VOL. 25, NO. 2, 2026, 113-119
elektrika.utm.my
EISSN 3083-9394

ELEKTRIKA
Journal of Electrical Engineering

The Influence of Fine-Tuning on Design Space Exploration in Joint Pruning-Quantization

Muhammad Nor Azzafri Nor-Azman[1], **Usman Ullah Sheikh**[1] and **Muhammad Nadzir Marsono**[1*]

[1]Faculty of Electrical Engineering, Universiti Teknologi Malaysia, 81310 UTM Skudai, Johor, Malaysia.

[*]Corresponding author: mnadzir@utm.my

Abstract: Joint pruning-quantization is now a widely adopted method for reducing complexity of deep neural networks. Specifically, layer-wise compression has achieved significant success by exploiting each layer's varying sensitivity to pruning and quantization. Although effective at reducing complexity with minimal accuracy loss, this approach significantly complicates the design space exploration. Prior works estimate the order of complexity of the design space based on the number of pruning ratios, quantization bitwidth for weights and activations, layers, model variations, and hardware targets. However, none have considered the number of fine-tuning epochs required after joint pruning-quantization. This work hypothesizes that fine-tuning is often excluded from design space estimates because accuracy gains after the first epoch are minimal. To validate this, we identify layers in ResNet-20 that are highly sensitive to compression and evaluate their behavior under varying pruning ratios, bitwidth, and fine-tuning epochs. Results confirm that epochs beyond the first only provide marginal improvements, although fine-tuning is crucial for accuracy recovery. These findings support treating fine-tuning as a binary decision (applied or not) rather than a continuous design parameter, thus justifying its exclusion from the order of complexity in design space estimation.

Keywords: CNN, Deep Learning, Pruning, Quantization

Article History: received 4 June 2025; accepted 13 May 2026; published 31 August 2026
Digital Object Identifier 10.11113/elektrika.v25n2.754

1. INTRODUCTION

Deep learning (DL) has experienced remarkable success across various fields [1, 2]. Nevertheless, DL models demand substantial computation and memory [3]. This hinders their deployment on resource-constrained edge devices [4] or increases the total cost of ownership in data centers [5]. Prior works have proposed joint pruning-quantization (e.g., [6, 7]), which prunes low-ranking weights and quantizes its bitwidth for more efficient computation. Earlier approaches employed global pruning and quantization [7, 8, 9], giving each layer identical pruning and quantization configuration. Such uniform compression can be suboptimal: if compressed aggressively, accuracy drops sharply; if compressed conservatively, the model would be less lightweight. This is because each layer has different sensitivity to pruning and quantization [6, 10]. This motivates layer-wise joint pruning–quantization, which prunes and quantize more aggressively in non-sensitive layers and retains more filters or higher precision in sensitive layers for better accuracy.

Unfortunately, layer-wise joint pruning-quantization expands the design space complexity considerably. The search order of complexity grows from $O(PQ_aQ_w)$ in the global case to $O(PQ_aQ_w)^L$ in the layer-wise case, where P is the set of pruning ratios and Q_a, Q_w are possible activation and weight bitwidth, and L is the number of layers. An exhaustive search is therefore impossible. For instance, with 19 convolutional layers in ResNet-20, ten possible pruning ratios (0–90 % in 10 % increments) and 8 bit-width options for both activations and weights (1–8 bits), exhaustive search would require 2.08×10^{53} configurations. At an average of 2.5 seconds per simulation on our system, this would take an impractical 16454 tredecillion years. To the extreme, some works (e.g. HAQ [11]) include model variations, V and target hardware platforms, T in the search. Although they only perform quantization, the order of complexity that could be interpreted would be $O(V \times T \times (PQ_aQ_w)^L)$. However, to the best of our knowledge, none have explicitly considered fine-tuning epochs as a parameter in estimating the order of complexity. Since pruned and quantized models often recover accuracy through fine-tuning [12, 13], fine-tuning epochs, F could in principle form an additional design dimension. This study evaluates whether that dimension is necessary, and makes two contributions:

1. We show that fine-tuning does not need to be treated as a separate design dimension when estimating design space complexity in layer-wise joint pruning-quantization, because the recoverable accuracy largely saturates after one epoch.
2. We provide layer-wise sensitivity profiling under pruning-only, quantization-only, and joint pruning-quantization settings.

