DEEPCABAC: PLUG&PLAY COMPRESSION OF NEURAL NETWORK WEIGHTS AND WEIGHT UPDATES

David Neumann[1] *Felix Sattler*[1] *Heiner Kirchhoffer*[1] *Simon Wiedemann*[1]
Karsten Müller[1] *Heiko Schwarz*[1,2] *Thomas Wiegand*[1,3] *Detlev Marpe*[1]
Wojciech Samek[1]

[1]Fraunhofer Heinrich Hertz Institute, Berlin, Germany

[2]Institute of Computer Science, Free University of Berlin, Germany

[3]Department of Telecommunication Systems, Technical University of Berlin, Germany

ABSTRACT

An increasing number of distributed machine learning applications require efficient communication of neural network parameterizations. DeepCABAC, an algorithm in the current working draft of the emerging MPEG-7 part 17 standard for compression of neural networks for multimedia content description and analysis, has demonstrated high compression gains for a variety of neural network models. In this paper we propose a method for employing DeepCABAC in a Federated Learning scenario for exchange of intermediate differential parameterizations. Furthermore, we discuss the efficiency of DeepCABAC when compressing trained neural networks. Our experiments on large neural networks show that in both scenarios, DeepCABAC achieves competitive compression rates, without degrading the network accuracy.

Index Terms— Neural Networks, Federated Learning, Neural Network Compression, CABAC, Distributed Training

1. INTRODUCTION

Neural networks (NNs) have demonstrated impressive results in a multitude of machine learning (ML) tasks and are now being applied in a wide variety of applications. At the same time these models have grown in size with some of the latest state-of-the-art models containing up to multiple billions of parameters [1, 2]. Many application areas that could benefit from deep learning (DL) solutions however are highly resource constrained (e.g. mobile phones and IoT devices) and only have access to bandwidth-constrained communication channels. This often renders the usage of DL impractical in these situations.

At the same time, ML and communications are converging as new distributed training schemes such as peer-to-peer learning [3] and federated learning (FL) [4] emerge. In these settings, NN parameterizations need to be frequently communicated and the resulting communication overhead is typically the main limiting factor for the performance of distributed training solutions [5].

Research and industry have realized the need for compact and efficient NN representations and proposed specialized compression algorithms for different applications. Popular approaches for NN compression include pruning [6], distillation [7], and trained quantization [8], among others. In FL, quantization and sparsification methods [9, 10, 11] have been proposed alongside specialized solutions, such as federated dropout [5]. These methods, however, are all optimized only towards specific applications (e.g. FL) and/or require expensive re-training of the NN.

General, easy-to-use, and efficient compression methods applicable to different NNs within different usage scenarios are highly desired. Accordingly, a corresponding standardization activity within ISO/IEC MPEG towards compression of NNs is currently carried out. DeepCABAC [12], a recent algorithm that was adopted to the current working draft of the MPEG-7 part 17 standardization effort, has demonstrated great compression gains for a variety of NN models on the task of (trained) model compression. DeepCABAC is based on Context-based adaptive binary arithmetic coding (CABAC) [13], an entropy coding scheme widely used in video compression standards.

A number of favorable properties make DeepCABAC the ideal *universal* compression algorithm.

1. It is designed for the lossless compression of integers and can thus be combined with arbitrary quantization schemes.
2. It achieves a high compression efficiency when combined with a simple uniform reconstruction quantizer that only has one hyperparameter: The quantization strength.
3. It is adaptive towards any kind of tensor-shaped data, so many different kinds of neural data, e.g. weights or weight updates, can be compressed.

