Universal Deep Neural Network Compression

Anonymous Author(s)
Affiliation
Address
email

Abstract
In this paper, we investigate lossy compression of deep neural networks (DNNs) by weight quantization and lossless source coding for memory-efficient deployment. Whereas the previous work addressed non-universal scalar quantization and entropy coding of DNN weights, we for the first time introduce universal DNN compression by universal vector quantization and universal source coding. In particular, we examine universal randomized lattice quantization of DNNs, which randomizes DNN weights by uniform random dithering before lattice quantization and can perform near-optimally on any source without relying on knowledge of its probability distribution. Moreover, we present a method of fine-tuning vector quantized DNNs to recover the performance loss after quantization. Our experimental results show that the proposed universal DNN compression scheme compresses the 32-layer ResNet (trained on CIFAR-10) and the AlexNet (trained on ImageNet) with compression ratios of 47.1 and 42.5, respectively.

1 Introduction
Compression of deep neural networks (DNNs) has been actively studied in deep learning to develop compact DNN models for memory-efficient and computation-efficient deployment. Han et al. [1] showed impressive compression results by weight pruning, $k$-means clustering, and Huffman coding. It is further optimized in [2] using Hessian-weighted $k$-means clustering. Recently, it is shown how soft weight sharing or soft quantization can be employed for DNN weight quantization in [3, 4]. On the other hand, weight pruning is also extensively studied, e.g., in [5–9]. In this paper, we focus on DNN weight quantization, which can be used together with pruning to generate compressed models.

Vector quantization reduces the gap to the rate-distortion bound by jointly quantizing multiple symbols. Since conjectured by Gersho in [10], lattice quantization has been presumed to be the most efficient entropy coded vector quantization in the high resolution regime asymptotically, as the rate goes to infinity and the distortion diminishes [11]. Although lattice quantizers are simple and empirically shown to perform well even at finite rates, their efficiency depends on source statistics. Thus, we consider universal quantization that provides near-optimal performance for any source distribution [12]. Of particular interest is randomized lattice quantization, where uniform random dithering makes the distortion independent of the source, and the gap of its rate from the rate-distortion bound at any distortion level is provably no more than 0.754 bits per sample for any finite dimension, and no more than 0.5 bits per sample asymptotically as the dimension grows for any sources [13].

From the classical lossy compression results, this paper establishes a universal DNN compression framework consisting of universal quantization and universal lossless source coding such as Lempel–Ziv–Welch [14, 16] and the Burrows–Wheeler transform [17, 18]. In order to recover any accuracy loss resulting from weight quantization, we furthermore propose a fine-tuning algorithm for vector quantized DNNs. The gain of fine-tuning becomes larger as the vector dimension increases, due to the fact that the number of shared quantized values that are tunable (trainable) in a vector quantized model increases as the dimension increases.

Figure 1: Universal DNN compression by universal quantization and universal source coding.

2 Universal DNN compression

For vector quantization, given \( N \) weights denoted by \( w_1, \ldots, w_N \), we generate \( n \)-dimensional vectors \( v_1, \ldots, v_{[N/n]} \) by concatenating \( n \) distinct weights into one vector, e.g., as follows:

\[
v_i = [w_{ni−n+1} w_{ni−n+2} \cdots w_{ni}]^T, \quad 1 \leq i \leq \lfloor N/n \rfloor,
\]

where \( w_j = 0 \) for \( N + 1 \leq j \leq \lfloor N/n \rfloor n \) and \( \lfloor x \rfloor \) is the smallest integer larger than or equal to \( x \).

Vector quantization partitions these \( n \)-dimensional vectors into a finite number of clusters, and the vectors in each cluster share their quantized value, i.e., the cluster center.

- Randomized lattice quantization: Randomized lattice quantization [12] achieves universally good performance regardless of source statistics at any rates, and this leads us to the following universal DNN quantization method.

We randomize the \( n \)-dimensional vectors in (1) by adding uniform random dithers as follows:

\[
\tilde{v}_i = v_i + u_i, \quad u_i = [U_1 \ U_2 \ \cdots \ U_i]^T, \quad 1 \leq i \leq \lfloor N/n \rfloor,
\]

where each dithering vector \( u_i \) consists of \( n \) repetitions of a single uniform random variable \( U_i \) and \( U_1, \ldots, U_{\lfloor N/n \rfloor} \) are independent and identically distributed (i.i.d.) uniform random variables of support \([-\Delta/2, \Delta/2]\); \( \Delta \) is the quantization bin size. In each dimension, dithering values are i.i.d. uniform, which is sufficient to make the quantization error independent of source statistics.