2. RELATED WORKS

Numerous studies have proposed methods to navigate joint pruning-quantization design space efficiently. Several approaches (e.g., [14, 15, 16]) separate pruning and quantization into sequential stages. Pruning is performed first, then quantization on the pruned model. Separating the stages reduces the search space to $O(P^L + (Q_a Q_w)^L)$, but it fails to fully exploit the complementary nature of pruning and quantization [17]. Any connection pruned in the first stage cannot be *unpruned* or recovered in the quantization stage, potentially locking the optimization to a local minima. To address this, Tung *et al.* proposed to prune and quantize simultaneously in one training stage [18] achieving higher accuracy across multiple models. Other single-stage approaches treat pruning as a form of *zero-bit* quantization [19, 20], effectively performing only quantization. This simplification narrows the design space to $O(Q_a Q_w)^L$, but it can lead to suboptimal outcomes due to granularity issues. For example, allocating a single bitwidth per layer might force overly aggressive pruning (zeroing a whole layer), while allocating per-channel greatly expand the search space to $O(Q_a Q_w)^C$, where C is the total number of channels in the model and is always much larger than L.

Most works (e.g., [10, 18, 19]) acknowledge the huge design space in layer-wise compression. Wang *et al.* [11] estimated the complexity order as $O(V \times T \times (PQ_a Q_w)^L)$. Table 1 compares their estimate with the objective of this study, which is to assess whether fine tuning epochs should be included as an additional design dimension. As noted, adding V and T dimensions increase the design space further, yet no prior work includes fine-tuning in their estimates to the order of complexity for layer-wise joint pruning-quantization. This study aims to empirically demonstrate why fine-tuning is often excluded from complexity estimates, thereby providing justification for its exclusion from the order of complexity in design space estimation.

Table 1. Comparison of design space complexity estimates with respect to fine-tuning epochs.

Work	Complexity estimate	Treatment of fine-tuning epochs
[11]	$O(V \times T \times (PQ_a Q_w)^L)$	Not counted
This work	$O(V \times T \times F \times (PQ_a Q_w)^L)$	Tested, excluded

3. METHODOLOGY

In order to test the impact of fine-tuning on design space complexity, we design several experiments to see the impact of increasing the number of fine-tuning epochs on accuracy improvements after joint pruning-quantization. To eliminate the possibilities where accuracy is already saturated at a high level, which leaves little room for improvement, we focus these fine-tuning experiments on configurations that cause substantial accuracy degradation, i.e. layers that are sensitive to pruning or quantization. The overall procedure is as follows. First, we estimate each layer's sensitivity to pruning, to quantization, and to joint pruning-quantization in isolation, through a layer-wise sensitivity matrix. Then, we identify a few of the most sensitive layers based on their lowest average accuracy, and compress them under varying fine-tuning epoch counts to observe the accuracy trend. If fine-tuning with differing epochs has negligible effect, it implies that fine-tuning does not need to be considered as a separate dimension in the order of complexity for joint pruning-quantization.

3.1 Layer-wise Joint Pruning-Quantization Sensitivity

We perform three independent layer-wise pruning, quantization, and joint pruning-quantization experiments on the 19 convolutional layers of ResNet-20 [21]. In each case, a single target layer is pruned or quantized while all other layers remain at baseline settings (unpruned for the pruning test, or quantized to a standard 8-bit precision for the quantization test).

3.1.1 Pruning

Pruning was performed on each convolutional layer (Layers 1 through 19) with various pruning ratios. Here, the pruning ratio refers to the number of filters (output channels) to be removed. For example, a 0.3 pruning ratio in a 100-channel layer would remove 30 filters. We adopt an L_2-norm (Euclidean norm) magnitude criterion to rank filters that would be pruned in ascending order, which is formulated as

$$\| L_i \|_2 = \sqrt{\sum_j F_{i,j}^2} \tag{1}$$

where $F_{i,j}$ is the j-th weight of filter i, and j spans every weight element in that filter. Ranking through L_2-norm is intuitive since lower-magnitude filters should have less impact on the final output. This magnitude-based strategy is implemented via binary masks on the corresponding filters during inference or training.

