Exploring Model Compression Techniques for Deep Learning based Image Compression Models

Jenny Yu, Rui Zhu, Parinita Edke

July 10, 2023

{jenniferjie.yu; v.zhu; parinita.edke}@mail.utoronto.ca

Abstract

Image compression is crucial as images continue to increase in number due to advancements in computing power and mobile camera capabilities. To save space and transfer images more quickly, it is necessary to reduce the size of digital images while maintaining their necessary information. Deep Neural Networks (DNNs) have become powerful tools for various computer vision tasks, including image compression. However, DNNs are computationally intensive, requiring acceleration techniques to meet increasing needs. These techniques can be categorized into three categories: memory compression, computational optimization, and dataflow optimization. Model compression falls under the category of memory compression and the goal is to achieve a model that is simplified from the original without significantly diminishing accuracy and with a reduction in size and/or latency from the original. This project focuses on model compression techniques, such as pruning and quantization, on image compression algorithms to compare the performance of the compressed models to the original models. Our results show that pruning has different effects when applying to different components of the image compression models.

1 Introduction

Image compression is essential nowadays and it has become an important topic as image processing applications come to age. Due to the fast improvements in computing power and mobile camera capabilities, there has been an exponential increase in images. Therefore, it is crucial to reduce the size of digital images while maintaining necessary information, allowing more photos to be stored in a given amount of disk space. As a result, image compression enables saving more images in a limited storage space and increases the speed of transferring images faster between sites.

There are two types of image compression - lossy and lossless [1]. In lossy compression, some of the original data is permanently discarded to reduce the file size. In lossless compression, only redundant metadata is removed without any permanent loss of original data. In this project, we will mainly focus on lossy methods since it is more widely used for storing images. The traditional lossy image compression methods, such as Discrete Cosine Transform (DCT) [2] and vector quantization [3], are deterministic which rely on image filters, discrete transformations and quantization. But those deterministic methods have limitations such as bad compression quality and low compression ratio. In recent years, Deep Neural Networks (DNNs) have emerged as a powerful tool for various computer vision tasks, including image compression [4].

However, DNNs are very computationally intensive. Therefore, existing techniques for DNN acceleration techniques must be revisited and evaluated to meet the increasing needs for DNNs. Acceleration techniques can be categorized into three categories: memory compression, computational optimization and dataflow optimization [5]. Memory compression reduces memory usage while maintaining high solution quality which reduces energy consumption. Computation optimization utilizes mathematical transformations and circuit optimizations to minimize the number of mathematical operations required, reduce power consumption, and increase computational speed. Dataflow optimization uses

mapping, scheduling, and reordering data and/or operations to improve data reuse and eliminate redundant operations, resulting in increased energy efficiency and high throughput. In this project, we focus on memory compression techniques due to the high cost associated with transferring data to/from accelerating-chip memory, as well as the computation cost and the overhead of data transfer between different units [5].

Model compression falls under the category of memory compression, where the goal is to achieve a model that is simplified from the original without significantly diminishing accuracy and with a reduction in size and/or latency from the original. Some examples of model compression techniques involve pruning, quantization and knowledge distillation. By compressing the model, we can reduce its memory and computational requirements while still maintaining a decent image quality. This can lead to faster inference times, reduced energy consumption, and overall improved efficiency of image compression algorithms.

In short, the objective of this project is to use DNNs to perform lossy image compression with acceleration techniques using model compression and evaluate its effectiveness in comparison with existing image processing algorithms.

2 Prior Work

There has been a variety of work done previously in the space of DNNs for image compression.

2.1 GANs

High Fidelity Compression (HiFiC) proposed by Mentzer et al.[6] achieved state-of-the-art generative lossy compression results with conditional GANs [7] and a loss function that combines MSE with LPIPS [8], which measures the "perceptual distortion". Agustsson et al.[9] also proposed using GAN with entropy constraint on image representation for image compression of extremely low bitrate.

2.2 CNNs

Cavigalli et al presented a novel 12-layer deep convolutional neural network for image compression artifact suppression in JPEG images with hierarchical skip connections and a multi-scale loss function [10]. The result is a new state-of-the-art ConvNet achieving a boost of up to 1.79 dB in PSNR over ordinary JPEG and showing an improvement of up to 0.36 dB over the best previous ConvNet result.

Jiang et al presented a novel compression framework by seamlessly integrating two CNNs into an end-to-end compression framework [11]. The first CNN, called the compact convolutional neural network (ComCNN), learns an optimal compact representation from an input image, preserving the structural information and then encoding it using an image codec [11]. The second CNN, called the reconstruction convolutional neural network (RecCNN), then reconstructs the decoded image with high-quality in the decoding end [11]. The paper also presents an unified end-to-end learning algorithm to simultaneously learn ComCNN and RecCNN, facilitating the accurate reconstruction of the decoded image using RecCNN [11].

