**Efficient Compression of ResNet-18 for CIFAR-10 Using Pruning and Quantization**

Din Alon 207131525

Yakir Witkin 328606983

Faculty of Electrical and Computer Engineering, Technion - Israel Institute of Technology  
**Spring 2025**

Table of Contents

[Abstract 2](#_Toc203492925)

[1. Introduction 2](#_Toc203492926)

[2. Optimization Techniques 3](#_Toc203492927)

[3. Experiments on ResNet-20 6](#_Toc203492928)

[4. Comparison: ResNet 18 vs ResNet 20 7](#_Toc203492936)

[5. Results & Conclusions 7](#_Toc203492941)

[6. Future Work 8](#_Toc203492946)

[7. ethics 9](#_Toc203492948)

[8. References 10](#_Toc203492949)

Abstract

This project explores compressing the ResNet-18 architecture, originally designed for the large-scale ImageNet dataset, to optimize its efficiency for CIFAR-10. We applied automatic mixed precision (AMP), structured pruning using the torch-pruning library, and post-training quantization (PTQ) using TensorRT. The baseline ResNet-18 model achieves 92.02 percent test accuracy with a size of 42.66 MB. After pruning 60 percent of the model's parameters and quantizing it to INT8, we reduced the model size to 2.96 MB while maintaining 89.03 percent accuracy. The compressed ResNet-18 was compared to ResNet-20, a smaller architecture designed for CIFAR-10, to evaluate trade-offs in complexity, performance, and suitability for deployment in resource-constrained environments.

1. Introduction

ResNet-18 is a widely used convolutional neural network architecture known for its strong feature extraction capabilities. Originally designed for the large-scale ImageNet dataset, it contains a relatively large number of parameters and has substantial memory and computational demands. For smaller datasets such as CIFAR-10, this level of complexity is often unnecessary and inefficient, particularly in resource-constrained or edge deployment scenarios.

In this project, we explore methods to adapt and compress ResNet-18 to make it suitable for efficient inference on CIFAR-10. We trained ResNet-18 from scratch on the CIFAR-10 dataset, achieving 92.02 percent test accuracy with a model size of 42.66 MB. We then investigated three optimization techniques: automatic mixed precision (AMP), structured pruning using the torch-pruning library, and post-training quantization using TensorRT. We also experimented with combining pruning and quantization into a single pipeline.

Our compression efforts were primarily focused on convolutional layers, since the majority of the model's parameters and computation are concentrated in these layers, as shown in the original ResNet paper by He et al. [1]. We also took inspiration from the work of Banner et al. [2], which demonstrated that post-training quantization of convolutional networks is feasible even without access to the full dataset or fine-tuning, making it well suited for rapid deployment in practical settings.

To evaluate the effectiveness of these methods, we benchmarked the compressed ResNet-18 models against ResNet-20, a lightweight architecture explicitly designed for CIFAR-10. ResNet-20 has significantly fewer parameters, making it a strong baseline for comparison. By analyzing the trade-offs between model size, inference speed, and classification accuracy, we aim to determine how effectively a general-purpose architecture like ResNet-18 can be adapted for small-scale image classification tasks.

2. Optimization Techniques

This section presents the methods used to compress and accelerate the ResNet-18 model. Each technique was chosen to address a different dimension of efficiency: reducing memory, speeding up inference, or minimizing parameter count. The techniques evaluated include automatic mixed precision (AMP), structured pruning, post-training quantization (PTQ), and a combined pruning-plus-quantization pipeline.

We detail the implementation and results of each method, analyzing their impact on model size, accuracy, and practical deployability.