3.1.2 Quantization

Similar to pruning, quantization is applied to each convolutional layer 1–19, using bitwidth from 7-bit to 1-bit for the target layer. While one layer is quantized to a low bitwidth, the rest of the network's layers remain at 8-bit quantization. We adopt quantization-aware training (QAT) [22] to simulate the effect of quantizing a layer and to facilitate fine-tuning during quantization. In QAT, we insert *fake quantization* operator during training that track the distribution of tensor values and learn optimal scaling factors. Each weight and activation tensors get a scale factor S and zero-point Z computed from the minimum and maximum values (α and β respectively) observed during training. The quantized representation $Q(r)$ could be formulated as:

$$Q(r) = \text{round}\left(\frac{r}{S} + Z\right) \tag{2}$$

$$S = \frac{\beta - \alpha}{2^n} \tag{3}$$

For activations, we use an asymmetric minimum-maximum observer since the activations are always

positive due to the Rectified Linear Unit (ReLU) [23] activation function. For weights we use a symmetric observer. In each quantization experiment, we include 1 fine-tuning epoch so that the observers can calibrate on the training data at least once. One epoch of training ensures the collected statistics reflect the dataset, which is crucial for QAT to adjust quantization parameters. To vary the effective bitwidth b of layer index l in simulation, we constrain the range of values for the quantized layer's weights and activations to $\left[-\frac{2^{b_l}}{2}, +\frac{2^{b_l}}{2}-1\right]$. This technique emulates lower-precision by restricting the dynamic range, since the byte-wise storage convention stores tensors in bytes (8-bit).

3.1.3 Joint Pruning-Quantization

For joint pruning-quantization sensitivity, we combine the above methods: we prune a target layer at various ratios and also quantize it. However, we fix quantization at 8-bit for this analysis due to the huge design space. We examine sensitivity when a layer is pruned by a certain amount, while other layers are left uncompressed (without pruning) but all layers are quantized to 8-bit. We chose to keep 8-bit for quantization here to limit the complexity; varying both pruning ratio and bitwidth per layer would drastically increase the combinations even for a single layer analysis. During joint pruning-quantization experiments, the same binary mask from the pruning step is applied during quantization-aware training so that the pruned filters remain inactive. Similar to quantization-only case, we fine-tune for 1 epoch in QAT to calibrate quantization observers.

3.2 Design Space Complexity Analysis with Fine-Tuning

After profiling the sensitivity of each layer, we select the most sensitive layers and repeat the joint pruning-quantization experiments with different fine-tuning epochs. The rationale is: if fine-tuning with higher epochs does not significantly improve accuracy, then fine-tuning does not need to be considered in the design space complexity estimates. For instance, if one epoch of retraining yields nearly all accuracy recovery, while the subsequent five epochs contribute an additional improvement of less than 5% relative to the gain obtained in the first epoch, then the extra benefit is considered negligible. In this case, treating fine-tuning as a simple binary switch (no fine-tune vs. with fine-tune) is sufficient. We test this by measuring the accuracy when fine-tuning the compressed model with varying epoch using the previously identified sensitive layers.

Furthermore, we also evaluated an aggressive setting in which every layer was fixed at 2-bit quantization to ensure that accuracy saturation was not masking potential gains. By examining both moderate and aggressive compression configurations by varying the global bitwidth, we aim to exclude the possibility that the absence of accuracy improvements is due to saturation effects.

4. EXPERIMENT DETAILS

We evaluated our methods on the CIFAR-10 dataset [24], which consists of 60k images across 10 classes, where we split the dataset to a 5:1 training and testing ratio. Standard preprocessing was applied which are random cropping, horizontal flipping, and normalization. The baseline model is a pre-trained ResNet-20 [21], a smaller variant of the ResNet family designed for efficiency. ResNet-20 was originally derived from ResNet-50 by reducing depth, making it suitable for CIFAR-10. The network consists of 19 convolutional layers and 1 fully-connected output layer. We applied pruning and quantization only to the convolutional layers, as they dominate the overall computation in deep neural networks [3, 13]. All accuracy results are reported on the test set as top-1 accuracy. The fine-tuning protocol used for all pruning and quantization experiments is summarized in Table 2. Experiments were run on a workstation with an Intel i7-13700KF CPU, 32 GB RAM, and an NVIDIA RTX 4070 GPU, using Windows 10, CUDA 12.1 and PyTorch 2.6.0 for implementation.