Prakash et al proposed a new CNN architecture directed specifically for image compression [12]. This technique makes JPEG content-aware by creating a map that highlights semantically-salient regions such that they can be encoded at a higher quality as compared to the background regions [12]. By improving the signal-to-noise ratio within multiple regions of interest, they are able to improve upon the visual quality of those regions, while preserving overall PSNR and compression ratio [12].

2.3 RNNs

Toderici, et al proposed a Recurrent Neural Network (RNN) architecture for lossy image compression [13]. This architecture was created to handle the limited network bandwidth of handheld devices. The architecture allows for the gradual improvement and refinement of the output image as more data is

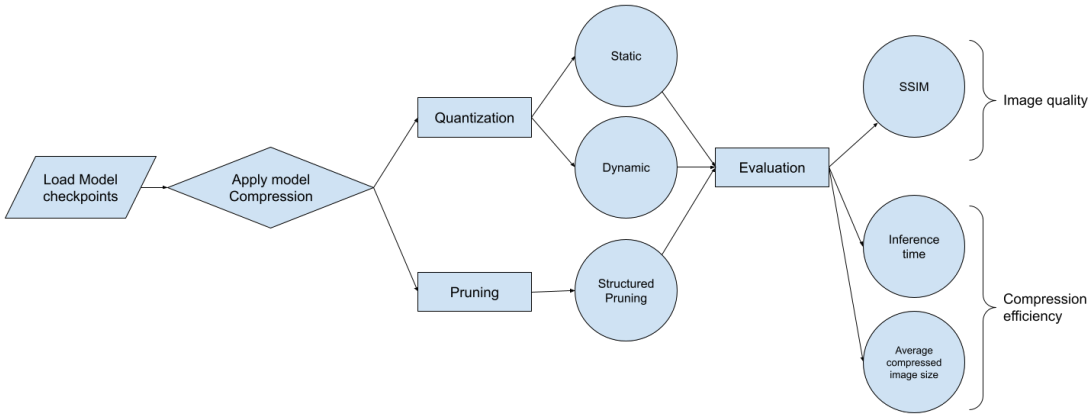

Figure 1: A system diagram of the method overview

received from the network. The network consists of a RNN-based encoder and decoder, binizer and entropy coding This RNN architecture outperforms JPEG method on Kodak dataset images.

2.4 Model compression

Tantawy et al. [5] summarized three techniques for model compression, especially memory compression for CNN-based GAN: pruning, where certain edges of the model are eliminated to make the model more light-weighted without much loss of results quality; knowledge distillation, where the knowledge learned by the original model (teacher model) is transferred to a smaller model (student model); lowering numeric precision, where fewer bits are used to represent a parameter in the model, is called quantization when the precision is lowered to an integer.

Novelty

We investigate the trade-off between size and latency reduction, and image quality on deep learning-based image compression approaches and extend previous research on model compression on a variety of image compression models by benchmarking the SOTA model compression approaches.

3 Methodology

For our project, the two models we picked to investigate are the HiFiC [6] and the RNN-based models [13]. A baseline is generated by testing on the two different image compression algorithms' checkpoints (HiFiC and RNN-based) with target bitrates of 0.30 and 0.375, respectively. The KODAK dataset is used to test the model performance, and performance metrics such as SSIM, total inference time, and average compressed image size are recorded.

After setting up a baseline, model compression techniques are applied to improve the baseline model. The two techniques used are model pruning, which involves pruning some connections within the network to boost robustness, and quantization, which involves using a reduced precision integer representation for the weights and/or activations. Finally, performance metrics are generated and compared with the baseline performance to evaluate the effectiveness of the compression techniques. By comparing the performance metrics, the impact of the compression techniques on the model's performance can be assessed. Please see the system diagram of the method overview in Figure 1.

3.1 Experimental models

We used checkpoints of existing image compression models: HiFiC [6] and an RNN image compression model proposed by Toderici et al. [13] and apply model compression methods [5, 14] including quantization and pruning on them. For the HiFiC architecture, the network consists of a convolution-based

encoder E, generator G, a probability model P for modeling posterior probability, and discriminator D for adversarial training; for the RNN-based architecture, the network consists of an RNN-based encoder and decoder, binarizer and entropy coding. Pruning was done on all the convolution layers and transpose convolution layers of the models since the majority layers of the models were convolutional layers and transpose convolution layers. Unstructured pruning was applied to different components of the model to see the effect of pruning on them.