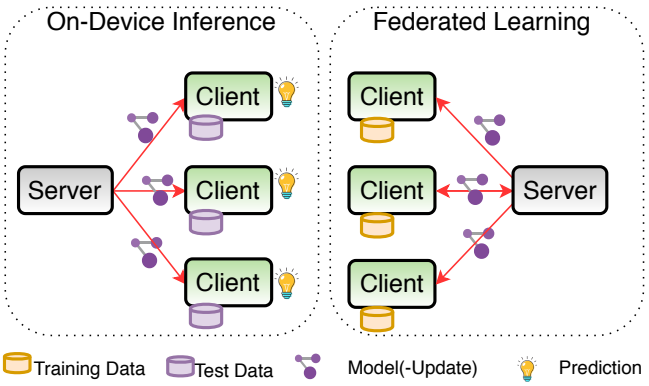

Fig. 1. Two common scenarios of NN communication: In on-device inference, a (typically fully trained) NN representation is communicated from a central server to a potentially large number of distributed devices. In Federated Learning a large number of devices jointly trains a NN on their combined data under the orchestration of a central server. This requires frequent down- and upload of NN parameterization from and to the server. DeepCABAC can accommodate both compression scenarios.

4. It is fast and efficient and does not require the compressed model to undergo expensive re-training.
5. It can be used in a plug & play fashion, i.e. can be easily integrated, e.g. in an existing FL pipeline.

In this paper, we demonstrate that DeepCABAC, without harming the performance of the models, provides excellent compression rates when it is applied to 1) trained NNs, e.g. reducing the size of a trained VGG16 model by 88.42%, as well as, 2) when applied to weight updates exchanged in FL settings, where it reduces the size of the communicated differential parameter updates by more than 96.5% on all tested architectures.

The remainder of this document is organized as follows: In section 2 we review the two most typical scenarios of NN communication. In section 3 we describe the DeepCABAC compression method and discuss how it can be applied in all these scenarios. Finally, in section 4, we perform experiments in the neural network compression and federated learning use-case, by applying DeepCABAC to trained models and differential parameter updates on a wide variety of neural network architectures.

2. COMPRESSION SCENARIOS

In the following we describe the most relevant scenarios, in which NNs need to be communicated. In NNs compression (figure 1, left), the sender communicates the parametrization of a (typically fully trained) model θ to one or more recipients. This form of model communication is often necessary

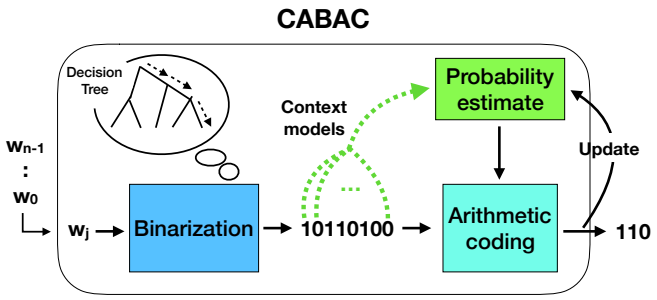

Fig. 2. Block diagram of DeepCABAC

if the NN was trained on a centralized server, but the specific application requires the model to be present on-device. The goal of model compression is usually to minimize the size of the NN representation while preserving it's performance (e.g. accuracy on the validation data).

If the recipient of the model already possesses an outdated parametrization θ' of the to be communicated model, it is often advantageous to instead communicate the differential model $\Delta\theta = \theta - \theta'$ (where the difference is taken element-wise for every tensor of parameters in the model), as the differential model typically has a much lower information content and is thus more amenable to compression.

Federated Learning [14] is a practical application area where NN models and differential models are communicated with a high frequency. In FL (figure 1, right), a number of client devices jointly trains a NN model on their combined data by alternating between local training and model aggregation. Every such communication round requires the clients to once download and upload a neural network parametrization from/to a centralized server. Efficient communication in FL is of paramount importance, because both the total number of communication rounds and the number of participating devices typically are very high, while at the same time mobile and embedded devices participating in FL often only have access to severely bandwidth-constrained mobile connections.