Table 2. Fine-tuning hyperparameter settings.

Item	Setting
Optimizer	Stochastic gradient descent
Learning rate	0.005
Momentum	0.9
Weight decay	4×10^{-5}
Batch size	256

5. RESULTS AND DISCUSSION

This section presents the layer-wise sensitivity results, followed by an analysis of why fine-tuning does not affect the design space complexity.

5.1 Layer-wise Joint Pruning-Quantization Sensitivity

This subsection presents the layer-wise sensitivity outcome and analysis for pruning, quantization, and joint pruning-quantization.

5.1.1 Pruning

As expected, pruning with a higher pruning ratio reduces accuracy even further, as shown in Figure 1. With the exception of Layer 1, sensitivity to pruning increases progressively for deeper layers (Layers 2–19), when using a 90% top-1 accuracy threshold as the acceptability criterion. Shallow layers can sustain higher pruning ratios before accuracy falls below 90%, whereas deeper layers hit this accuracy threshold at lower pruning ratios. Some suggests that this is because the feature maps in deeper layers have smaller dimensions due to max pooling [25], making them more sensitive to pruning. Another possible reason is that pruning sensitivity in deeper layers are proportional to the number of classes, as it requires more channels to preserve sufficient feature maps [26]. On the other hand, Layer 1 is sensitive because it propagates information to all subsequent layers. In many heuristic approaches, the first layer is typically left unpruned and unquantized [13].

Beyond the 90% accuracy threshold, evaluating how much information each layer contributes to the network becomes more complex. For example, pruning Layers 9 and 10 with a 0.3 ratio still keeps accuracy above 90% for both. However, under a much more aggressive 0.8 pruning ratio, Layer 9 becomes highly sensitive (accuracy drops to

41%) while Layer 10 remains comparatively robust (accuracy maintained at 83%).

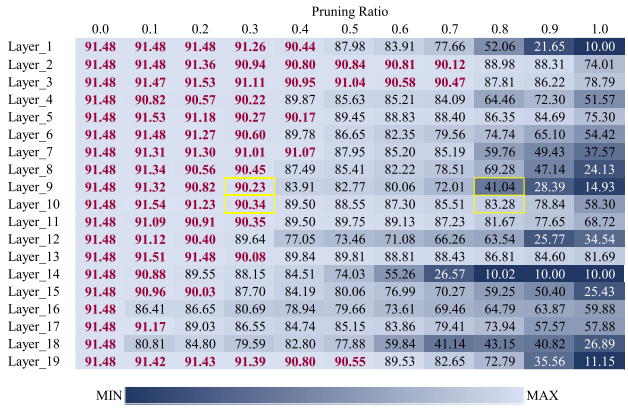

Figure 1. Layer-wise pruning sensitivity.

5.1.2 Quantization

Quantization's impact on the model accuracy shows a different pattern from pruning. As shown in Figure 2, reducing the bitwidth of individual layers causes nearly all layers to incur a similar drop in accuracy. For example, when each layer (except the first) is individually quantized to 3-bit, the average accuracy is approximately 80%. However, when quantized to 2-bit, the average accuracy drops to around 60%. This relatively uniform impact suggests that quantization error in any of the layers has a comparable effect on final accuracy.

