Implementation of Model Compression Techniques on Deep Neural Networks

Arnav Aghav[#], Avani Mundra[#], Anshika Pandey[#]
{aaghav,amudra,apande80}@asu.edu
[#]*School of Computing & Augmented Intelligence (SCAI), Arizona State University*

***Abstract* - This project explores the application of compression techniques to address the challenges posed by the increasing size and computational complexity of deep learning networks. Focusing on VGG-16 architectures, the study investigates various compression methods like quantization, knowledge distillation and pruning, and their impact on model performance when applied to the MNIST and CIFAR-10 datasets. Through comparative analysis, the project aims to assess the effectiveness of compression techniques in reducing model size while preserving performance, thereby contributing to the advancement of efficient and sustainable deep learning solutions.**

I. Introduction

In the dynamic realm of deep learning, the evolution of potent models has posed a significant challenge: the escalating size and computational intricacy of modern deep neural networks (DNNs). As these models expand, their deployment and utilization become progressively impractical, notably in constrained environments such as mobile and edge devices.

To tackle this challenge, compression techniques have emerged as indispensable tools in the deep learning arsenal. These techniques endeavor to diminish the size of DNNs without compromising their efficacy, thus facilitating more efficient deployment and inference. Key motivations driving the adoption of compression techniques include resource efficiency, deployment feasibility in constrained environments, expedited inference times, and minimized environmental impact.

This project proposal presents a concise overview of various compression techniques applied to VGG-16 architecture, alongside a comparative analysis of their implementation on MNIST and CIFAR-10 datasets. By examining the advantages and trade-offs inherent in these techniques, our goal is to underscore their pivotal role in advancing deep learning and fostering the creation of more accessible, deployable, and sustainable deep learning solutions.

II. Related Works

Quantization methods aim to reduce model size and computational demand by lowering the precision of the weights and activations in neural networks. Notably, Han et al. [3] introduced a comprehensive approach involving pruning, trained quantization, and Huffman coding, which collectively achieve significant reductions in storage requirements. Incremental approaches to network quantization were further explored by Zhou et al. [4], which aim to achieve near-lossless compression. More recently, adaptive binary-ternary quantization strategies have been demonstrated by Razani et al. [2], which offer dynamic precision adjustments to balance performance and compression.

Pruning is targeted at eliminating non-critical weights from a DNN to reduce its complexity and enhance computational efficiency. Li et al. [9] presented methods for pruning filters in convolutional networks, showing substantial reductions in resource demands without a severe impact on accuracy. Further exploration by Blakeney et al. [10] and Li et al. [11] has examined the implications of pruning on network layers, indicating that strategic removal of weights can maintain or even improve performance in certain contexts.

Knowledge distillation involves training a smaller, more efficient model (student) to emulate the performance of a larger model (teacher), which helps in deploying powerful DNNs on resource-limited devices. Gao et al. [17] proposed a straightforward approach for distilling knowledge into smaller networks, while Li et al. [18] focused on efficient network compression through few-sample distillation techniques, providing a pathway for reducing model size without substantially sacrificing accuracy.

III. Architecture & Dataset

The evaluation plan for assessing the effectiveness of the compression techniques on ResNet-18 and VGG-16 architectures will involve several key metrics and benchmarks. Firstly, the reduction in model size will be quantified, providing a clear measure of the efficiency gains from each compression technique.

A. *MNIST*

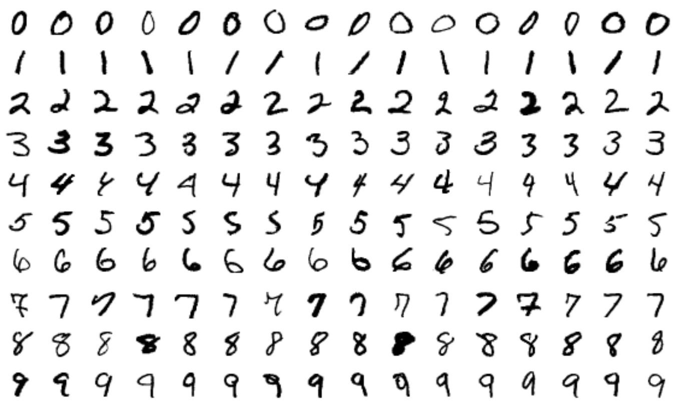

Fig. 1. MNIST Dataset

The MNIST dataset is a foundational resource in the machine learning community, containing 70,000 grayscale images of handwritten digits from 0 to 9. Each image is formatted as a 28x28 pixel square, making MNIST a highly accessible dataset for training and testing image recognition algorithms. A figure included with the dataset provides a visual representation of the various digit styles in the collection, showcasing the variability in handwriting that algorithms must learn to interpret. The simplicity of MNIST makes it ideal for beginners yet still presents a useful challenge for more advanced studies, promoting ongoing improvements in techniques for effective digit classification.

