = 4.0$ for all three bit-rate models.

For training, we used CLIC training dataset and subsets of Open Image dataset [14]. We used 256×256 patches extracted from training images in the first stage and 192×192 patches in the second and third stages. During third stage training, we randomly chose reference patches from the image. We used Adam optimizer [12], and the learning rate is set to 1×10^{-4}. The training iteration of the first, second, and third stages are about $1M$, $700K$, and $300K$, respectively.

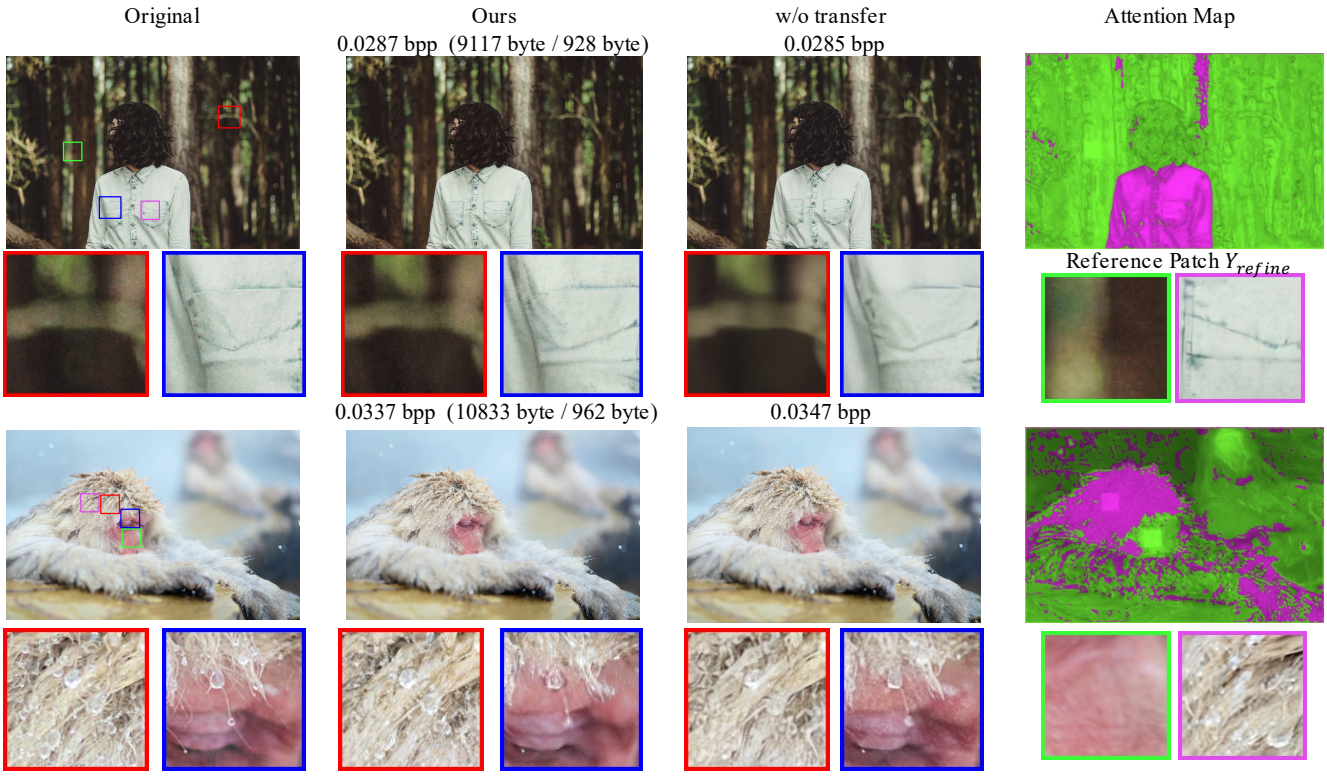

Figure 4. Qualitative results, reference patches, and attention map. The numbers in column "Ours" represent the total bitrate, data size of the base image, and data size of the reference patches, respectively.

	bpp	PSNR	MS-SSIM
Ours	0.074	27.696	0.91327
w/o transfer	0.076	27.876	0.92488
Ours	0.150	27.719	0.95983
w/o transfer	0.148	27.737	0.96186
Ours	0.293	29.472	0.97932
w/o transfer	0.295	29.591	0.98010

Table 1. Quantitative results on CLIC validation dataset. Our team name is "iiclab".

4. Results

To confirm the validity of self texture transfer model, we trained our model without the refine compression model, attention module, and fusion module. In this model, the RFB-ESRGAN based feature extraction module enhances the output of the base compression model, but self texture transfer is not used. For this model, we trained another base compression model so that the bit-rate would be approximately the same. This model is denoted "w/o transfer."