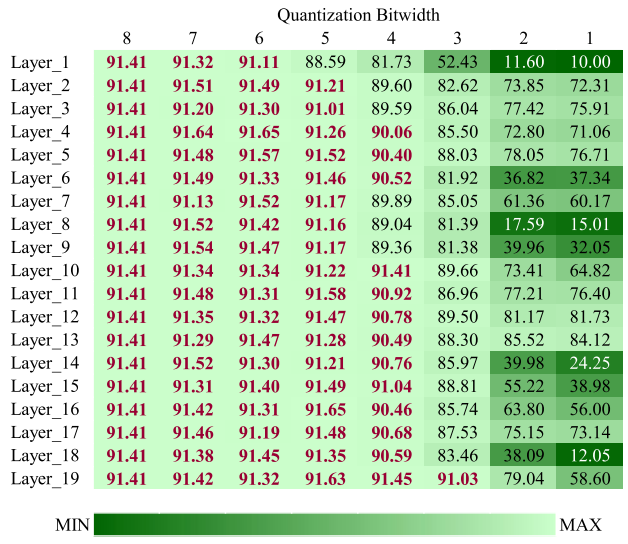

Figure 2. Layer-wise quantization sensitivity.

Although the pruning and quantization sensitivity matrices may suggest that ResNet-20 is less sensitive to quantization than pruning, it is important to clarify that our quantization experiments included a one-epoch of fine-tuning whereas pruning did not. Quantization-aware fine-tuning helped the network tolerate low-bit weights, whereas the pruning results were worst-case (no fine-tuning). Generally, CNN models are more sensitive to quantization than pruning, as reducing the bitwidth by one-bit halves the number of possible representable values in a layer, whereas pruning merely removes filters with the lowest magnitudes [14]. Thus, this also highlights the importance of even minimal fine-tuning.

5.1.3 Joint Pruning-Quantization

When pruning and quantization are applied together, we observed that with fine-tuning, the model can maintain high accuracy even under significant compression. In our joint pruning-quantization test (with each layer pruned by various amounts and all layers are quantized to 8-bit), most configurations achieved reasonably high accuracy after the 1-epoch fine-tune, as depicted in Figure 3. By contrast, disabling fine-tuning led to a severe accuracy drop which is around 10% for most layers across all pruning ratios.

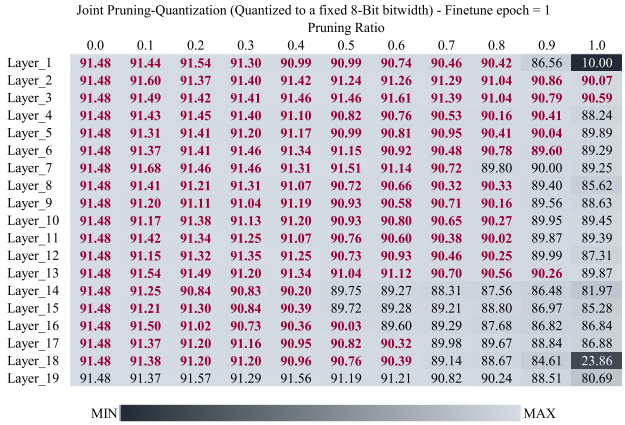

Figure 3. Layer-wise joint pruning-quantization sensitivity fixed at 8-bits.

Ideally, quantization bitwidth would also be varied per layer in these joint experiments to fully assess layer-wise sensitivity under mixed compression. However, the search space becomes too large if both pruning ratios and bitwidth are exhaustively simulated for every layer. Thus, our fixed 8-bit joint analysis represents a constrained view of the broader problem.

It is also important to note that our layer-wise studies (Sections 5.1.1–5.1.3) isolate each layer by assessing it independently. In practical scenario, compressing multiple layers will have compounding effects. The sensitivity of a given layer could change if the preceding layer has already been pruned or quantized, since less information is flowing into it. Similarly, a layer's sensitivity under pruning and quantization may differ from its sensitivity to either method alone, since the techniques can complement each other. For example, pruning reduces the number of weights to be quantized, potentially allowing for more aggressive quantization.