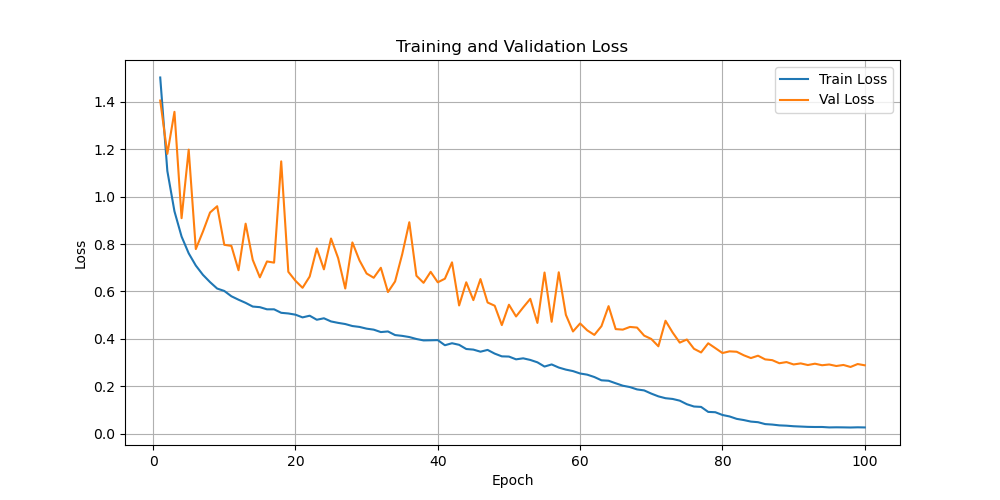
2.1 Automatic Mixed Precision (AMP*)*

Automatic mixed precision was used during training to reduce memory consumption and accelerate computation by casting selected operations to 16-bit floating point, while preserving accuracy-critical operations in 32-bit. The model was trained once with full 32-bit precision and once using AMP, and the results were compared.

As shown in the training and validation loss curves (Figure 1a and Figure 1b), both models followed a nearly identical learning trajectory over 100 epochs. The 32-bit model achieved 92.02 percent test accuracy with an average epoch time of 3.93 seconds. The AMP-trained model reached 91.95 percent accuracy and reduced the epoch time to 3.66 seconds.

Although AMP introduced a slight speed improvement, the difference was not substantial. This is likely due to the modest size of ResNet-18, which does not fully utilize the GPU’s memory bandwidth or compute throughput. As a result, the performance gains from mixed precision were limited in this setting.

We conclude that AMP can be safely applied to small-scale models like ResNet-18 without impacting accuracy, though its benefits in terms of speed and efficiency are more pronounced in larger architectures.

A graph of a graph showing the loss of a graph

AI-generated content may be incorrect.

**Figure 1b**: Loss curves for ResNet-18 trained with AMP.

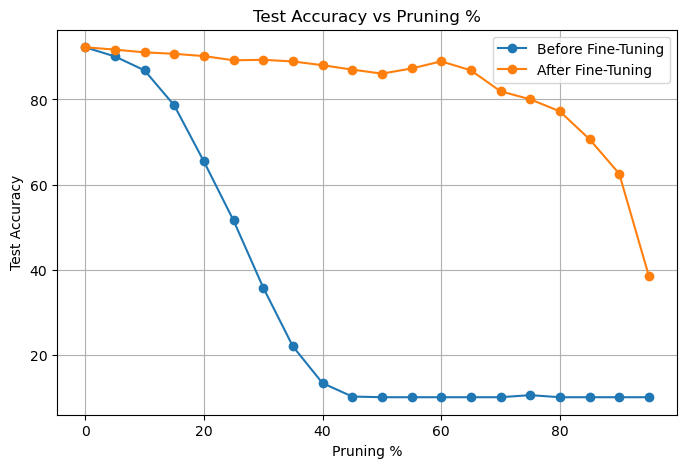
**Figure 1a**: Loss curves for ResNet-18 trained in 32-bit precision.

2.2 Structured Pruning

We applied layer-wise structured pruning to progressively remove filters from the convolutional layers. Filters were selected for removal based on the magnitude of their weights, using the L1 norm as a ranking criterion. This assumes that filters with smaller absolute weights contribute less to the model’s predictions and can be safely discarded.

After each pruning step, the model was fine tuned to recover performance loss. As shown in the pruning curve (Figure 2a), we were able to maintain high test accuracy, above 85 percent, even with pruning levels up to 65 percent when fine tuning was applied. Accuracy drops significantly without retraining, but fine tuning successfully restores performance. According to the summary table (Figure 2b), the best tradeoff between accuracy and model size occurs at 60 percent pruning, where the model is reduced from 42.66 MB to 6.79 MB while still achieving 88.96 percent test accuracy.