3. DEEPCABAC: A UNIVERSAL COMPRESSION ALGORITHM

For the efficient compression of NN parameters, DeepCABAC [12] has proven to be the most suitable arithmetic coding method. It is based on the CABAC scheme of the very successful video compression standards H.264/AVC and H.265/HEVC. We use a DeepCABAC configuration which comprises of the following building blocks (c.f. Figure 2):

- **Uniform Reconstruction Quantization (URQ):** Parameters are quantized to the nearest integer multiple of a predefined quantization step size. This also opens up the opportunity for an integer-based model inference,

which is usually less complex as, e.g., using floating-point operations.

- **Binarization:** Each quantized (integer) parameter is converted to a string of binary decisions (called "bins"). Concatenating the bin strings of all quantized parameters in a predefined order forms a binary representation of the model.
- **Context modeling:** Each non-bypassed bin, that means each bin that is not assumed to be uniformly distributed, is associated with one of a number of context models in a way that bins with similar statistics share the same context model. The context model estimates for each associated bin a probability distribution to be used for entropy coding in a backward-adaptive manner.
- **Entropy coding:** The multiplication-free binary arithmetic coder of CABAC (M coder), including its fast bypass mode, is employed to encode the sequence of bins.

Note that this version of DeepCABAC relies on only one hyperparameter (namely the quantization strength), and does not require expensive retraining of the network, which makes it particularly suitable for the application on resource-constrained distributed devices. We apply DeepCABAC to both trained neural networks and and differential parametrizations exchanged in federated learning, without making any modifications to the algorithm and only adjust the quantization step size of the URQ, in order to minimize the size of the compressed parametrizations while preserving the accuracy.

4. EXPERIMENTAL EVALUATION

4.1. Neural Network Compression

In this section, we present compression results for DeepCABAC (DC-v2) when applied to the compression of fully trained large-scale neural networks. We define the compression ratio (CR) as the fraction between the size of the compressed neural network and the size of the uncompressed version (for which all parameters are represented as 32 bit floating point numbers). Fig. 3 displays rate-distortion curves obtained by compressing three different neural network parametrizations, that were pretrained on the ImageNet [15] dataset, at different levels of quantization strength. The tag "DC" denotes networks compressed with DeepCABAC, whereas "Bl" denotes a baseline compression algorithm, for which we use bZip compression. As one can see, DeepCABAC consistently attains better compression results than the baseline across all levels of quantization strength. In practice, one is typically interested in compressed representations which preserve the accuracy of the original uncompressed model. We list the corresponding compression rates obtained by DeepCABAC for the three architectures in Table 1 along with additional results for Audio-Net and FACE.

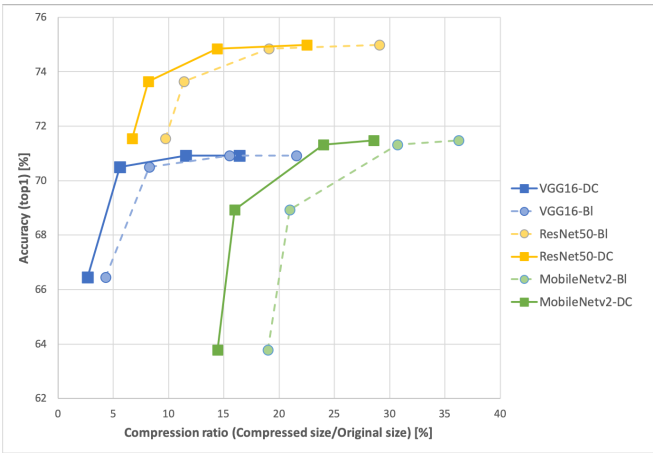

Fig. 3. Accuracy vs. byte-size points of different neural networks pretrained on the ImageNet dataset. The tag "DC" denotes networks compressed with DeepCABAC, whereas "Bl" denotes the baseline (Zip) compression algorithm. As one can see, DeepCABAC consistently attains better curves than the baseline.