Accordingly, the results should be interpreted as a simplified approximation of the full behavior. Another limitation of this study is that the evaluation is restricted to

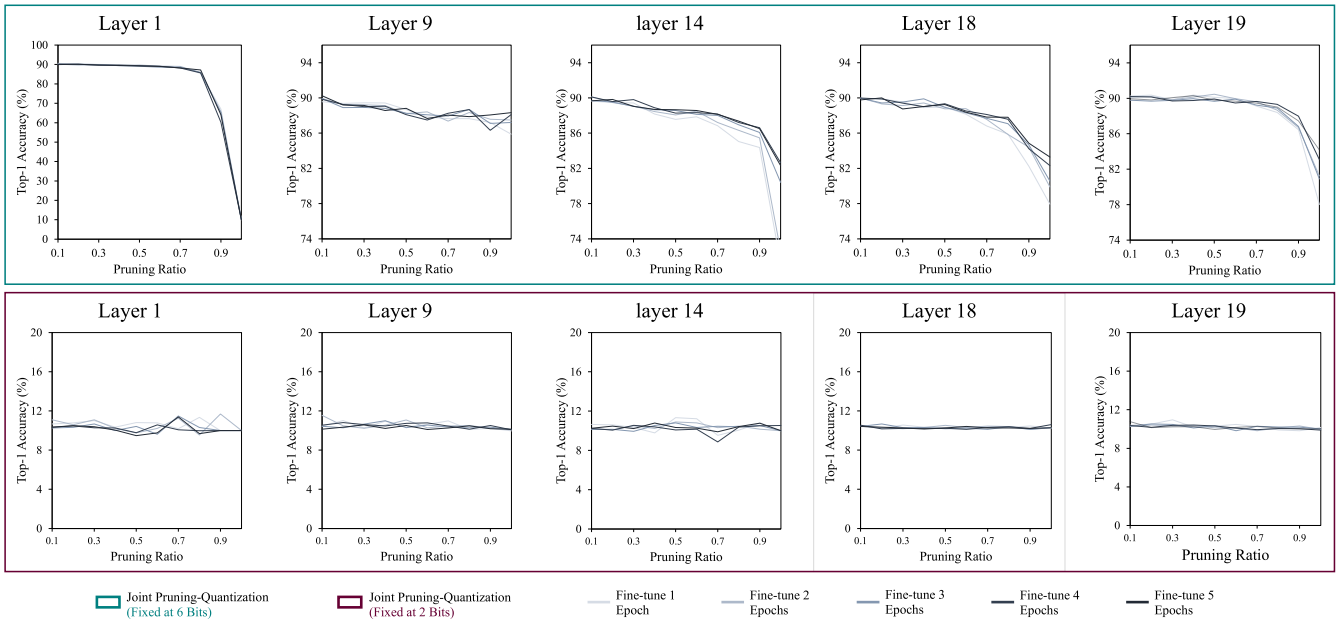

Figure 4. Accuracy with differing pruning ratio at 2-bits and 6-bits with varying fine-tuning epoch on sensitive layers.

a single model and dataset, namely ResNet-20 on CIFAR-10. The observed trends, including fine-tuning saturation, may vary for larger architectures or datasets with different data complexity. This motivates future validation beyond the current scope, as joint pruning-quantization has also been studied beyond image classification and evaluated across multiple datasets in other domains [27]. This design space complexity explains why many recent works formulate layer-wise joint pruning-quantization as an optimization problem rather than exhaustive evaluation. For instance, using optimization algorithm such as Alternating Direction Method of Multipliers (ADMM) [14], Bayesian [18, 19], or reinforcement learning [28].

5.2 Design Space Complexity Analysis with Fine-Tuning

The above sensitivity analyses underscore that fine-tuning plays a huge role in restoring accuracy to a compressed model. However, many works ignore fine-tuning when estimating the design space complexity for layer-wise joint pruning-quantization. Our experiments validate the reasoning behind this. In Figure 4, we plot the test accuracy for two representative joint compression settings as a function of pruning ratio, under different fine-tuning epochs. One setting uses a “moderate” global quantization (6-bit weights and activations) and the other an “aggressive” global quantization (2-bit weights and activations). In both cases, the accuracy curves show a similar trend. Once at least 1 epoch of retraining is applied, further fine-tuning (e.g. 2 to 5 epochs) yields only minor incremental gains. In our experiments, the accuracy curves for 1 and >1 fine-tuning epochs are almost overlapping. Quantitatively, in 99 out of 100 layer-wise joint pruning-quantization scenarios, extending fine-tuning from 1 to 5 epochs changes the Top-1 accuracy by less than 5%. We observed one exception at global 6-bit quantization with 100% pruning on layer 14, where the difference is approximately 11%. Even in this extreme boundary case (100% pruning), extending fine-tuning from 1 to 5 epochs only yields a limited additional accuracy change. Overall, these results indicate that one epoch of retraining already saturates the recoverable accuracy in layer-wise joint pruning-quantization scenarios. Given these findings, fine-tuning does not meaningfully increase the design space complexity for layer-wise compression. Fine-tuning can be treated as a binary decision (to fine-tune or not) rather than a continuous hyper-parameter to optimize.