- After dithering, uniform quantization in each dimension (i.e., lattice quantization) follows, i.e.,

\[
q_i = \Delta \ \text{round}(\tilde{v}_i/\Delta),
\]

where \( q_i \) is the quantized vector of \( \tilde{v}_i \); the rounding and scaling are element-wise operations in (3), where the rounding yields the closest integer values of the input.

Remark 1. Vector quantization theoretically provides a better rate-distortion trade-off. However, in practice, for compression of a finite number of data, the gain of vector quantization is limited by the codebook overhead, which becomes more considerable as dimension \( n \) increases and becomes the dominant factor that degrades the compression ratio after some point (see Figure 2 and compare the cases of \( n = 4 \) and \( n = 6 \)).

Fine-tuning of vector quantized DNNs: We fine-tune the vector quantization codebook to recover the loss after quantization. Each element of a shared quantized vector in the codebook is fine-tuned separately. That is, if we have \( k_{VQ} \) clusters of \( n \)-dimensional vectors, we effectively divide weights into \( nk_{VQ} \) groups and fine-tune their shared quantized values separately. The average gradient of the network loss function with respect to weights is computed in each group and used to update their shared quantized value, as will be clarified below.

Let \( c_i = [c_{i,1} \cdots c_{i,n}]^T \) be the shared quantized vector present in the codebook for cluster \( i \), and let \( I_{i,j} \) be the index set of all weights that are quantized to the same value \( c_{i,j} \) from (3), for \( 1 \leq i \leq k_{VQ} \) and \( 1 \leq j \leq n \). The gradient descent for the shared quantized value \( c_{i,j} \) is then given by

\[
c_{i,j}(t) = c_{i,j}(t−1) − \eta \frac{1}{|I_{i,j}|} \sum_{l \in I_{i,j}} \nabla w_l L(t − 1),
\]

where \( t \) is the iteration time, \( L \) is the network loss function, and \( \eta \) is the learning rate. The individual quantized vectors from (3) are also updated following their shared quantized vectors in the codebook.

In randomized lattice quantization, we would like to clarify that the average gradients are computed from the network loss function evaluated at the quantized weights after canceling random dithers (see (3)), while we fine-tune the shared values obtained before canceling random dithers (see (3)).
which yields the deployed weights of the universally quantized DNN at the inference step.

Universal source coding: Universal source coding algorithms are more convenient in practice than Huffman coding since they do not require us to estimate source statistics as in Huffman coding. Moreover, they utilize dictionary-based coding, where the codebook (i.e., dictionary) is built from source symbols adaptively in encoding and decoding, and therefore the codebook overhead is smaller than Huffman coding (see Remark 1).

For universal compression, the indices in the codebook of the lattice quantization output are passed as an input stream to a universal source coding scheme, which produces a compressed stream. The decoder needs to deploy the codebook that contains the indices and their corresponding fine-tuned shared quantized values for decompression.

Decompression: Using randomized lattice quantization, the encoder and the decoder are assumed to share the information on random dithers or their random seed. Under this assumption, the decoder decompresses the fine-tuned lattice quantization output from the compressed stream and then cancels the dithering vectors to obtain

\[ \bar{v}_i = q_i - u_i, \]

which yields the deployed weights of the universally quantized DNN at the inference step.

3 Experiments and discussion

We evaluate our DNN compression scheme for the 32-layer ResNet [19] (ResNet-32) model on CIFAR-10 dataset [20]. We consider two cases for (randomized) lattice quantization where the uniform boundaries in each dimension are set from (a) \( \{ i\Delta, i \in \mathbb{Z} \} \) and (b) \( \{(2i+1)\Delta/2, i \in \mathbb{Z}\} \), respectively; \( \mathbb{Z} \) is the set of integers. The quantization bin size \( \Delta \) is the same for both cases, but case (a) has the zero at a bin boundary while in case (b) the zero element is at the center of the middle bin.

For unpruned models, we often have a high volume of weights concentrated around zero, and thus case (b) that assigns one bin to include all the weights near zero is expected to outperform case (a), which is aligned with our lattice quantization results in Figure 2. However, it is interesting to observe that randomized lattice quantization provides similarly good performance in both cases, which is the main benefit of randomizing the source by uniform dithering before quantization. Figure 2 also shows that vector quantization provides additional gain over scalar quantization particularly when the compression ratio is large.

Finally, Table 1 summarizes the compression ratios that we obtain from our universal DNN compression method for pruned ResNet-32 and AlexNet [21] models. The proposed universal DNN compression scheme with the bzip2 [18] universal source coding algorithm yields \( 47.10 \times \) and \( 42.46 \times \) compression for ResNet-32 and AlexNet, respectively. Compared with [1,2,4] which need to optimize and/or calculate source statistics for compression, we achieved a better trade-off between rate (compression ratio) and distortion (loss in accuracy) through the universal compression of DNNs.
References


