

Chain of Compression: A Systematic Approach to Combinationally Compress Convolutional Neural Networks

Yingtao Shen¹ Mingqing Sun¹ Jie Zhao² An Zou¹
¹Shanghai Jiao Tong University ²Microsoft Corporation

Abstract

Convolutional neural networks (CNNs) have achieved significant popularity, but their computational and memory intensity pose challenges for resource-constrained computing systems, particularly with the prerequisite of real-time performance. To release this burden, model compression has become an important research focus. Many approaches like quantization, pruning, early exit, and knowledge distillation have demonstrated the effect of reducing redundancy in neural networks. Upon closer examination, it becomes apparent that each approach capitalizes on its unique features to compress the neural network, and they can also exhibit complementary behavior when combined. To explore the interactions and reap the benefits from the complementary features, we propose the Chain of Compression, which works on the combinational sequence to apply these common techniques to compress the neural network. Validated on image-based regression and classification networks across different data sets, our proposed Chain of Compression can significantly compress the computation cost by 100-1000 times with an ignorable accuracy loss compared with the baseline model.

1. Introduction

Deep Learning models have gained considerable attention due to their wide applicability across diverse domains. However, deploying deep learning models on resource-constrained systems such as mobile and embedded systems is challenging because of its high computational and energy cost. (Ke et al., 2018). To address this challenge, a variety of compression techniques have been developed to reduce the computational cost of these intricate models.

These approaches operate at different granularities or stages, either offline or dynamically at runtime. For example, before the network is executed, knowledge distillation compresses the architecture of the neural network, pruning removes neurons within layers, and quantization reduces the hardware bits used in the computation of each neuron (Mishra et al.,

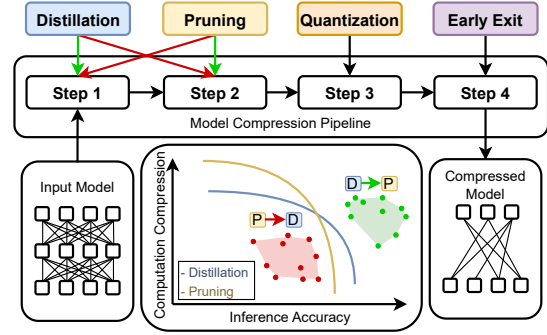


Figure 1. The Chain of Combination Framework: diverse compression techniques (e.g., distillation, pruning, etc.) can be integrated into the pipeline, forming a sequential chain.

2020; Neill, 2020). Meanwhile, dynamic inference techniques such as early exit or layer skipping actively compress the network structure, adapting to each input (Han et al., 2021) at the inference runtime.

Several studies have used multiple compression techniques, particularly for CNN-based models. For example, Qi et al. (2021) combine pruning and quantization to compress CNNs, achieving approximately a 50% reduction in FLOPs with only 0.15%-0.37% accuracy drops. Li et al. (2023) integrate early exit with quantization on CNNs, which not only avoids floating point computation but also leads to over 50% reduction in FLOPs with 1%-3% accuracy drops. More complex combinations include Deep Compression Han et al. (2015), employing pruning, trained quantization, and Huffman coding to minimize the storage of network parameters and the energy cost in the execution of neural networks; and the Deep Hybrid Compression Network (Zhao et al., 2023), using quantization, relaxed weight pruning, and knowledge distillation to overcome uniform quantization limitations.

However, most of the existing work is focused on proposing a possible combinational method, shedding little light on a general approach. Consequently, a crucial question arises: *is it necessary to use multiple compression approaches; if so what are the interactions between different compression approaches and what is the optimal sequence to apply different compression approaches?* In this work, we introduce the Chain of Compression, a novel perspective on combinational compression by treating each compression method

An Zou and Jie Zhao are the corresponding authors.

as a standard building block, as illustrated in Fig. 1. This modular design enables greater flexibility and adaptability in creating customized compression strategies customized to specific network architectures and performance requirements. Our objective is to devise a systematic method to find the best empirical combinational sequence, focusing on determining how compression method(s) should be chained together to achieve optimal performance and in what order. Specifically, we consider four typical types of compression approaches: knowledge distillation (Tian et al., 2020), pruning (Fang et al., 2023), and quantization (Zhou et al., 2016), which are mainstream compression methods (Mishra et al., 2020; Neill, 2020). Additionally, we incorporate early exit (Li et al., 2023) as a representative dynamic compression approach (Han et al., 2021).

Investigating the chain to compress the neural network in combination is far from trivial, as the number of combinations can increase exponentially when considering more configurations. To address this challenge, we propose to initially concentrate on examining the interaction and sequence of any two compression methods and comparing the pairwise order. This examination yields valuable insights that posit when pairwise orders are combined, they form a directed acyclic graph containing a single choice of topological sorting. This hypothesis is further corroborated through our extensive experiments, demonstrating that the order remains consistent as we insert other methods in between. Our final solution leverages the combinational law, employing multiple compression methods within an optimal combinational sequence. Experiments demonstrate that applying multiple compressions with the proposed optimal sequence can reduce the model size by 100-1000 times with an ignorable accuracy loss.

The contributions and insights of this paper are follows:

- Combining compression methods can introduce additional benefits. Despite each method’s performance varying greatly, they still complement each other. Using four methods altogether could be much more powerful than the best results of two methods combined.
- Order matters and the best-practice order between two methods are likely to be consistent if multiple compression methods are used together. This suggests a systematic way to explore the optimal sequence when compressing a network.
- The optimal sequence to apply multiple compression approaches is derived with topological sorting, which is from static to dynamic and granularity large to small.
- Multiple compressions with the proposed optimal sequence can reduce the model size by 100-1000 times with an ignorable accuracy loss.