5.3 Robustness Check with Learning Rate Variation

To assess robustness, we repeat fine-tuning on a representative compression configuration (global 6-bit quantization with 70% pruning on layer 1) using different learning rates. Table 3 reports the Top-1 accuracy after 1 and 5 fine-tuning epochs and the corresponding accuracy difference. Across the tested learning rates, the qualitative behavior remains consistent, where extending fine-tuning from 1 to 5 epochs yields only a small additional change in Top-1 accuracy. Specifically, the difference remains within a narrow range (approximately −0.8% to +1.2%) across learning rates. This suggests that even when the learning rate varies, fine-tuning epochs do not need to be treated as an additional design-space dimension in layer-wise joint pruning-quantization and can be modeled as a binary decision when estimating design space complexity.

Table 3. Learning rate sensitivity for the outlier case.

Learning rate	Accuracy (1 epoch)	Accuracy (5 epoch)	Difference in accuracy
0.001	87.09%	88.28%	1.2%
0.003	88.23%	88.83%	0.6%
0.005	88.32%	88.24%	-0.1%
0.010	88.83%	88.00%	-0.8%

6. CONCLUSION

We have presented an in-depth analysis of layer-wise sensitivity to pruning and quantization in ResNet-20, and examined the role of fine-tuning in the design space complexity estimates of layer-wise joint pruning-

quantization. First, we found that deeper layers become progressively more sensitive to pruning, with the exception of the first layer. Second, quantization leads to relatively uniform accuracy degradation across intermediate layers, i.e., quantizing any single middle layer to a low bitwidth tends to have a similar impact on accuracy. Third, joint pruning-quantization with a fixed 8-bit quantization yields high accuracy, largely due to the effect of fine-tuning. While further compression is possible, the design space becomes prohibitively complex.

Finally, we demonstrated that the accuracy does not change significantly with varying numbers of fine-tuning epochs, as accuracy saturates after 1 epoch. These findings validate earlier estimates that justify excluding fine-tuning when approximating the order of complexity for layer-wise joint pruning-quantization. In essence, the influence of fine-tuning on design space complexity is a binary decision. The results depend on whether fine-tuning is applied or not, rather than dependent on the number of fine-tuning epochs. These findings suggest that future research should prioritize strategies for selectively compressing layers, determining which layers to compress more or less, rather than focusing on fine-tuning duration. Efficiently navigating the design space of layer-wise joint pruning-quantization will be critical as DL models' complexity continue to scale.

ACKNOWLEDGMENT

The authors would like to acknowledge Universiti Teknologi Malaysia (UTM) for the sponsorship provided under the Skim Latihan Akademik Muda (SLAM).

REFERENCES

[1] Y. LeCun, Y. Bengio and G. Hinton, "Deep learning," *Nature,* vol. 521, p. 436–444, 2015.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems (NIPS),* vol. 30, 2017.

[3] V. Sze, Y.-H. Chen, T.-J. Yang and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE,* vol. 105, p. 2295–2329, 2017.

[4] D. Xu, T. Li, Y. Li, X. Su, S. Tarkoma, T. Jiang, J. Crowcroft and P. Hui, "Edge intelligence: Empowering intelligence to the edge of network," *Proceedings of the IEEE,* vol. 109, p. 1778–1837, 2021.

[5] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers and others, "In-datacenter performance analysis of a tensor processing unit," in *Proceedings of the 44th Annual International Symposium on Computer Architecture (ISCA)*, 2017.

[6] Y. Wang, Y. Lu and T. Blankevoort, "Differentiable joint pruning and quantization for hardware efficiency," in *European Conference on Computer Vision (ECCV)*, 2020.