Fig 4 shows the results. We chose two 128×128 patches for reference (green and magenta squares in fig 4). As shown in fig 4, our method reconstructs more accurate texture by transferring the texture of the reference patches into other similar regions. On the contrary, the reconstructions of the w/o transfer model are blurry or noisy.

The rightmost column of fig 4 shows attention maps. The hue represents which patch is chosen as a reference for each pixel (hard attention T), and the brightness represents the confidence of transferring (soft attention S). They show that the attention module chooses appropriate regions as a reference, and a soft attention map can represent similarity.

Table 1 shows the quantitative results. Though our method recovers more plausible textures by transferring features to a similar region, it does not improve per-pixel fidelity. However, our main focus is on perceptual quality rather than distortion.

5. Conclusion

In this paper, we proposed a self texture transfer model. We use another image compression model to generate reference patches and transfer the texture of them into a compressed image. Our experiments showed that our method succeeds in transferring the texture. In future work, we will consider an algorithm to choose the optimal reference patches. For the CLIC2021 competition, we manually chose the reference patches for encoding. However, for practical usage, the encoding process should be fully automatic.

References

[1] E. Agustsson, M. Tschannen, F. Mentzer, R. Timofte, and L. Van Gool. Generative adversarial networks for extreme learned image compression. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 1

[2] H. Akutsu, A. Suzuki, Z. Zhong, and K. Aizawa. Ultra low bitrate learned image compression by selective detail decoding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020. 1, 2

[3] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston. Variational image compression with a scale hyperprior. In *International Conference on Learning Representations, ICLR*, 2018. 1

[4] F. Bellard. Bpg image format. 1

[5] T. Chen, H. Liu, Z. Ma, Q. Shen, X. Cao, and Y. Wang. Neural image compression via non-local attention optimization and improved context modeling, 2019. 1

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680. 2014. 1

[7] Z. Guo, Z. Zhang, R. Feng, and Z.Chen. Causal contextual prediction for learned image compression, 2020. 1, 2

[8] S. Iwai, T. Miyazaki, Y. Sugaya, and S. Omachi. Fidelity-controllable extreme image compression with generative adversarial networks. In *25th International Conference on Pattern Recognition (ICPR)*, page 8235–8242, January 2021. 1

[9] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 2016. 3

[10] A. Jolicoeur-Martineau. The relativistic discriminator: a key element missing from standard gan. *arXiv preprint arXiv:1807.00734*, 2018. 3

[11] J. Kim, J. Choi, M. Cheon, and J. Lee. Mamnet: Multi-path adaptive modulation network for image super-resolution. *Neurocomputing*, 402:38–49, Aug 2020. 1, 2

[12] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations, ICLR*, 2015. 3

[13] K.Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition, 2014. 2

[14] A. Kuznetsova, H. Rom, N Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, A. Kolesnikov, and et al. The open images dataset v4. *International Journal of Computer Vision*, 128(7):1956–1981, Mar 2020. 3

[15] S. Liu, D. Huang, and Y. Wang. Receptive field block net for accurate and fast object detection. In *The European Conference on Computer Vision (ECCV)*, September 2018. 2

[16] F. Mentzer, G. Toderici, M. Tschannen, and E. Agustsson. High-fidelity generative image compression. *Advances in Neural Information Processing Systems*, 33, 2020. 1

[17] D. Minnen, J. Ballé, and G. Toderici. Joint autoregressive and hierarchical priors for learned image compression. In *Advances in Neural Information Processing Systems*, pages 10771–10780. 2018. 1

[18] O Rippel and L. D. Bourdev. Real-time adaptive image compression. In *International Conference on Machine Learning, ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 2922–2930. PMLR, 2017. 1

[19] T. Shang, Q. Dai, S. Zhu, T. Yang, and Y. Guo. Perceptual extreme super-resolution network with receptive field block. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020. 2

[20] A. van den Oord, N. Kalchbrenner, L. Espeholt, K. Kavukcuoglu, O. Vinyals, and A. Graves. Conditional image generation with pixelcnn decoders. In *Advances in Neural Information Processing Systems*, pages 4790–4798. Curran Associates, Inc., 2016. 1

[21] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *The European Conference on Computer Vision Workshops (ECCVW)*, September 2018. 2, 3

[22] F. Yang, H. Yang, J. Fu, H. Lu, and B. Guo. Learning texture transformer network for image super-resolution. In *CVPR*, June 2020. 1, 2

[23] K. Zhang, S. Gu, and R. Timofte. Ntire 2020 challenge on perceptual extreme super-resolution: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020. 2

[24] R. Zhang, P. Isola, A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 3