3-D Context Entropy Model for Improved Practical Image Compression

Zongyu Guo, Yaojun Wu, Runsen Feng, Zhizheng Zhang, Zhibo Chen*
CAS Key Laboratory of Technology in Geo-spatial Information Processing and Application System
University of Science and Technology of China
{guozy,yaojunwu,fengruns,zhizheng}@mail.ustc.edu.cn, chenzhibo@ustc.edu.cn

Abstract

In this paper, we present our image compression framework designed for CLIC 2020 competition. Our method is based on Variational AutoEncoder (VAE) architecture which is strengthened with residual structures. In short, we make three noteworthy improvements here. First, we propose a 3-D context entropy model which can take advantage of known latent representation in current spatial locations for better entropy estimation. Second, a light-weighted residual structure is adopted for feature learning during entropy estimation. Finally, an effective training strategy is introduced for practical adaptation with different resolutions. Experiment results indicate our image compression method achieves 0.9775 MS-SSIM on CLIC validation set and 0.9809 MS-SSIM on test set.

1. Introduction

Image compression is a ubiquitous technique in the digital age. Traditional image compression standards take years to develop a new generation. With the rapid development of Deep Neural Networks (DNNs), learning-based image compression method presently is attractive and achieves some promising breakthroughs [4, 7, 8]. Early learning-based method [14] is based on RNN and supports coding scalability. However, image compression is a rate-distortion trade-off game and such RNN-related work cannot directly optimize the rate during network training.

Recently, most learning-based image compression approaches are based on VAE architecture, where rate R and distortion D are jointly optimized in an end-to-end manner [2]. Ballé *et al*. [3] propose a hyperprior entropy model, which parameterizes the latent distribution and predicts their standard deviations as Gaussian Scale Model (GSM). After that, [10] and [12] introduce context entropy model to utilize adjacent known regions for better parameter estimation and improve original GSM to Single Gaussian Model (SGM). Recent works [11, 6] further suggest a more generalized format to predict the distribution of latent representation, *i.e*., Gaussian Mixture Model (GMM). GMM theoretically is able to approximate arbitrary continuous probability distribution. Those impressive improvements mentioned above mainly concentrate on the hyperprior model for parameter estimation. Additionally, the backbone network can also be enhanced with some techniques such as attention mechanism [15, 6] and post-processing network [11].

In this paper, motivated by the aforementioned methods, we build our image compression network for CLIC 2020 low rate track and highlight three main improvements. First, we propose a 3-D context entropy model which divides latent representations into two groups across channels. This 3-D context model can better extract correlations of latent features which are in the same spatial location but vary in channel. Second, a residual structure is adopted to refine the estimated entropy parameters. The designed residual parameter estimation (RPE) module efficiently cooperates with the 3-D context model thanks to the light-weighted but effective structure. Third, a novel training strategy is employed for practical image compression. We know that due to the downsampling layer in network, learning-based codec usually requires the input to have an integer-multiple resolution of values such as 32 or 64. Consequently, when dealing with such images with different resolutions, we should first conduct padding process. This may lead to unnecessary bit waste on padded areas. The proposed training strategy enables network to adapt to different padding situations in the time of training.

In CLIC 2020 low rate track, our team IMCL_IMG _MSSSIM got 0.9775 and 0.9809 MS-SSIM during the validation phase and test phase respectively.

2. Method

2.1. The overall framework

Before introducing our proposed new techniques, we first present our overall framework, which is shown in Figure 1. Similar to previous work [12], this pipeline can be di-

*Zongyu Guo and Yaojun Wu contribute equally to this work. Zhibo Chen is the corresponding author.