We also observed that both the learning rate and the number of fine-tuning epochs needed to increase with higher pruning levels. This is expected because the model was hurt more by aggressive pruning and required stronger updates and longer training to recover its performance.



**Figure 2a**: Test accuracy before and after fine tuning for different pruning levels

תמונה שמכילה טקסט, מספר, צילום מסך, קו

תוכן בינה מלאכותית גנרטיבית עשוי להיות שגוי.

**Figure 2b**: Summary of pruning results including model size, accuracy, and training settings

2.3 Post-Training Quantization (PTQ)

We used TensorRT to apply post training quantization in both FP16 and INT8 formats. Calibration was class balanced, and the quantization was performed entirely on the GPU and without retraining. The model was evaluated on both CIFAR 10 and the more challenging CIFAR 100 dataset. As shown in Figures 3a and 3b, quantization reduced model size and inference time, while the number of parameters remained unchanged.

Accuracy remained nearly identical. On CIFAR 10, the INT8 model scored 92.20 percent compared to 92.26 percent for FP32. On CIFAR 100, the INT8 model achieved 79.07 percent versus 79.29 percent for FP32. These results are consistent with Intel’s findings, where post training static quantization of ResNet 18 resulted in only a 0.24 percent drop in accuracy compared to full precision on the ImageNet dataset [3].

Quantization proved highly effective at reducing model size and latency while maintaining accuracy, all without requiring fine tuning.

תמונה שמכילה טקסט, צילום מסך, קבלה, מספר

תוכן בינה מלאכותית גנרטיבית עשוי להיות שגוי. A table with numbers and text

AI-generated content may be incorrect.

**Figure 3b**: Summary of ResNet 18 quantization results on CIFAR 100

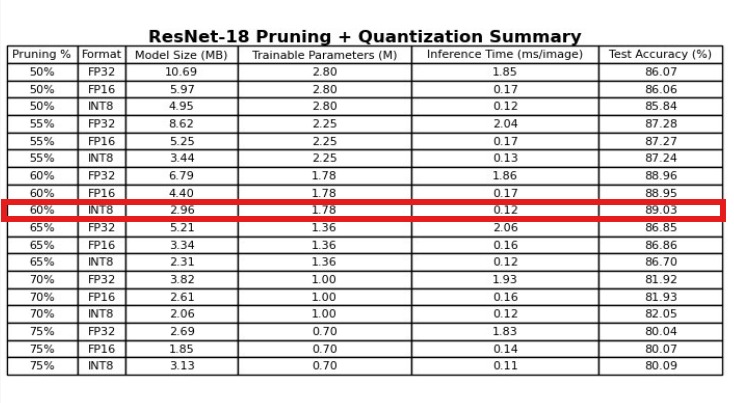
**Figure 3a**: Summary of ResNet 18 quantization results on CIFAR 10

2.4 Combined Pruning + Quantization

In this experiment, we combined structured pruning with post training quantization. After pruning and retraining to recover accuracy, we quantized the model using TensorRT in both FP16 and INT8 formats. We focused on pruning levels between 50 and 75 percent.

As shown in Figure 4, this combination produced extremely compact models with relatively high accuracy. The best tradeoff between size and accuracy was achieved with 60 percent pruning followed by INT8 quantization. This model scored 89.03 percent on the test set compared to 92.02 percent for the original full precision model. Its size was reduced from 42.66 MB to just 2.96 MB.

It is important to note that the full precision models were evaluated using PyTorch, while the FP16 and INT8 models were run using TensorRT. This explains the noticeable differences in inference time across formats. TensorRT is optimized for deployment and provides lower latency, especially in low precision settings.