Models	Org. size	Org. Acc. (top1 [%])	bZip (CR [%])	Cabac (CR [%])	Acc. (top1 [%])
VGG16	553.43 [MB]	70.93	15.52	**11.58**	70.92
ResNet50	102.23 [MB]	74.98	29.09	**22.52**	74.99
MobileNet-v2	14.16 [MB]	71.47	36.24	**28.57**	71.48
Audio-Net	467.27 [KB]	58.27	15.15	**10.93**	59.51
FCAE	304.72 [KB]	30.13 [PSNR]	39.28	**30.63**	30.17 [PSNR]

Table 1. Compression ratios achieved at no loss of accuracy when applying DeepCABAC to a wide set of neural network architectures trained on different tasks.

For the very popular language model BERT [16] DeepCABAC is able to attain a compression ratio of 9% at a F1-score of 86%, which is competitive with the reported results from the literature [17] (compression ratio of 11% at a F1-score of 89%). However, [17] attained these results after applying expensive retraining/fine-tuning procedures, plus a compression technique known as distilling [18] which modifies the network architecture. In contrast, DeepCABAC achieved the above results by simply applying URQ plus its entropy coding techniques.

4.2. Federated Learning

In order to evaluate the performance of DeepCABAC on the federated learning use case, we perform experiments on the commonly used CIFAR-10 dataset. We randomly split the training data into 10 disjunct shards, each containing 5000 data points, and assign each shard to a different client. The clients then jointly train a neural network model using the federated averaging algorithm [14] described in section 2. We configure the federated training in such a way that all 10 clients participate in every communication round and perform one epoch of local training using a batch-size of 128 and a fixed learning rate of 0.01.

To demonstrate the versatility of DeepCABAC, we train three different convolutional neural network architectures and measure the total number of bits communicated from all clients to the server during execution of the federated averaging protocol. For all networks, we quantize all weight layers of the differential models $\Delta\theta = \theta_i' - \theta_i$ to a 2 bit representation using nearest neighbor quantization and apply Cabac respectively bZip to the quantized values.

Figure 4 shows the resulting convergence speed in terms of accuracy per uploaded bits for LeNet, VGG11 and VGG16. For reference we also show the uncompressed baseline, where every parameter is communicated as a 32 bit floating point number. As we can see (c.f. table 2), for all model architectures, DeepCABAC distinctively outperforms bZip as encoding mechanism and is able to reduce the total communication to less than 4% of the original size on all architectures, without harming the convergence speed or the accuracy achieved by the final model.

These compression rates are also notably higher than the ones achieved in neural network compression (sec 4.1).

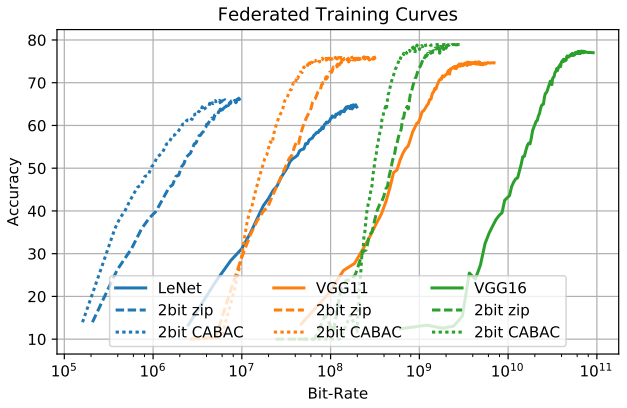

Fig. 4. Convergence speed w.r.t. communicated bits for LeNet, VGG11 and VGG16 trained using Federated Averaging with 10 Clients on CIFAR-10.