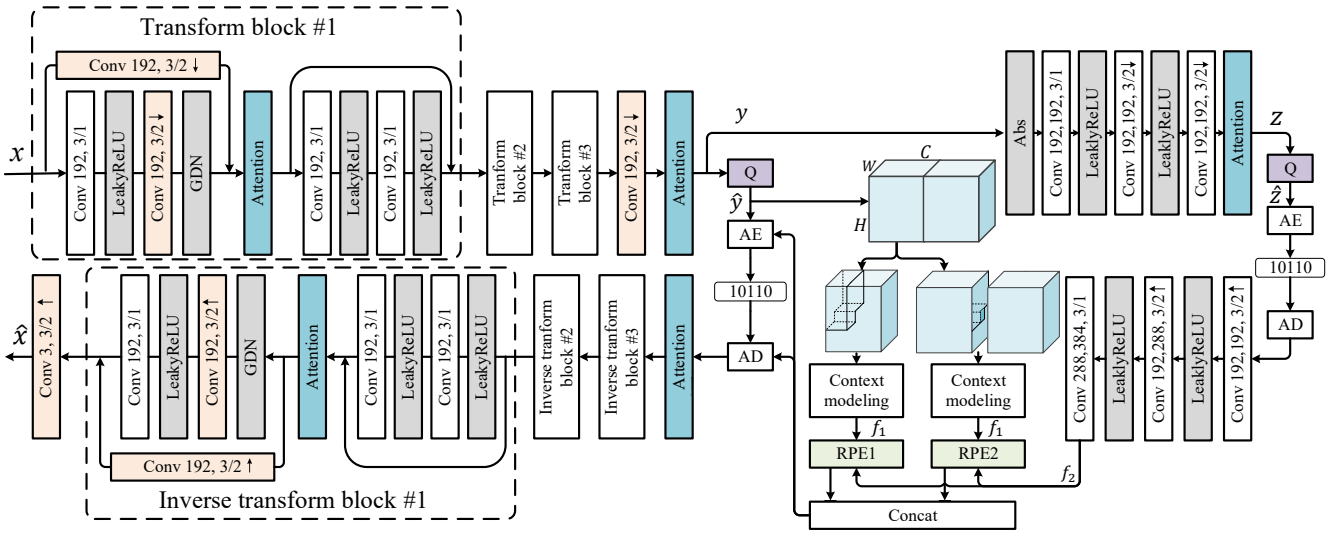

Figure 1: The overall framework of our image compression model. The context entropy model is a 3-D version.

vided into three parts: an analysis transform encoder, a synthesis transform decoder, and a hyperprior entropy model (including a hyper encoder and a hyper decoder). The hyperprior part will be discussed later. This backbone network is an improved version based on [6]. Specifically, the raw input image x will be transformed to the latent features y, which will be quantized to $\hat{y}$ and then decoded to reconstructed image $\hat{x}$. The analysis transform encoder contains three transform blocks, each of which is made of a residual downsampling layer, an attention layer and a residual enhancement layer. After three transform blocks, there are a downsampling convolution layer and an attention layer to increase receptive field. The architecture of synthesis decoder is symmetric, *i.e.*, an attention layer, three inverse transform blocks and an additional upsampling layer.

Compare with the baseline network [6], we modify their model with several extra attention modules in the encoder side, which has no increasing complexity for decoding (two in analysis transform encoder and one in hyper encoder). Besides, GRDN [9], a post-processing network recommended in [11], is adopted following the main compression network to further enhance image quality, which is omitted in Figure 1.

2.2. 3-D context entropy model

As a part of hyperprior model, context entropy model was first proposed in [12] and [10]. This context model is autoregressive over latents and is usually implemented in the format of 5×5 mask convolution [12]. Such context entropy model plays an important role for the estimation of feature parameters though it would increase decoding time complexity dramatically.

The mask convolution layer in previous context model can effectively capture spatial correlations to predict current pixel, which is similar to classical intra prediction. Our experiments indicate that not only spatial redundancy can be eliminated, there also exists channel-wise redundancy, even though Generalized Divisive Normalization (GDN) is proved to well Gaussianize features in the channel direction.

Assuming we are predicting current latent representation y, its location is $[i, j, k]$, where i and j are the coordinate of height and width and k is the channel location index. While original 2-D context model concentrates on the left and up features $\hat{y}_{i-h,j-w}$, the proposed 3-D context model further leverages known (decoded) features in current spatial location, *i.e.*, $\hat{y}_{i,j,k-c}$. Ideally, different channel requires different mask convolution in our 3-D context model, *e.g.*, feature in the first channel can be predicted only with the up and left features but feature in the last channel can be predicted with those known features in current spatial location. However, this ideal situation will complicate model because in this case, every channel should have its own parameter estimation module. Therefore, we finally choose to compromise which divides all channels into two groups. Each group has its own weights of mask convolution and now there are two independent parameter estimation modules for those two groups. As shown in Figure 1, the first group is predicted as usual but the second group can be predicted based on the first group.