**Figure 4**: Summary of ResNet 18 test accuracy, size, and latency after pruning and quantization

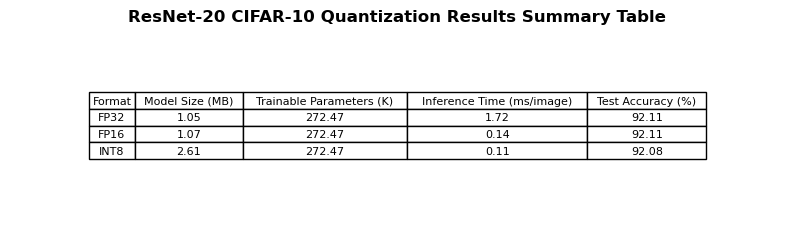
3. Experiments on ResNet-20

To evaluate how well our compression techniques generalize to smaller architectures, we applied the same pruning and quantization methods to ResNet 20. This model is significantly smaller than ResNet 18, with a size of just 1.05 MB and 20 convolutional layers compared to ResNet 18’s 42.66 MB and 18 layers.

As shown in the quantization summary (Figure 5a), post training quantization did not reduce model size meaningfully. The INT8 version was even larger than the FP32 baseline. This is likely due to the added overhead of scaling factors and zero point information, which becomes non negligible when the base model is already very small.

Pruning results are shown in Figure 5b. After aggressive pruning, test accuracy dropped sharply and became difficult to recover, even with extended fine tuning. This suggests that ResNet 20 has little redundancy to remove, and its capacity is already close to the minimum required for the CIFAR 10 task.

These results demonstrate that compression techniques are far more effective on larger models. For very compact architectures, the performance loss may outweigh the benefits in size and speed.



**Figure 5a**: Quantization results on ResNet 20 for CIFAR 10

A table with numbers and lines

AI-generated content may be incorrect.

**Figure 5b**: Pruning results on ResNet 20

4. Comparison: ResNet 18 vs ResNet 20

To better understand the difference in capacity between ResNet 18 and ResNet 20, we referenced publicly reported results, as we did not have time to conduct experiments on the Tiny ImageNet dataset ourselves.

Tiny ImageNet is a down-sampled version of the ImageNet dataset, consisting of 200 classes with 64×64 resolution images. It is significantly more challenging than CIFAR-10 or CIFAR-100 due to the greater number of classes and more complex visual patterns.

According to Li et al. [4], ResNet 18 achieves 68.89 percent top-1 accuracy on Tiny ImageNet. In comparison, Yu [5] reports that ResNet 20 achieves only 55.40 percent on the same dataset. This substantial accuracy gap highlights the difference in representational capacity between the two models.

While ResNet 20 is highly compact and suitable for simple classification tasks, it lacks the flexibility to generalize well on more complex datasets

5. Results & Conclusions

This study evaluated multiple compression techniques to reduce the size and inference time of convolutional neural networks, focusing on ResNet 18 and ResNet 20 trained on CIFAR 10 and CIFAR 100.

Method Summaries

תמונה שמכילה טקסט, מספר, צילום מסך, גופן

תוכן בינה מלאכותית גנרטיבית עשוי להיות שגוי.

For ResNet 18, both post training quantization and structured pruning were effective individually, but the best results were achieved by combining them. The 60 percent pruned and INT8 quantized model reduced memory from 42.66 MB to 2.96 MB while maintaining 89.03 percent test accuracy. This configuration represents the best balance between accuracy and deployment efficiency.

These techniques proved robust across datasets. However, our experiments also showed that model size plays a critical role. Compression was far less effective on ResNet 20, a much smaller network. Quantization overhead became significant, and pruning caused irreversible accuracy drops. This highlights the limitation of applying these techniques to already-compact models.

While ResNet 20 is highly efficient and suitable for simple classification tasks, it lacks the flexibility to generalize well on more complex datasets. In contrast, larger models like ResNet 18 are better suited for tasks that involve greater class diversity or require stronger representational capacity. Although they demand more resources, they can be effectively compressed and optimized after training. Ultimately, model selection should be guided by the complexity of the target application - compact models are ideal for lightweight scenarios, whereas larger models are necessary for more challenging tasks and can still be made deployable through compression.

6. Future Work

Future work could explore several promising directions. First, using quantization-aware training (QAT) may help bridge the remaining accuracy gap for highly compressed models. Second, adaptive pruning strategies that adjust pruning ratios per layer could preserve critical network structure more effectively. Third, investigating newer transformer-based or hybrid CNN architectures may reveal models that are inherently more compressible. Finally, evaluating these techniques under real-time deployment constraints, such as on embedded hardware or mobile platforms, would provide practical insight into their usability in edge applications.

