Compressed Representation of Neural Networks for Use Cases of Video/Image Compression in MPEG-NNR

Hyeoncheol Moon and Jae-Gon Kim
Korea Aerospace University
hcmoon@kau.kr, jgkim@kau.ac.kr

Abstract

MPEG-NNR (Compressed Representation of Neural Networks) aims to define a compressed and interoperable representation of trained neural networks. In this paper, a compressed representation of NN and its evaluation performance along with use cases of image/video compression in MPEG-NNR are presented. In the compression of NN, a CNN to replace the in-loop filter in VVC (Versatile Video Coding) intra coding is compressed by applying uniform quantization to reduce the trained weights, and the compressed CNN is evaluated in terms of compression ratio and coding efficiency compared to the original CNN. Evaluation results show that CNN could be compressed to about quarter with negligible coding loss by applying simple quantization to the trained weights.

1. Introduction

Neural networks have been adopted for a broad range of tasks in multimedia analysis and processing, media coding, data analysis and many other fields. Their recent success is based on the feasibility of processing much larger and complex neural networks than in the past, and the availability of large-scale training data sets. However, many applications require the deployment of a particular trained network instance across different NN frameworks, potentially to a larger number of devices, which may have limitations in terms of processing power, memory and interoperability. In order to address these issues, exchange formats that can be interoperable and optionally compressed by various deep learning frameworks models have been developed in industry standard groups [1], [2], and the standardization activity for these formats is called NNR and is underway in MPEG.

2. MPEG-NNR

MPEG recognizes the necessity of interoperable and/or compressed neural network models and is working on NNR activities [3]. The MPEG activity on Compressed Representation of Neural Network (NNR) aims to define a compressed, interpretable and interoperable representation for trained neural networks. In addition, for the requirements for the neural network may vary depending on the given use cases, the scenarios and requirements for the use cases are collected and summarized in [4], and the existing exchange formats for representing the models are also presented in [5].

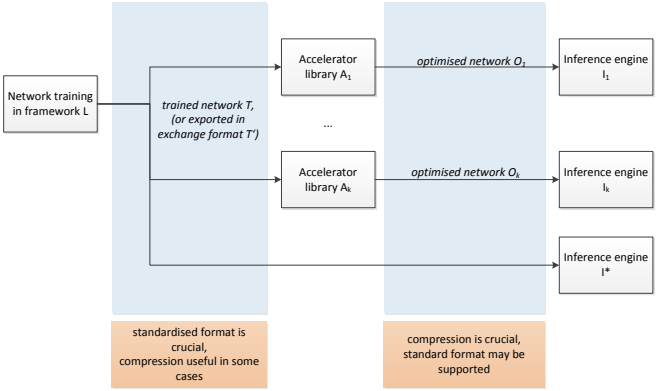

Fig. 1. MPEG-NNR Framework [4]

Fig. 1 shows the MPEG-NNR framework. When a trained model is sent to an acceleration library provided by a certain framework, the acceleration library optionally accelerates the processing of the network according to the requirements of each case, and the optimized network is transmitted to the inference engine in an integrated exchange format.

Fig. 2 shows the evaluation framework that confirms that the network compressed by the acceleration library in Figure 1 is applied well in a real case. *O_Per* and *R_Per* each of which represents the performance of the original network and the reconstructed network, respectively, should be as close as possible to each other, and *R_size* should be met the requirements for that use case.

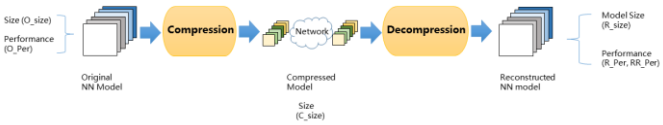

Fig. 2. Evaluation Framework [6]

3. Video/Image Compression Use Case

There are two types of use cases of video/image compression MPEG- tool-by-tool case and end-to-end case [4]. The tool-by-tool use case is used mainly in video compression in which certain coding tools are replaced by neural networks based methods. This use case requires information on which tool to be replaced, whether the tool should be replaced in the encoder and decoder or not. On the other hand, the end-to-end case applies neural network to replace the entire codec, and it is mainly used in image compression. It is based on an auto-encoder, so that the network model of the encoder and the decoder should be defined and replaced, respectively.

4. Compression of NN: Quantization

Quantization is used to compress the trained network model. In the experiment, a scalar quantization is applied to the trained weights, and the evaluation framework shown in Fig 3 is used. It is assumed that minimum value, maximum value, and the number of bits representing the trained weights are known in advanced. In the evaluation process, the quantized weights are dequantized to reconstruct the network model, and the performance of the original network and the reconstructed network including the quantization errors are compared in terms of coding efficiency..