2. Design Overview

Given the multitude of compression approaches available, it is impractical to exhaustively test every possible combina-

tion of these methods. Consequently, we pick four common compression techniques to investigate the interactions and determine the optimal sequence for applying multiple compression approaches. To overcome the need for brute-force exploration of every possible combination, we introduce a systematic roadmap for exploring the chain of compression for CNNs.

The Chain of Compression RoadMap:

- We start with the exploration of the interactions and the practical application sequence between any two compressions (in Section 3).
- We then demonstrate that inserting additional compression between any two compressions will not break the application sequence of these two compressions (in Section 4).
- Built on the application sequence between any two approaches, topological sorting can be used to generate the optimal sequence for applying multiple compressions (in Section 5).
- After the optimal sequence of multiple compressions is established, the necessity of repeatedly applying any compression is evaluated (in Section 6).

In the following, we briefly introduce the four compression approaches explored in this work: knowledge distillation, early exit, pruning, and quantization. In the present study, in order to ensure the broadest applicability of the compressed sequence obtained, it is imperative to extract the fundamental relationship between each compression method. Therefore, we have opted for utilizing the classic versions of the four compression methods and refrained from employing any advanced variants that are limited to specific scenarios.

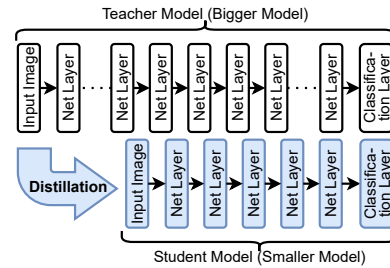


Figure 2. Knowledge Distillation.

Knowledge Distillation Neural network distillation, or knowledge distillation, is a method where knowledge from a larger, more accurate model (teacher model) is transferred to a smaller, more computationally efficient model (student model). The student model is trained not only on the task-specific labeled data but also on the distilled knowledge from the teacher model. By mimicking the teacher model’s output probabilities or representations, the student model becomes a compressed version, suitable for deployment on resource-constrained platforms. This technique offers advantages such as model compression, improved generalization, and facilitates transfer learning, making it valuable

for applications on mobile devices and other platforms with limited computational resources. The implementation of distillation in this work follows (Tian et al., 2020).

Early Exit Neural network early exit is a strategy in deep learning where intermediate exit points are introduced within the network architecture during the inference phase. These points allow the model to make predictions before reaching the final layers, based on criteria such as confidence levels. If a certain confidence threshold is met at an early exit point, the model stops processing and provides the prediction, reducing computation costs and enabling lower latency. This technique is valuable for real-time applications and scenarios with limited computational resources, offering a balance between efficiency and accuracy. In this work, the implementation of Early Exit is based on (Passalis et al., 2020; Li et al., 2023).

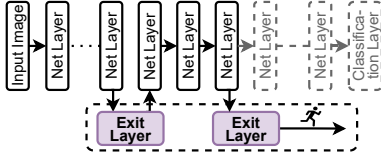


Figure 3. Early Exit.

Pruning Neural network pruning is a technique aimed at reducing the size and computational complexity of a neural network. It involves selectively removing less crucial connections, weights, or entire neurons from the model. This process results in a more compact network while maintaining or improving its performance. Pruning can occur during training or as a post-training optimization step, and it comes in various forms, including weight pruning, neuron pruning, and filter pruning in convolutional neural networks (CNNs). Key advantages include a smaller model size, reduced memory requirements, and faster inference times, making pruned models particularly suitable for resource-constrained environments and real-time applications. This work chooses channel pruning for consideration of hardware optimization difficulty and universality (Fang et al., 2023).

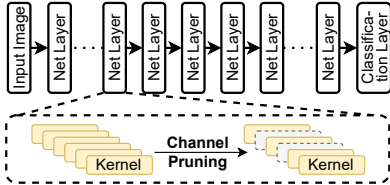


Figure 4. Pruning.

Quantization Neural network quantization is a technique that involves reducing the precision of weights and activations in a neural network. Typically implemented with lower bit-width representations, such as 8-bit integers, quantization significantly reduces the memory footprint of the model and accelerates inference. This process can occur during or after training and is effective in deploying models

on resource-constrained devices, such as edge devices or mobile platforms. Weight quantization focuses on reducing the precision of network weights, while activation quantization extends this to intermediate activation values during the forward pass. Post-training quantization is a common approach that applies quantization after the model has been trained with higher precision, striking a balance between model size and computational efficiency without compromising accuracy. This paper uses fixed-point uniform QAT (quantization-aware training) (Zhou et al., 2016) because it contains fine-tuning and thus has higher accuracy. also, fixed-point uniform quantization is more hardware-friendly and general.

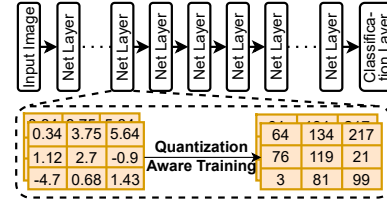


Figure 5. Quantization.

3. Interaction Between Two Approaches

In this section, we use ResNet34 (and CIFAR10 dataset) as a representative convolutional neural network to explore the interactions between two compression approaches. The key insights are also validated on VGG and MobileNet.