7. Ethics Statement

To ensure that any project benefits society, it is important to consider its impact on various groups of people and specific individuals. The following section discusses the ethical considerations regarding this project, including identifying the stakeholders directly affected by this project, how they should be informed about the existence of the project and who bears the responsibility to inform them. To avoid potential biases, we presented a series of questions to ChatGPT and gathered its answers. We will present and review the answers manually.

7.1 Presenting the answers given by the LLM

Question: list 3 types of stakeholders that will be affected by this project

Answer:

1. **AI/ML Engineers and Researchers** – who design and maintain deep learning models.
2. **Product and Deployment Teams** – who deploy models into production, especially in resource-constrained environments like mobile or edge devices.
3. **End Users** – who indirectly benefit from faster, lighter, and more efficient applications that use these compressed models.

Question: What will an explanation that is given to each stakeholder look like?

Answer:

**To AI/ML Engineers and Researchers:**

Our project shows how combining structured pruning and post-training quantization can drastically reduce the size and latency of ResNet-18 for CIFAR-10 with minimal accuracy loss. This makes it feasible to use general-purpose models in constrained settings without building specialized small networks from scratch. The report details trade-offs and practical results that could guide similar efforts on other datasets or architectures.

**To Product and Deployment Teams:**

By compressing ResNet-18, we reduced the model size from 42.66 MB to just 2.96 MB while maintaining competitive accuracy. This enables deployment of the model on resource-constrained devices without major changes to existing infrastructure. The project includes performance measurements using TensorRT, offering realistic estimates for latency and storage that help in planning deployment and scaling.

**To End Users:**

Thanks to model compression, apps that use image classification (like photo tagging, object detection, or AR) can now run faster and use less battery and storage, leading to a smoother and more responsive user experience on your phone or tablet.

Question: Who is responsible for giving the explanation to the stakeholders?

Answer:

The explanations should be given by the project team (Yakir Witkin and Din Alon) or the supervising faculty member. For AI/ML engineers, the team should present technical findings in written reports or internal presentations. For product teams, the explanation should come via briefings or documentation that translates technical results into business impact and deployment benefits. For end users, communication usually flows indirectly: product teams update user-facing release notes, app descriptions, or marketing materials to explain the benefits of faster and smaller models.

7.2 Reflecting on the LLM Response

The LLM identified relevant stakeholders and gave appropriate explanations to each of them. In addition, the LLM was able to properly assess the responsibilities of those involved in this project. However, we believe that end users don’t qualify as stakeholders in the same way researchers, engineers and product developers do. A better example of a stakeholder would be medical research clinics that utilize neural networks. By reducing the number of engineers working on building neural networks and allocating them to optimizing and quantizing the network we can reduce the number of people that require access to the database which in this case would contain medical information of patients. This could contribute to reducing medical information leaks and boosting patients’ trust in the medical research world opening the gateway for more research projects that will benefit society. Another example of a stakeholder would be environmentalists and environment policy makers. By reducing the size and inference time of the model, we can reduce the amount of energy required to run the model causing less pollution and keeping our air cleaner allowing these models to be considered safe for deployment and daily use.

8. References

[1] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), 2016. https://arxiv.org/pdf/1512.03385

[2] Banner, Ron, Yury Nahshan, and Daniel Soudry. "Post training 4-bit quantization of convolutional networks for rapid-deployment." arXiv preprint arXiv:1810.05723 (2018).

[3] Intel Neural Compressor – Validated Model List.  
<https://intel.github.io/neural-compressor/latest/docs/source/validated_model_list.html>

[4] Li, Siyuan, Liu, Zicheng, Wang, Zedong, Wu, Di, Liu, Zihan, and Li, Stan Z.

"Boosting Discriminative Visual Representation Learning with Scenario-Agnostic Mixup."

AI Lab, Westlake University. https://arxiv.org/pdf/2111.15454v3

[5] Yu, Hujia.

"Deep Convolutional Neural Networks for Tiny ImageNet Classification."

Stanford University, CS231n Course Project Report, 2017. https://cs231n.stanford.edu/reports/2017/pdfs/931.pdf