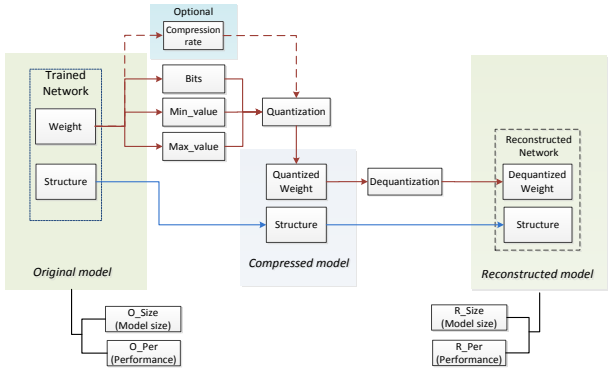

Fig. 3. Evaluation Framework - Quantization

5. Experimental Results

In this paper, the performance of the original network and the reconstructed network are compared in terms of coding efficiency for both cases of the tool-by-tool case and the end-to-end case.

In the tool-by-tool case, the in-loop filter of VTM 1.0 was replaced by CNN [7]. The CNN is applied in All Intra mode, which means that CNN is applied as a post-filter. The test sequences of Class B, C, and D in the JVET CTC [8] were used. In addition, testing environment is All Intra, the Structural Similarity Index Measure (SSIM)and PSNR were used to compare the original frame and the reconstructed frame.

Table 1 shows the comparison of model size between the original and the compressed model. The compression ratio is about 3.89. The reason why the compression ratio of the model size is not exactly 4 is that the model includes structural information as well as the weight.

Table 1. Comparison Model Size

	Original (32 bits)	Compressed (8 bits)
Model size	781KB	204KB

As shown in Table 2 and Table 3, the CNN based in-loop filter give better coding performance than the existing one in VTM 1.0 with gains of 0.21 dB in PSNR and 0.019 in SSIM. In addition, there is a minor performance loss when the reconstructed CNN is applied to in-loop filtering in video compression. In other words, the compression of network model representation does not hurt the coding efficiency in this evaluation.

Table 2: Evaluation results (PSNR) – QP=32, 37

	In-loop filter	Original CNN [10]	Reconstructed CNN
Class B	34.23	34.29	34.27
Class C	33.28	33.41	33.39
Class D	33.02	33.19	33.18
Overall	33.52	33.73	33.71

Table 3: Evaluation results (SSIM) – QP=32, 37

	In-loop filter	Original CNN [10]	Reconstructed CNN
Class B	0.9588	0.9591	0.9589
Class C	0.9612	0.9653	0.9641
Class D	0.9631	0.9674	0.9665
Overall	0.9610	0.9639	0.9631

The end-to-end case is based on an auto-encoder, replacing both encoder and decoder with CNN. The CIFAR 10 (1,000 images in RGB format) were used as the test images. In the evaluation, the performance of the auto-encoder in both cases of the original representation and the reconstructed representation were compared with JPEG in terms of the PSNR and SSIM.

As shown in Table 4, the CNN-AE based image compression gives better coding performance than JPEG compression with the gains of 0.877 dB in PSNR and 0.0042 in SSIM. In addition, there is a minor performance lose when the reconstructed CNN-AE is applied to image compression with the end-to-end approach. In other words, the compression of network model representation does not hurt the coding efficiency in this end-to-end case.

Table 4: Evaluation results: bpp (bit per pixel)= 4.3

	JPEG	CNN-AE	Reconstructed CNN-AE
PSNR	28.244	29.121	29.103
SSIM	0.812	0.8541	0.8534

6. Conclusions

In this paper, we presented the evaluation results of the compressed representation of neural networks for the use case of image/video compression, which consists of a tool-by-tool case and end-to-end case. The evaluation was performed in accordance with the procedure of the evaluation framework. In both cases, the difference in performance between the original model and the reconstructed model is minor. Based on the evaluation results, it is note that scalar quantization works well for the compression of the trained weights in video/image compression cases.

However, this work has some limitations on compression methods and performance evaluation methods. In the future, the combination of multiple compression methods as well as enhancing quantization itself need to be studied.

ACKNOWLEDGMENT

This paper was supported by National Standards Technology Promotion Program of Korean Agency for Technology and Standards (KATS) grant funded by MOTIE (10084981).

References

[1] Available at [Online]: https://www.khronos.org/nnef

[2] Available at [Online]: https://onnx.ai/

[3] W. Bailer, S. Chun, “AHG on Coded Representation of Neural Networks,” ISO/IEC JTC1/SC29/WG11 m41942, Jan. 2018.

[4] W. Bailer, et al, “Use cases and requirements for compressed representation of neural networks,” ISO/IEC JTC1/SC29/WG11 N17740, July. 2018.

[5] W. Bailer, et al, “Neural Network Exchange format and Acceleration Libraries,” ISO/IEC JTC1/SC29/WG11 m44592, Oct. 2018.

[6] W. Bailer, et al, “Draft Evaluation Framework for Compressed Representation of Neural Networks,” ISO/IEC JTC1/SC29/WG11 N17750, July. 2018.

[7] H. C. Moon, J.-G. Kim, “CNN Based In-Loop Filter for Versatile Video Coding (VVC),” In Proc. The Korean Institute of Broadcast and Media Engineers Conference, Jeju, Korea, 2018. pp. 270-271.

[8] A. Segall, et al, “JVET common test conditions and evaluation procedures for HDR/WCG Video Coding,” JVET document, JVET-D1020, Oct. 2016.