3.1. Complement and Sequence

In the following subsections, we present the performance of compression (noted as compression ratio) and inference accuracy of the neural network under any two compression approaches. In each scenario, we systematically investigate each compression approach by fine-tuning hyperparameters, which balance the tradeoff between compression ratio and inference accuracy. The performance resulting from the combination of two compression approaches is graphically represented with scatter points, each distinguished by tunable hyperparameters for the respective approach. To maintain methodological consistency throughout the study, we adhere to the following rules:

- The BitOps Compression Ratio (BitOpsCR) is used as the metric of compression performance, as we focus more on computation cost in this work and BitOpsCR is suitable for both static and dynamic compression methods. For storage metric, we use Compression Ratio (CR). To standardize the different bit widths between floating point and integer operations, we follow the same BitOps count strategy as (Li et al., 2019) and (Liu et al., 2021).
- After each compression operation we will instantly do fine-tuning. To be fair, we keep the same 200 training epochs for both model training (original model, early exit layer, and distillation) and fine-tuning (quantization-aware training and training after pruned).

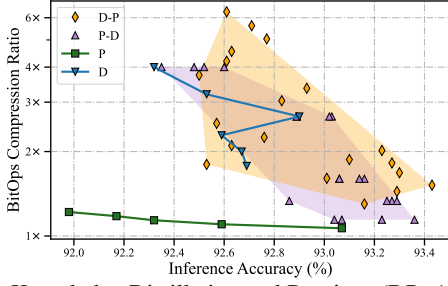


Figure 6. Knowledge Distillation and Pruning. (DP: After distillation, prune student model. PD: Before distillation, prune the teacher model first.)

ing) after each compression, while fine-tuning will have 1/10 of the initial learning rate.

- For each configuration, we train around 20 cases with the same training hyperparameters but different compression hyper-parameters for each compression method combination pipeline. Each case without early exit will provide one inference accuracy and BitOpsCR sample. Each case with Early Exit will provide several inference accuracy and BitOpsCR samples with different early exit threshold hyperparameters. We believe that doing so maximizes coverage of all cases where compression methods are combined with pipeline.

3.1.1. KNOWLEDGE DISTILLATION AND PRUNING

For the knowledge distillation and pruning shown in Fig. 6, the knowledge distillation achieves better BitOps compression ratios given the same inference accuracy. This is due to the fact that uniform channel pruning may not be equally effective for all scenes, as it comes with the price of high hardware compatibility. However, when both knowledge distillation and pruning are applied, the sequence of knowledge distillation followed by pruning consistently yields a better BitOps compression ratio while maintaining a comparable inference accuracy. Notably, the scatter plots for first knowledge distillation and then pruning mostly reside in the top-right corner, distinguishing them from the plots corresponding to the reverse sequence.

Intuitively, any form of compression before distillation may not appear logical. However, in contrast, (Aghli & Ribeiro, 2021) adopts the order of PD and claims that applying P first helps eliminate redundancies in the student model. Our experiment has shown that their assumption is partially correct, because DP performs even better without any redesigns.

3.1.2. KNOWLEDGE DISTILLATION AND QUANTIZATION

For the knowledge distillation and quantization shown in Fig. 7, quantization achieves better BitOps compression ratios given the same inference accuracy. When both knowledge distillation and quantization are combined, the sequence of knowledge distillation followed by quantization consistently yields a significantly better BitOps compression ratio while maintaining comparable inference accuracy. Similarly to DP, directly quantizing the student model removes redundancies

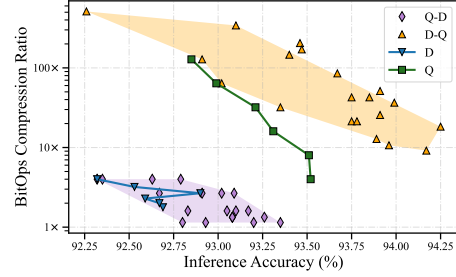


Figure 7. Knowledge Distillation and Quantization. (QD: Before distillation, quantize the teacher model. DQ: Quantize the student model after distillation.)

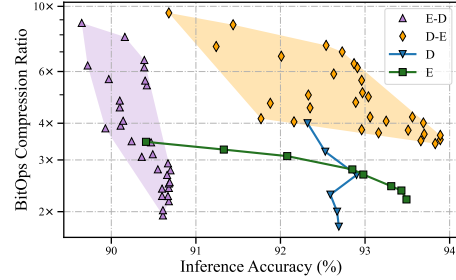


Figure 8. Knowledge Distillation and Early Exit. (ED: Train the teacher model’s exit layers first, then each teacher model’s exit layers together with final Softmax output serve as distillation head of student’s exit layers and body layers correspondingly. DE: After distillation, train exit layers on the student model.)

more efficiently.

3.1.3. KNOWLEDGE DISTILLATION AND EARLY EXIT

The performance of knowledge distillation and early exit is shown in Fig. 8. Knowledge distillation and early exit have relatively close BitOps compression ratios indicated by the cross of these two curves. Clearly, first the static compression approach knowledge distillation and then the dynamic compression early exit will achieve much better performance compared to the reverse sequence. The utilization of the exit layer in exit-aware distillation seems a false statement because the accuracy of the teacher model’s last layer return is typically better than most of exit layer returns, so the exit layer return becomes a worse teacher.

Training exit layer during distillation (use D as fine-tuning of exit layer) results in even lower accuracy. Failures of the train exit layer from the information of the teacher (ED or D as fine-tuning of E) show that the information of the student’s own body layer is more important for its exit layer.

3.1.4. PRUNING AND QUANTIZATION

The compression performance of pruning and quantization is presented in Fig. 9. Benefiting from BitOps as computation cost metric, the quantization achieves better compression than pruning (this is difficult to achieve in hardware, depends on the hardware support for bit operations).