[7] S. Han, H. Mao and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," in *International Conference on Learning Representations (ICLR)*, 2016.

[8] S. Han, J. Pool, J. Tran and W. Dally, "Learning both weights and connections for efficient neural network," *Advances in Neural Information Processing Systems,* vol. 28, 2015.

[9] S. Wu, G. Li, F. Chen and L. Shi, "Training and Inference with Integers in Deep Neural Networks," in *International Conference on Learning Representations (ICLR)*, 2018.

[10] T. Wang, K. Wang, H. Cai, J. Lin, Z. Liu, H. Wang, Y. Lin and S. Han, "APQ: Joint search for network architecture, pruning and quantization policy," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[11] K. Wang, Z. Liu, Y. Lin, J. Lin and S. Han, "HAQ: Hardware-aware automated quantization with mixed precision," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[12] S. Yu, A. Mazaheri and A. Jannesari, "Topology-aware network pruning using multi-stage graph embedding and reinforcement learning," in *International Conference on Machine Learning (ICML)*, 2022.

[13] L. Deng, G. Li, S. Han, L. Shi and Y. Xie, "Model compression and hardware acceleration for neural networks: A comprehensive survey," *Proceedings of the IEEE,* vol. 108, p. 485–532, 2020.

[14] A. Ren, T. Zhang, S. Ye, J. Li, W. Xu, X. Qian, X. Lin and Y. Wang, "Admm-nn: An algorithm-hardware co-design framework of dnns using alternating direction methods of multipliers," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2019.

[15] X. Zeng, T. Zhi, X. Zhou, Z. Du, Q. Guo, S. Liu, B. Wang, Y. Wen, C. Wang, X. Zhou and others, "Addressing irregularity in sparse neural networks through a cooperative software/hardware approach," *IEEE Transactions on Computers,* vol. 69, p. 968–985, 2020.

[16] T.-Y. Hsiao, Y.-C. Chang, H.-H. Chou and C.-T. Chiu, "Filter-based deep-compression with global average pooling for convolutional networks," *Journal of Systems Architecture,* vol. 95, p. 9–18, 2019.

[17] F. Tung and G. Mori, "Deep neural network compression by in-parallel pruning-quantization," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 42, p. 568–579, 2018.

[18] F. Tung and G. Mori, "Clip-q: Deep network compression learning by in-parallel pruning-quantization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[19] M. Van Baalen, C. Louizos, M. Nagel, R. A. Amjad, Y. Wang, T. Blankevoort and M. Welling, "Bayesian bits: Unifying quantization and pruning," *Advances in Neural Information Processing Systems (NIPS),* vol. 33, 2020.

[20] Y. Lin, L. Niu, Y. Xiao and R. Zhou, "Diluted binary neural network," *Pattern Recognition,* vol. 140, p. 109556, 2023.

[21] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[22] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney and K. Keutzer, "A survey of quantization methods for efficient neural network inference," in *Low-power Computer Vision*, Chapman and Hall/CRC, 2022, p. 291–326.

[23] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML)*, 2010.

[24] A. Krizhevsky, G. Hinton and others, "Learning multiple layers of features from tiny images," 2009.

[25] M. Zhang, X. Yu, J. Rong and L. Ou, "Graph pruning for model compression," *Applied Intelligence,* vol. 52, p. 11244–11256, 2022.

[26] Z. Liu, H. Mu, X. Zhang, Z. Guo, X. Yang, K.-T. Cheng and J. Sun, "Metapruning: Meta learning for automatic neural network channel pruning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[27] M. N. A. Nor-Azman, U. U. Sheikh, M. S. Mohammed, J. Sirkunan and M. N. Marsono, "Bufferless network traffic classification using reinforcement learning-based joint pruning-quantization," *Evolutionary Intelligence,* vol. 19, no. 1, p. 3, 2026.

[28] M. N. A. Nor-Azman, U. U. Sheikh, M. S. Mohammed, J. Sirkunan and M. N. Marsono, "Correlation-Aware Joint Pruning-Quantization using Graph Neural Networks," in *2024 IEEE International Conference on Image Processing (ICIP)*, 2024.