3.2 Model compression techniques

Model pruning helps reduce the size of the network and can aid the model to generalize better. For the project, we utilized the unstructured pruning mechanism, which can be understood as finding and removing the less salient connection in the model wherever they are. Unstructured pruning does not consider any relationship between the pruned weights. Quantizing a network means converting it to use a reduced precision integer representation for the weights and/or activations. This saves on model size and allows the use of higher throughput math operations on your CPU or GPU. When converting from floating point to integer values, we essentially multiply the floating point value by some scale factor and round the result to a whole number. In dynamic quantization, the model parameters are known during model conversion and they are converted ahead of time and stored in INT8 form. The key idea is that the scale factor for activations is dynamically based on the data range observed at runtime. This ensures that the scale factor is "tuned" so that as much signal as possible about each observed dataset is preserved.

3.3 Datasets

Following the same setup of test set used in [6], we evaluate and compare the performance of the original models and the compressed models on the Kodak dataset (24 images) [15]. The Kodak dataset is a widely used benchmark dataset in evaluating image compression algorithms.

3.4 Metrics

We use MS-SSIM[16] for evaluating the compressed images' quality compressed by different models as they were widely used metrics. It serves as a metric for comparing the similarity of two images based on 3 characteristics: luminance, contrast, and structure. Total inference time, and average compressed image size are used for measuring the effectiveness of model compression on different models.

4 Results and Discussion

4.1 Pruning

For the RNN-based model, the pruning was done on the convolution layers. Different ratios of pruning were applied for different portions of the model (encoder, decoder). The results of applying pruning on the RNN-based model are shown in Table 1. The pruning did not reduce total inference time and no difference in performance was observed. However, pruning the encoder results in a lower compressed image size.

As for the HiFiC model, the pruning was done on both convolution layers and transpose convolution layers. The results of pruning the HiFiC model are shown in Table 4.1. The inference time was slightly reduced as the overall pruning ratio increased. The model's performance was worse when pruning was done on the generator. Pruning the encoder and the probability model caused the compressed image size to increase.

The RNN-based model and the HiFiC model had different behaviors after pruning. Some example images are shown in Figure 2. The RNN-based model had a significantly longer inference time than the HiFiC model. After pruning on both the encoder and decoder, the compressed image generated by the RNN-based model (Figure 2C) does not look different with the one generated by the unpruned RNN-based model (Figure 2B), which coincides with the SSIM score in Table 1. The compressed image generated by the RNN-based model (Figure 2B, 2C) also look more blurry in the details than

Table 1: Performance of the RNN-based model with pruning. EPR represents the encoder pruning ratio. DPR represents the decoder pruning ratio.

EPR	DPR	Total inference time	SSIM	Avg. compressed image size (B)
0	0	00:17:36	0.7886	17545.666
0.3	0	00:17:35	0.7858	17439.416
0	0.3	00:17:36	0.7876	17545.666
0.3	0.3	00:17:36	0.785	17439.416

Table 2: Performance of HiFiC model with pruning. EPR represents the encoder pruning ratio. GPR represents the generator (decoder) pruning ratio. PPR represents the probability model pruning ratio.

EPR	GPR	PPR	Total inference time	SSIM	Avg. compressed image size (B)
0	0	0	00:01:45	0.8179	18850.333
0.3	0	0	00:01:43	0.817	19048.167
0	0.3	0	00:01:40	0.7145	18850.333
0	0	0.3	00:01:40	0.8171	19192
0	0.3	0.3	00:01:40	0.7142	19048.167
0.3	0.3	0	00:01:41	0.7145	19048.167
0	0.3	0.3	00:01:40	0.7142	19192
0.3	0	0.3	00:01:42	0.8164	19429.5
0.3	0.3	0.3	00:01:37	0.7139	19429.5

the compressed image by HiFiC models (Figure 2D, 2E, 2F). However, after the generator (decoder) of the HiFiC model was pruned, the compressed image (Figure 2F) had a change in tone, but the details of the image were kept. The change in tone could potentially be mitigated by post-training image transformations. As this change in tone was not observed on the images generated by the RNN-based models, the changes caused by the pruning may be model-specific. Therefore, we recommend that users of model pruning techniques on image compression models examine the results images and then decide on using pruning to what extent.

4.2 Quantization

For the RNN-based model, as the model size decreases, the range of values is compressed resulting in a loss of important information, which leads to inaccurate outputs. It was evident that the output of the model after post-training static quantization was zero, and there are a few possible reasons for this behavior which are covered in the Limitations section. Quantization can not be applied to the HiFiC model as it is not implemented for transpose convolution layers.

5 Limitations and Future Work

The evaluations of the pruned model were done on GPU instead of CPU or edge devices, which is a possible reason why the inference time reduction brought by the pruning was not significant. In addition, due to the time constraint, we did not retrain the model using pruning-aware training, under which the models' performance should be evaluated in the future.