Quantization limits the amount of information of all neurons in the network, which damages the high neuron resolution requirements of pruning (to select the least important batch

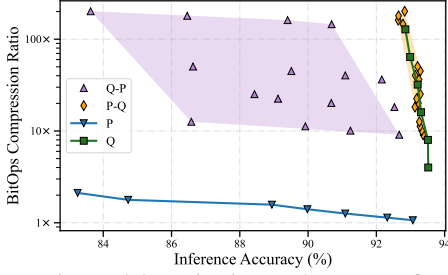


Figure 9. Pruning and Quantization. (PQ: Prune + fine-tune first, then quantize + fine-tune the model. QP: Quantize + fine-tune first, then prune and do QAT(Quantization-Aware-Training) as fine-tuning.)

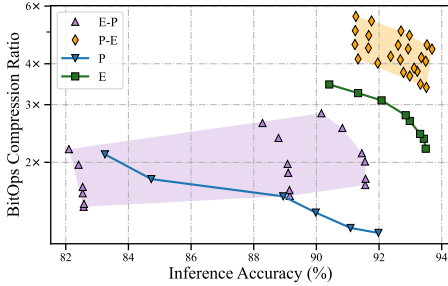


Figure 10. Pruning and Early Exit. (EP: Train exit layers first, then prune and fine-tune whole networks including exit layers. PE: Prune and fine-tune the original network first, then train unpruned exit layers.)

of neurons/filters/channels). This is corroborated by our experiment: When applying both pruning and quantization, first the neuron-level compression pruning and then the subneuron-level compression quantization achieves significantly better performance than the first quantization and then pruning.

3.1.5. PRUNING AND EARLY EXIT

The compression performance of pruning and early exit is presented in Fig.10. The exit layer allows inference to exit in the middle; therefore, to maintain good exit accuracy, the information density of the exit layer is typically larger than the body layer. It is safer to prune the body layer first as it has more redundancy. The experimental results also confirm this point: when both pruning and early exit are deployed, first pruning (static compression) and then early exit (dynamic compression) will perform significantly better than first early exit than pruning, even though the early exit compresses the neural network from a larger granularity.

3.1.6. QUANTIZATION AND EARLY EXIT

The performance of the last two compression approaches, quantization and early exit, is shown in Fig. 11. Individually, quantization performs better than early exit. When applying two compression approaches, first the static subneuron-level quantization and then the dynamic architectural level early exit will achieve better compression performance, especially with the requirement of high neural network inference accuracy.

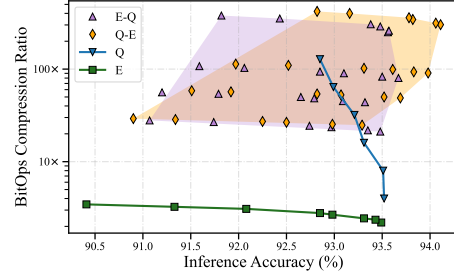


Figure 11. Quantization and Early Exit. (EQ: Train exit layers first, then quantize and fine-tune whole networks include exit layers. QE: Quantize and fine-tune original network first, then do QAT from scratch on exit layers.)

Unlike pruning, the exit layer itself should accept a quantized activation value. As a result, in QE even Q first, the later created exit layer should be quantized from the beginning and do QAT from scratch. The advantage of DE, QE, and PE indicates that fine-tuning on early exit layers is not as effective as on original network for accuracy recovery.

3.2. Summary of Interactions

Based on the explorations of any two compression techniques, we summarize the key insights here.

- A clear complementary feature is observed. Applying two compressions with the optimal sequence can achieve better compression performance compared to an individual single compression.
- The sequence of applying two compression approaches will directly impact the compression rate and inference accuracy.
- There is no evidence to support that first applying the compression with a large compression ratio and then applying the compression with a smaller compression ratio is an optimal sequence.
- First applying the compression working on large granularity then the compression on small granularity, or first applying the static compression then the dynamic compression can achieve better performance.

4. Adding Additional Compression

In this section, we aim to validate the integration of an additional compression step without disrupting the established interaction and sequence delineated in Section 3. The insertion of compression, whether positioned before or after the previously outlined two approaches, inherently modifies the input or output models. However, the impact on the established flow is comparatively less disruptive than introducing an additional compression step between the two already established approaches.

Since knowledge distillation should be placed ahead of other approaches as studied before, we study inserting quantization, pruning, and early exit into the other two compressions. Fig. 12 presents the performance of inserting quantization, pruning, or early exit into other two compressions.

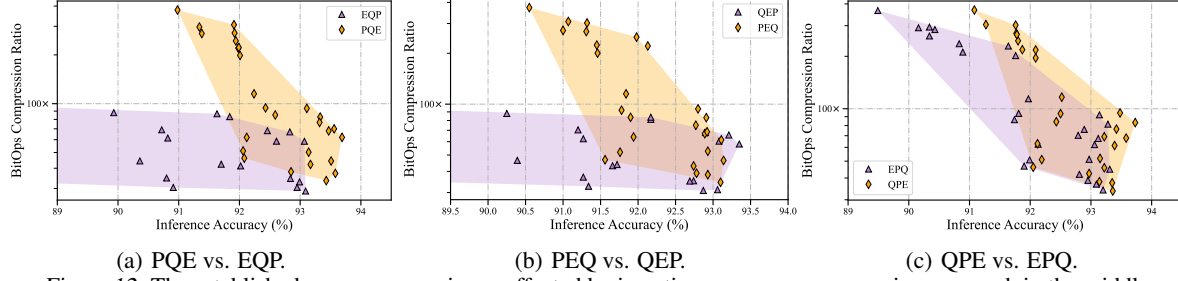


Figure 12. The established sequence remains unaffected by inserting one more compression approach in the middle.