The proposed 3-D context model enables the sequential decoding process to be more sequential. We have tried to divide channels into more groups, which was found to improve little. This 3-D context model is analogous to the conditional RGB prediction model in PixelCNN [13]. The difference is that here are more channels rather than only three in PixelCNN and we divide these channels into two

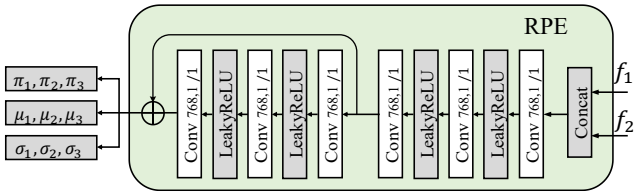

Figure 2: Residual parameter estimation module. There is only one residual connection and thus the RPE module is relatively lighted-weighted but effective.

groups for simplification.

2.3. Residual parameter estimation module

Cheng *et al*. [5] comprehensively discuss the residual architecture for image compression. As they shown, residual structures in analysis transform and synthesis transform obviously strengthen the capability of network. Motivated by this, we think the entropy parameter estimation module can also be enhanced with the help of residual structure.

As shown in Figure 1, after obtaining context feature f_1 and hyper feature f_2 reconstructed from $\hat{y}$ and $\hat{z}$, we employ a residual parameter estimation (RPE) module to estimate the probability distribution of $\hat{y}$. As mentioned before, the distribution of latent features is modeled as Gaussian Mixture Model (GMM) following [11, 6], *i.e*.,

$$p(\hat{y}) \sim \sum_{k=1}^{K} \pi_k \mathcal{N}(\mu_k, \sigma_k^2). \tag{1}$$

In our experiments, we find that $K = 3$ is enough to accurately estimate the distribution of latent representations. The structure of RPE module is presented in Figure 2. There are three 1×1 convolution layers to process the concatenation of f_1 and f_2. Then there follows a residual component which also contains three 1×1 layers. Such 1×1 residual convolution layers, which can also be regarded as fully connected layers, mainly work for features across the entire channels instead of spatial features. Therefore, it will not influence the sequential decoding process.

Lee *et al*. [11] propose a Model Parameter Refinement Module (MPRM) to cooperate with global context. Our designed residual entropy parameter module is partially different from theirs because here we only have one residual block, which is effective and light-weighted. The moderate parameter number is also advantageous for the 3-D context model because the 3-D context model here doubles the whole parameter number of entropy estimation module.

2.4. More practical image compression

Practical image compression codec requires to handle those images with different resolutions. It is problematic even for traditional block-based image compression methods, *e.g*., VTM (VVC test model) [1] would first change image resolution to an integer multiple of 8. Considering learning-based image compression methods, this problem is always more serious because there are many downsampling layers in network which would cause resolution inconsistency after inversion. A conventional solution would be extra padding process before encoding. However, learning-based network is usually trained with full cropped patches such as 256×256 patches. As a result, the network cannot handle these padded image properly in practical applications and then performance usually drops.

In our framework, there are totally six downsampling layers and thus input images should have an integer multiple resolution of $2^6 = 64$. First we note that experiments prove that zero-padding is optimal than other padding methods such as reflection-padding. Here we propose a strategy to enable network to adapt to the padding effects during training. The pseudo code of proposed algorithm is as following in Algorithm 1.

Algorithm 1 Training strategy for practical compression

Input: A mini-batch data x randomly cropped from training dataset, the shape of which is $[B, C, H, W]$

Flag $\leftarrow$ random sample $\in \{0, 1\}$.
if *Flag* is 0 **then**
  Normally optimize your network.
else
  Randomly get the padding size for current batch.
  Select $h_{pad} \in [0, P_1]$, $w_{pad} \in [0, P_2]$.
  Zero-pad input x right and down. Then its shape is $[B, C, H + h_{pad}, W + w_{pad}]$.
  Crop x to simulate the real input image after padding: $x = x[:, :, h_{pad} :, w_{pad} :]$.
  Calculate pixel number $(H - h_{pad}) \times (W - w_{pad})$ to obtain actual bitrate R of current batch. Then aggregate the distortion loss D which only covers unpadding area.
  Optimize your network.
end if

In this algorithm, P_1 and P_2 are given upper bound to control the padding size during training. In our experiments, considering that input patch is 256×256 (H=W=256), we empirically set P_1 = P_2 = 20. In short, we want to enable network to have access to padded images even if those images are imitated by manually crafted padding. Randomly choosing padding size will help network adapt to different images with different padding situation. This training strategy is verified to largely improve the performance in CLIC validation dataset (the actual required bitrate decreases).