Models	Total Comm.	Org. Acc. (top1 [%])	bZip (CR [%])	Cabac (CR [%])	Acc. (top1 [%])
LeNet	197.25 [MB]	64.84	4.84	**3.29**	66.39
VGG-11	6.98 [GB]	74.91	4.90	**2.76**	76.15
VGG-16	90.86 [GB]	77.44	3.36	**2.30**	78.98

Table 2. Federated Learning with 10 Clients on the CIFAR-10 data set. Compression Results for 2 bit nearest neighbor quantization encoded with bZip and CABAC.

5. CONCLUSION

The compression of NN parametrizations is a young research field, which has recently gained a lot of attention by research, industry and in standardization. While a number of specialized solutions have been proposed for different use-cases over the last couple of years, there still remains the need for general and easy-to-use compression methods. In this paper we addressed this issue and presented DeepCABAC, a universal compression tool, which is currently the selected technology in the Motion Picture Experts Group (MPEG) standardization effort. We demonstrated that DeepCABAC can easily be integrated with distributed training pipelines and achieves highly competitive compression rates in both NN compression and federated learning.

6. REFERENCES

[1] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever, "Language models are unsupervised multitask learners," 2019.

[2] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro, "Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism," *arXiv e-prints*, p. arXiv:1909.08053, Sep 2019.

[3] Anusha Lalitha, Osman Cihan Kilinc, Tara Javidi, and Farinaz Koushanfar, "Peer-to-peer Federated Learning on Graphs," *arXiv e-prints*, p. arXiv:1901.11173, Jan 2019.

[4] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al., "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

[5] S. Caldas, J. Konečny, HB. McMahan, and A. Talwalkar, "Expanding the Reach of Federated Learning by Reducing Client Resource Requirements," *arXiv e-prints*, p. arXiv:1812.07210, Dec 2018.

[6] Yann LeCun, John S. Denker, and Sara A. Solla, "Optimal brain damage," in *Advances in Neural Information Processing Systems 2*, D. S. Touretzky, Ed., pp. 598–605. Morgan-Kaufmann, 1990.

[7] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean, "Distilling the knowledge in a neural network," in *NIPS Deep Learning and Representation Learning Workshop*, 2015.

[8] Song Han, Huizi Mao, and William J. Dally, "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding," *arXiv e-prints*, p. arXiv:1510.00149, Oct 2015.

[9] J. Konečný, HB. McMahan, F. X. Yu, P. Richtárik, A. Theertha Suresh, and D. Bacon, "Federated Learning: Strategies for Improving Communication Efficiency," *arXiv e-prints*, p. arXiv:1610.05492, Oct 2016.

[10] F. Sattler, S. Wiedemann, K. Müller, and W. Samek, "Sparse binary compression: Towards distributed deep learning with minimal communication," in *2019 International Joint Conference on Neural Networks (IJCNN)*, July 2019, pp. 1–8.

[11] F. Sattler, S. Wiedemann, K. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-i.i.d. data," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2019.

[12] S. Wiedemann, H. Kirchhoffer, S. Matlage, P. Haase, A. Marban, T. Marinc, D. Neumann, T. Nguyen, H. Schwarz, T. Wiegand, D. Marpe, and W. Samek, "Deepcabac: A universal compression algorithm for deep neural networks," *IEEE Journal of Selected Topics in Signal Processing*, pp. 1–1, 2020.

[13] D. Marpe, H. Schwarz, and T. Wiegand, "Context-based adaptive binary arithmetic coding in the h.264/avc video compression standard," *IEEE Trans. Cir. and Sys. for Video Technol.*, vol. 13, no. 7, pp. 620–636, July 2003.

[14] HB. McMahan, E. Moore, D. Ramage, Seth Hampson, and B. Agüera y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," *arXiv e-prints*, p. arXiv:1602.05629, Feb 2016.

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[17] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut, "Albert: A lite bert for self-supervised learning of language representations," *arXiv preprint arXiv:1909.11942*, 2019.

[18] G. Hinton, O. Vinyals, and J. Dean, "Distilling the Knowledge in a Neural Network," *arXiv preprint arXiv:1503.02531*, 2015.