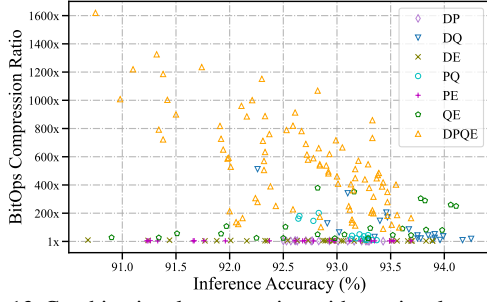


Figure 13. Combinational compression with previously established two approaches.

Clearly, after inserting another compression, pruning should be ahead of early exit, pruning should be ahead of quantization, and quantization should be ahead of early exit. Importantly, regardless of whether we insert quantization, pruning, or early exit, the established sequence remains unaffected compared to the one established in previous sections.

5. Combinational Sequence Law

Having established the sequence between each pair of compression approaches and validated that adding additional compression approaches will not break the sequence, we finally build the combinational sequence for applying multiple compressions using topological sorting. As the prerequisites of topological sorting, it is imperative to ensure the absence of cycles within the sorting process. Accordingly, we make the assumption that each compression approach is used only once in this section. In the following section, we will provide a brief discussion of the iterative application of compression approaches.

Following the established sequence in Section 3 and the topological sorting, the combination sequence to apply the compression approaches is

$$\begin{array}{ccccccc} \text{Distillation} & \text{Pruning} & \text{Quantization} & \text{Early Exit} \\ \text{(Static)} & \rightarrow & \text{(Static)} & \rightarrow & \text{(Static)} & \rightarrow & \text{(Dynamic)} \\ \text{(Architecture)} & & \text{(Neuron)} & & \text{(Sub-Neuron)} & & \text{(Architecture)} \end{array}$$

The topological sorting of the compression approach reveals a consistent adherence to the principles of transitioning from static to dynamic and progressing from a large granularity perspective to a small granularity one.

Fig. 13 presents the BitOps compression ratio and network accuracy using the full compression approaches (with combinational sequence) and the previously established two

Table 1. BitOps compression ratio of all distillation-started compression sequences (The original ResNet34 has an accuracy of 93.42%).

Acc. Loss	DPQE	DQPE	DPEQ	DQEP	DEPQ	DEQP
$\leq 0.2\%$	858 ×	555×	-	155×	217×	-
$\leq 0.6\%$	1068 ×	667×	221×	181×	245×	178×
$\leq 1.0\%$	1068 ×	923×	250×	181×	245×	181×
$\leq 2.0\%$	1235×	1238 ×	664×	223×	611×	231×

approaches in Sec. 3. Compared with the previously established two approaches, the combinational sequence may have slightly lower (less than 1%) reachable network inference accuracy. This is because the combination sequence incorporates pruning which limits the inference accuracy. If we compare the BitOps compression ratio within the reachable region of the combinational sequence (i.e., $\leq 93.5\%$), the combination sequence achieves a significantly improved BitOps compression ratio. For example, it can compress the neural network by about 1000 times with a network accuracy of 92.8% (i.e., 0.6% loss with the original network which has an accuracy of 93.42%¹. The accuracy is consistent with the classic ResNet.).

Table 1 provides a comprehensive comparison of the maximum achievable BitOps compression ratios under the optimal sequence law and other combination sequences, considering various tolerable accuracy losses. Given the demonstrated effectiveness of distillation as the initial compression step, we specifically evaluate sequences that start with distillation. The optimal sequence law, denoted as DPQE, consistently attains significant BitOps compression ratios across all acceptable accuracy loss thresholds. Comparatively, a sequence with closer alignment to the optimal sequence law (i.e., DQPE) achieves a secondary BitOps compression ratio. However, sequences such as DQEP and DEQP, which deviate significantly from the optimal sequence law, exhibit notably lower BitOps compression ratios. The performance across different sequences also proves the importance of the optimal sequence law in applying multiple compressions.

6. Repeating the Compression

As a complementary study to the combination sequence, we investigated the repeated application of compression

¹The original model of ResNet34 used in this work is the classic ResNet model in (He et al., 2015)

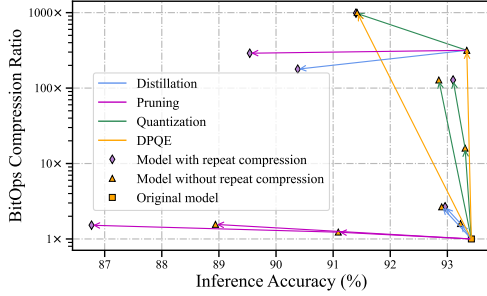


Figure 14. Repeating compressions.

approaches. Our exploration encompasses two scenarios: one involves the continuous repetition of a single compression method, while the other repeats a specific compression technique after employing the optimal sequence formed by combining all approaches. As the early exit is a dynamic compression at runtime that cannot be repeated, we do not consider the repeating of early exit. Fig. 14 presents the compression performance of repeatedly using one compression approach. In the figure, each marker represents a network model. The triangle marker means the model is not repeatedly compressed while the rhombus means the model is repeatedly compressed by one method. The line represents applying compression and the color of the line indicates the method of compression.

To assess the impact of continuous repetition of a single compression method, we additionally depict the compression performance using the corresponding single compression method with more aggressive hyperparameters to signify a higher BitOps compression ratio with increased accuracy loss. Notably, the results show that repeating distillation twice yields performance similar to applying distillation once with more aggressive hyperparameters. Conversely, repeating pruning twice leads to a diminished performance compared to applying pruning once with more aggressive hyperparameters. Repeating quantization twice results in a marginally improved performance compared to applying quantization once with more aggressive hyperparameters.