When performing quantization, we found that applying quantization resulted in inaccurate output images due to the compression of a range of values, which caused the loss of important information. Although higher precision for quantization may improve accuracy, it is currently unavailable in PyTorch. Furthermore, quantization cannot be applied to the HiFiC model as it lacks support for transpose convolution layers. Currently, static quantization only supports linear and convolutional layers (1D, 2D, and 3D), while dynamic quantization supports only linear layers, standard LSTM, and GRU. To further benchmark the effectiveness of quantization techniques on RNN-based image compression algorithms, a custom quantization function with custom operators and higher precision

should be implemented in the future.

Another potential future work that could be explored is the use of knowledge distillation for model compression [5]. Knowledge distillation is another model compression technique that trains a small model to achieve approximately the same performance as a large model by transferring knowledge [5]. The use of knowledge distillation for model compression has been shown to be effective in improving the performance of compression models in previous research. Therefore, it could be a promising area of future work to investigate for improving the HiFiC and RNN-based image compression models.

6 Conclusion

In conclusion, this paper explores DNN-based image compression methods with the use of model compression techniques. More specifically, model pruning and quantization techniques were applied to existing image compression models, HiFiC and RNN-based models, and their effectiveness was evaluated using MS-SSIM, total inference time and average compressed image size. We have shown that pruning has different effects when applying to different components of the image compression models. PyTorch implementation of quantization has limitations regarding change in precision and support of more complex modules. Future work should focus on implementing custom quantization functions with higher precision and custom operators to further evaluate quantization techniques. Other limitations and future works are also discussed including the need for evaluation on edge devices and investigation of knowledge distillation for model compression.

Figure 2: Examples of compressed images. **A.** Original image. **B.** Compressed image by original RNN-based model. **C.** Compressed image by RNN-based model with encoder pruning ratio=0.3, decoder pruning ratio=0.3. **D.** Compressed image by original HiFiC model. **E.** Compressed image by HiFiC model with encoder pruning ratio=0.3, probability model pruning ratio=0.3. **F.** Compressed image by HiFiC model with encoder pruning ratio=0.3, probability model pruning ratio=0.3, and generator pruning ratio=0.3.

References

[1] M. Rehman, M. Sharif, and M. Raza, "Image compression: A survey," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 7, pp. 656–672, 01 2014.

[2] N. Ahmed, T. Natarajan, and K. Rao, "Discrete cosine transform," *IEEE Transactions on Computers*, vol. C-23, no. 1, pp. 90–93, 1974.

[3] N. Akrout, R. Prost, and R. Goutte, "Image compression by vector quantization: a review focused on codebook generation," *Image and Vision Computing*, vol. 12, no. 10, pp. 627–637, 1994.

[4] S. Ma, X. Zhang, C. Jia, Z. Zhao, S. Wang, and S. Wang, "Image and video compression with neural networks: A review," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, pp. 1683–1698, jun 2020.

[5] D. Tantawy, M. Zahran, and A. Wassal, "A survey on gan acceleration using memory compression technique," 2021.

[6] F. Mentzer, G. Toderici, M. Tschannen, and E. Agustsson, "High-fidelity generative image compression," 2020.

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, pp. 139–144, 6 2014.

[8] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 1 2018.

[9] E. Agustsson, M. Tschannen, F. Mentzer, R. Timofte, and L. V. Gool, "Generative adversarial networks for extreme learned image compression," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2019-October, pp. 221–231, 4 2018.

[10] L. Cavigelli, P. Hager, and L. Benini, "Cas-cnn: A deep convolutional neural network for image compression artifact suppression," in *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 752–759, 2017.

[11] F. Jiang, W. Tao, S. Liu, J. Ren, X. Guo, and D. Zhao, "An end-to-end compression framework based on convolutional neural networks," 2017.

[12] A. Prakash, N. Moran, S. Garber, A. DiLillo, and J. Storer, "Semantic perceptual image compression using deep convolution networks," 2016.

[13] G. Toderici, D. Vincent, N. Johnston, S. J. Hwang, D. Minnen, J. Shor, and M. Covell, "Full resolution image compression with recurrent neural networks," 2016.

[14] A. Polino, R. Pascanu, and D. Alistarh, "Model compression via distillation and quantization," in *International Conference on Learning Representations*, 2018.

[15] E. Kodak, "Kodak PhotoCD dataset."

[16] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multi-scale structural similarity for image quality assessment," *Conference Record of the Asilomar Conference on Signals, Systems and Computers*, vol. 2, pp. 1398–1402, 2003.