Model	MSSSIM	PSNR	BPP
Single model ($\lambda = 16$)	0.97812	30.30	0.1548
Two models ($\lambda = \{12, 16\}$)	0.97753	30.19	0.1499
Four models (λ from $[10, 24]$)	0.97754	30.19	0.1499

Table 1: Performance on CLIC 2020 validation dataset. Optimized for MS-SSIM.

3. Implementation details

We train our network with 256×256 patches randomly cropped from CLIC training set, DIV2K and Flickr 2K dataset, which has the same setting as [15]. We divide the training period into three stages. We first train our main compression network without post-processing. Then we fix the parameters in the main compression network and train corresponding post-processing module GRDN [9]. At the last step, we jointly optimized the whole pipeline to achieve the best results. Notably, the proposed training strategy for padding effect is applied only at the third stage. At different training stages, we all take a learning rate decay strategy, *i.e.*, $lr = 1e-4$ in the initial 300,000 iterations and $lr = 1e-5$ for the rest 300,000 iterations. We train the network on two RTX 2080 Ti GPUs when batch size is set to 8.

Due to the limit of 0.15 bpp in CLIC competition, we train different models for different compression ratios. As usual, the loss function is $\mathcal{L} = R + \lambda D$, where $D = 1 - msssim$. Note that the loss function is modified when we employ the proposed training strategy for padding at the third training stage. Considering that we optimize for MS-SSIM, we select appropriate λ value ranging from 10 to 24. Table 1 shows the results of our methods including single model and multiple models. However, it seems that four models have little improvement compared with two models, which may imply that our rate control strategy is not satisfactory and has room to improve. Our final submitted version is this four-model codec, which achieves 0.9775 MS-SSIM score for validation and 0.9809 MS-SSIM for test.

4. Conclusion

In this paper, we introduce our image compression framework used in CLIC 2020 competition. Three useful techniques are adopted. First we improve the conventional 2-D context entropy model to a 3-D format which can better utilize decoded features in current spatial location. Second, the parameter estimation module is enhanced with a light-weighted residual structure. Lastly, we propose a training strategy to handle those images with different resolutions in real application. This training strategy is simple but effective to alleviate the bit waste due to preliminary padding. As shown on the leaderboard, our team IMCL_IMG_MSSSIM got the second place in terms of MS-SSIM in the validation phase. In the future, we will pay more attention to more practical image compression methods, *i.e.*, lighter, faster and more robust DNN-based codec.

Acknowledgment

This work was supported in part by NSFC under Grant U1908209, 61632001 and the National Key Research and Development Program of China 2018AAA0101400.

References

[1] Versatile video coding reference software version 8.0(VTM-8.0). `https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM/tags/VTM-8.0`, February 2020.

[2] Johannes Ballé, Valero Laparra, and Eero P Simoncelli. End-to-end optimized image compression. *arXiv preprint arXiv:1611.01704*, 2016.

[3] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. *arXiv preprint arXiv:1802.01436*, 2018.

[4] Zhibo Chen and Tianyu He. Learning based facial image compression with semantic fidelity metric. *Neurocomputing*, 338:16–25, 2019.

[5] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Deep residual learning for image compression. *arXiv preprint arXiv:1906.09731*, 2019.

[6] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Learned image compression with discretized gaussian mixture likelihoods and attention modules. *arXiv preprint arXiv:2001.01568*, 2020.

[7] Zongyu Guo, Zhizheng Zhang, and Zhibo Chen. Deep scalable image compression via hierarchical feature decorrelation. In *2019 Picture Coding Symposium (PCS)*, pages 1–5. IEEE, 2019.

[8] 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.

[9] Dong-Wook Kim, Jae Ryun Chung, and Seung-Won Jung. Grdn: Grouped residual dense network for real image denoising and gan-based real-world noise modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[10] Jooyoung Lee, Seunghyun Cho, and Seung-Kwon Beack. Context-adaptive entropy model for end-to-end optimized image compression. *arXiv preprint arXiv:1809.10452*, 2018.

[11] Jooyoung Lee, Seunghyun Cho, and Munchurl Kim. A hybrid architecture of jointly learning image compression and quality enhancement with improved entropy minimization. *arXiv preprint arXiv:1912.12817*, 2019.

[12] David Minnen, Johannes Ballé, and George D Toderici. Joint autoregressive and hierarchical priors for learned image compression. In *Advances in Neural Information Processing Systems*, pages 10771–10780, 2018.

[13] Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016.

[14] 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.

[15] Lei Zhou, Zhenhong Sun, Xiangji Wu, and Junmin Wu. End-to-end optimized image compression with attention mechanism. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.