Starting from the combinational applied four compression approach with the optimal sequence (noted as DPQE), repeating either distillation or pruning will not decrease the compression performance. Repeating the quantization will increase the BitOps compression ratio but an obvious accuracy loss is reported. Although the continuous repetition of quantization could lead to better compression, repeating the quantization after four compression approaches with the optimal sequence breaks the sequence of two approaches in the previous sections where the quantization (static compression) should be applied before the early exit (dynamic compression). Therefore, repeating the quantization after the four compression approaches with the optimal sequence will not lead to improved compression performance.

Therefore, we can conclude that repeating the compression

process does not significantly enhance compression performance, except in the case of continuous repetition of quantization. The importance of maintaining an optimal sequence becomes evident. After employing the four compression approaches in the optimal sequence, repeating a single compression method may disrupt this sequence and result in a reduction of compression performance.

7. Evaluation

We perform an extensive evaluation of end-to-end performance on popular CNN network architectures, namely VGG-19, ResNet34 and MobileNetV2, across diverse benchmark datasets, including Cifar-10, Cifar-100 (Krizhevsky et al., 2009), SVHN (Netzer et al., 2011) and CINIC10 (Darlow et al., 2018). To ensure compatibility with the datasets, we utilized a modified version of MobileNetV2 introduced by (Ayi & El-Sharkawy, 2020). Given that MobileNetV2 scales primarily by width (as opposed to the depth of ResNet), the student model of MobileNetV2 maintained the same depth while featuring a reduced width (i.e., a lower channel number on each layer).

Table 2. Accuracy change and CRs on VGG19.

Dataset	Original Acc.(%)	Compressed Acc.(%)	BitOpsCR	CR
CIFAR10	93.40	93.24(−0.16)	500×	105×
CIFAR100	73.47	72.85(−0.62)	387×	104×
SVHN	95.86	94.78(−1.08)	927×	185×
CINIC10	84.25	83.65(−0.60)	484×	105×

Table 3. Accuracy change and CRs on ResNet34.

Dataset	Original Acc.(%)	Compressed Acc.(%)	BitOpsCR	CR
CIFAR10	93.42	93.33(−0.09)	859×	114×
CIFAR100	78.00	77.30(−0.70)	68×	14×
SVHN	94.99	95.35(+0.36)	1121×	114×
CINIC10	83.54	83.24(−0.30)	564×	114×

Table 4. Accuracy change and CRs on MobileNetV2.

Dataset	Original Acc.(%)	Compressed Acc.(%)	BitOpsCR	CR
CIFAR10	92.56	92.18(−0.38)	208×	35×
CIFAR100	74.88	74.11(−0.78)	138×	23×
SVHN	95.70	94.94(−0.76)	193×	35×
CINIC10	83.34	83.63(+0.29)	194×	35×

Tables 2, 3 and 4 summarize the compression performance of the proposed Chain of Compression on VGG19, ResNet34, and MobileNetV2 models across diverse benchmarks. The Chain of Compression achieves remarkable compression ratios ranging from hundreds to over 1000 times. Upon closer examination of the Chain of Compression’s performance on each dataset, a clear pattern emerges, indicating that more substantial compression is attainable for simpler classification tasks such as CIFAR10, SVHN, and CINIC10. Conversely, for the intricate classification task of CIFAR100, compression ratios tend to be smaller, accompanied by the possibility of an increase in accuracy loss. This observed trend aligns with the compression performance of each individual compression technique presented in the previous sections.

Fig. 15 presents the BitOps compression ratios and network inference accuracies resulting from applying each compression technique within the proposed Chain of Compression. In the compression process of the three models,

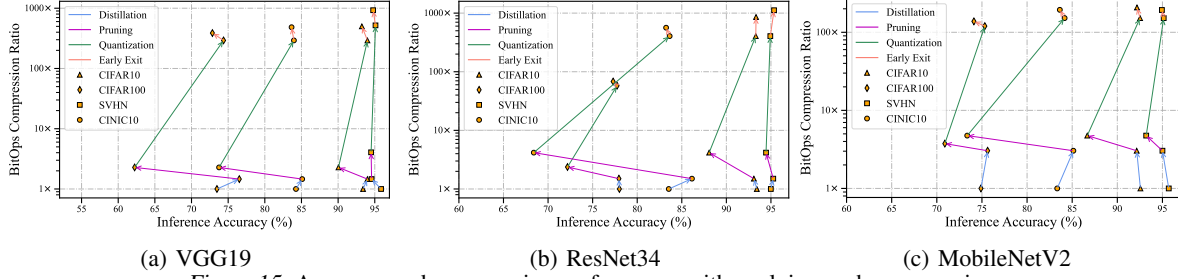


Figure 15. Accuracy and compression performance with applying each compression.

each compression technique plays a crucial role in reducing the computation cost of the neural network. A noteworthy observation is the automatic utilization of complementary features among these techniques within the proposed Chain of Compression. While one technique may lead to a slight reduction in network accuracy, the following compression techniques may effectively compensate for this loss while continuously decreasing the network computation costs.

Table 5. Comparison with other compression methods that do combination on multiple different compression techniques.

Dataset	Model	Method ¹	Acc. (%)	BitOpsCR	CR
CIFAR10	ResNet18	a. OICSR-GL	94.26(-0.40) ²	7.41	10.98
		b. PCQAT	85(-3)	-	11.82
		c. predictive E+Q	82(-3)	190.48	-
		d. CEA-MOP	92.01(-0.62)	1.70	1.62
	ResNet34	e. LQP	92.43(-0.43)	30.52	22.81
		Ours. DPQE	93.33(-0.09)	858.70	113.85
	ResNet110	f. PD	93.00(-1.27)	-	4.70
	ResNet164	f. PD	93.70(-0.82)	-	3.63
	ResNet56 ³	g. KDDP SN	81.93(-6.16)	-	20.37
		h. HMC	92.54(-0.89)	-	5.35
		i. DDSL	93.98(+0.17)	4.15	4.00
	wrn-40-16	j. Quantized Distill.	94.23(-1.47)	-	32.57
	VGG16	e. LQP	93.34(-0.21)	-	60.99
		k. Smart-DNN+	92.60(+0.00)	-	7.07
		l. hybrid search	93.95(-0.52)	4.50	9.79
		m. HFPQ	92.47(-1.11)	7.58	30.00
CIFAR100	VGG19	c. predictive E+Q	87(-1)	34.71	-
		Ours. DPQE	93.24(-0.16)	500.14	105.14
	ResNet34	c. predictive E+Q	60(-2)	86.49	-
		h. HMC	67.88(-0.94)	-	5.56
		Ours. DPQE⁴	77.3(-0.7)	68.07	14.21
		Ours. DPQE	73.5(-4.5)	306.88	64.17
	wrn-28-10	j. Quantized Distill.	76.31(-0.90)	-	17.80
	VGG16	k. Smart-DNN+	66.01(-0.03)	-	6.00
	VGG19	c. predictive E+Q	62(-2)	20.94	-
		Ours. DPQE	72.85(-0.62)	387.26	104.45

¹ a. OICSR-GL (Qi et al., 2021): structured sparsity regularization + greedy out-in-channel P. b. PCQAT (Predic et al., 2022): weight clustering + QAT. c. predictive E+Q (Li et al., 2023): predictive exit + PTQ. d. CEA-MOP (Zhang et al., 2022): NAS + filter P + coding. e. LQP (Idelbayev & Carreira-Perpinán, 2021): low-rank approximation + P + Q. f. PD (Aghli & Ribeiro, 2021): weight P via activation analysis + D. g. KDDP SN (SENER & AKBAS, 2022): D + L1 regularization + P on fully connected. h. HMC (Yang et al., 2023): low-rank tensor decomposition + structured P. i. DDSL (Liu et al., 2022): alternating direction method of multiplier based D + P. j. Quantized Distill. (Polino et al., 2018): D + Q. k. Smart-DNN+ (Wu et al., 2023): Q + bucket coding. l. hybrid search (Li et al., 2024): NAS + P. m. HFPQ (Fan et al., 2021): channel P + exponential Q.

² Since the accuracy of the original model reproduced in each work is different, both post-compression accuracy and accuracy loss play crucial roles in the comparison. Poor model accuracy reproduction can mitigate the compression accuracy loss, while it cannot conceal a very low post-compression accuracy. Only the coexistence of high post-compression accuracy and low accuracy loss can indicate that a compression achieves near-lossless.

³ ResNet56 is a Cifar-style variant of ResNet.

⁴ DPQE in this line applies 4w8a (4-bit weight, 8-bit activation) Q, trades a reduction in CRs for superior post-compression accuracy. Other lines of DPQE in this table apply 1w8a Q.

Table 5 presents a comparison of proposed Chain of Compression with other state-of-the-art methods that integrate two or more model compression techniques, such as Distil-

lation, Pruning, Quantization, Early Exit, Coding, Network Architecture Search (NAS), and Low-rank Matrix Factorization. Unlike some of these methods, to maximize generality and hardware friendliness, our chosen compression element is typical and not tailored to a specific network. However, as indicated in the table, proposed Chain of Compression still demonstrates superior performance compared to all other methods. On CIFAR10, proposed Chain of Compression exhibits outstanding BitOps compression while also achieving favorable model size reduction. On CIFAR100, proposed Chain of Compression maintains superior compression performance and achieves higher post-compression accuracy than others when compressing the same network.

8. Related Work

To support the neural network on lightweight computing platforms, several compression techniques have been proposed in recent years. These techniques utilize different perspectives, such as network architecture, neuron, and bits precision, to compress DNN with a minimal accuracy compromise (Mishra et al., 2020; Neill, 2020). Meanwhile, characterized by fixed computational graphs and parameters during the inference stage, dynamic compression exhibits the ability to adapt network structures or parameters to varying inputs at runtime (Han et al., 2021). In applying multiple compressions, (Qi et al., 2021) combine pruning and quantization to compress CNNs and achieve about 50% reduction in FLOPs with only 0.15% to 0.37% accuracy drops. (Li et al., 2023) combine the early exit with quantization on CNN, which not only avoids floating-point computation, but also reduces over 50% reduction in OPs with 1% - 3% accuracy drops. Zhao (Zhao et al., 2023) presents the Deep Hybrid Compression Network. It uses relaxed mixed-precision quantization, relaxed weight pruning, and knowledge distillation to overcome the limitations of uniform quantization. (Han et al., 2015) leverage pruning, trained quantization, and Huffman coding to reduce the network parameter storage and energy cost in executing the neural network. Although diverse compression methods are used on CNNs, limited research explores their interactions and optimal sequencing. Motivated by the potential to significantly reduce computation and parameter costs, this study investigates the interplay among compression techniques and establishes an optimal sequence for CNN compression.

9. Conclusion

To explore interactions and reap the benefits of applying multiple compression techniques, we present a study on the optimal combinational sequence to compress the neural network. Through the proposed roadmap we demonstrate the significant benefits of applying multiple compression approaches and also the importance of the optimal compression sequence. Based on the interactions between two compressions, we establish the optimal combinational sequence law in applying multiple compression technique. Validated on image-based regression and classification networks across different data sets, our proposed combinational sequence can significantly reduce the computation cost up to 1000 times with ignorable accuracy losses compared to baseline models.

References

- Aghli, N. and Ribeiro, E. Combining weight pruning and knowledge distillation for cnn compression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3191–3198, 2021.
- Ayi, M. and El-Sharkawy, M. Rmnv2: Reduced mobilenet v2 for cifar10. In *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 0287–0292, 2020. doi: 10.1109/CCWC47524.2020.9031131.
- Darlow, L. N., Crowley, E. J., Antoniou, A., and Storkey, A. J. Cinic-10 is not imagenet or cifar-10. *arXiv preprint arXiv:1810.03505*, 2018.
- Fan, Y., Pang, W., and Lu, S. Hfpq: deep neural network compression by hardware-friendly pruning-quantization. *Applied Intelligence*, pp. 1–13, 2021.
- Fang, G., Ma, X., Song, M., Mi, M. B., and Wang, X. Dep-graph: Towards any structural pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16091–16101, 2023.
- Han, S., Mao, H., and Dally, W. J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.
- Han, Y., Huang, G., Song, S., Yang, L., Wang, H., and Wang, Y. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7436–7456, 2021.
- He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition, 2015.
- Idelbayev, Y. and Carreira-Perpinán, M. A. More general and effective model compression via an additive combination of compressions. In *Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021, Bilbao, Spain, September 13–17, 2021, Proceedings, Part III 21*, pp. 233–248. Springer, 2021.
- Ke, L., He, X., and Zhang, X. Nnest: Early-stage design space exploration tool for neural network inference accelerators. In *Proceedings of the International Symposium on Low Power Electronics and Design*, pp. 1–6, 2018.
- Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. *Technical report, University of Toronto*, 2009.
- Li, G., Tang, L., and Zheng, X. Fast hybrid search for automatic model compression. *Electronics*, 13(4):688, 2024.
- Li, X., Lou, C., Chen, Y., Zhu, Z., Shen, Y., Ma, Y., and Zou, A. Predictive exit: Prediction of fine-grained early exits for computation-and energy-efficient inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 8657–8665, 2023.
- Li, Y., Dong, X., and Wang, W. Additive powers-of-two quantization: An efficient non-uniform discretization for neural networks. In *International Conference on Learning Representations*, 2019.
- Liu, X., Ye, M., Zhou, D., and Liu, Q. Post-training quantization with multiple points: Mixed precision without mixed precision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 8697–8705, 2021.
- Liu, Y., Cao, J., Li, B., Hu, W., and Maybank, S. Learning to explore distillability and sparsability: a joint framework for model compression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3378–3395, 2022.
- Mishra, R., Gupta, H. P., and Dutta, T. A survey on deep neural network compression: Challenges, overview, and solutions. *arXiv preprint arXiv:2010.03954*, 2020.
- Neill, J. O. An overview of neural network compression. *arXiv preprint arXiv:2006.03669*, 2020.
- Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. URL http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf.
- Passalis, N., Raitoharju, J., Tefas, A., and Gabbouj, M. Efficient adaptive inference for deep convolutional neural networks using hierarchical early exits. *Pattern Recognition*, 105:107346, 2020.

- Polino, A., Pascanu, R., and Alistarh, D. Model compression via distillation and quantization. *arXiv preprint arXiv:1802.05668*, 2018.
- Predić, B., Vukić, U., Saračević, M., Karabašević, D., and Stanujkić, D. The possibility of combining and implementing deep neural network compression methods. *Axioms*, 11(5):229, 2022.
- Qi, Q., Lu, Y., Li, J., Wang, J., Sun, H., and Liao, J. Learning low resource consumption cnn through pruning and quantization. *IEEE Transactions on Emerging Topics in Computing*, 10(2):886–903, 2021.
- Tian, Y., Krishnan, D., and Isola, P. Contrastive representation distillation. In *International Conference on Learning Representations*, 2020.
- Wu, D., Yang, W., Zou, X., Xia, W., Li, S., Hu, Z., Zhang, W., and Fang, B. Smart-dnn+: A memory-efficient neural networks compression framework for the model inference. *ACM Trans. Archit. Code Optim.*, 20(4), oct 2023. ISSN 1544-3566. doi: 10.1145/3617688. URL <https://doi.org/10.1145/3617688>.
- Yang, G., Yu, S., Yang, H., Nie, Z., and Wang, J. Hmc: Hybrid model compression method based on layer sensitivity grouping. *Plos one*, 18(10):e0292517, 2023.
- Zhang, Y., Wang, G., Yang, T., Pang, T., He, Z., and Lv, J. Compression of deep neural networks: bridging the gap between conventional-based pruning and evolutionary approach. *Neural Computing and Applications*, 34(19): 16493–16514, 2022.
- Zhao, Z., Ma, Y., Xu, K., and Wan, J. Deep hybrid compression network for lidar point cloud classification and segmentation. *Remote Sensing*, 15(16):4015, 2023.
- Zhou, S., Wu, Y., Ni, Z., Zhou, X., Wen, H., and Zou, Y. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016.
- ŞENER, E. and AKBAŞ, E. Improved knowledge distillation with dynamic network pruning. *Gazi University Journal of Science Part C: Design and Technology*, 10(3): 650–665, 2022. doi: 10.29109/gujsc.1141648.