B. *CIFAR-10*

The CIFAR-10 dataset, a staple in machine learning, comprises 60,000 32x32 color images segmented into ten distinct classes, with each class containing 6,000 images. This dataset is essential for testing and demonstrating the capabilities of various image recognition models. Figure 2. of the dataset visually exemplifies the diversity and complexity of the images, ranging from animals to vehicles, providing a clear benchmark for algorithm performance.

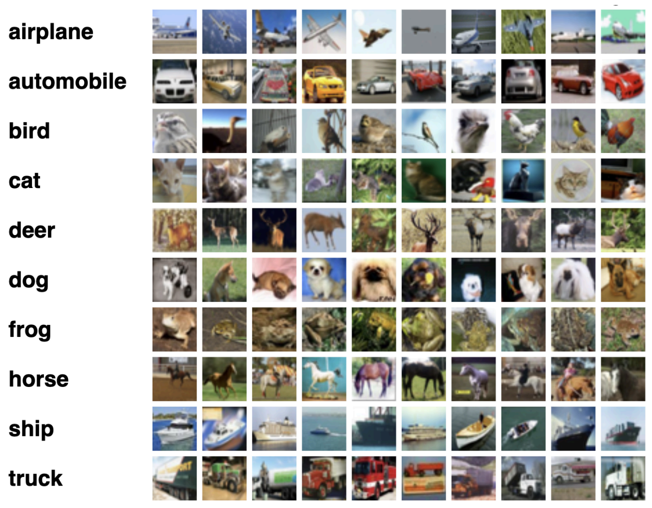

Fig. 2. CIFAR-10 Dataset

The compact size of these images ensures that CIFAR-10 is not only manageable but also challenging, given the limited resolution. As such, it continues to drive forward innovations and optimizations in model development tailored to handle its intricacies.

C. *VGG-16*

VGG'16, developed by K. Simonyan and A. Zisserman of the University of Oxford, significantly improved upon AlexNet's design by employing several sequential 3x3 sized filters instead of larger ones. This approach is detailed in their study "Very Deep Convolutional Networks for Large-Scale Image Recognition."

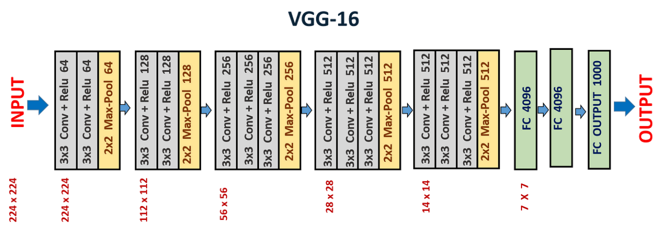

Fig. 3. VGG-16 Architecture

The architecture of VGG'16, depicted in Figure-1, efficiently processes images through its convolutional layers, crucial for feature detection, and its pooling layers, which handle spatial dimension reduction. Achieving an impressive 92.7% accuracy rate on the ImageNet challenge highlights its efficacy. However, the model's large size is a notable limitation, which has spurred ongoing modifications and improvements.

V. Model Compression Methods

A. Quantization

Quantization is a method aimed at diminishing the precision of numerical values utilized to represent the weights and activations within a deep neural network. By converting these values from their original floating-point format to lower-bit precision formats, such as 8-bit integers, quantization significantly reduces the model's size and accelerates the inference process. We explored two techniques - post-training quantization and quantization-aware training. Due to a trade-off with accuracy in post-training quantization, we adopted quantization-aware training in our project. During the training process, quantization-aware training simulates the effects of quantization by applying quantization functions or schemes to the network's parameters, mimicking how the network will behave once quantized during inference. This allows the model to adapt to the reduced precision and mitigate any potential loss of accuracy that may occur due to quantization. This process can also be considered as 'fine tuning' a quantized model by emulating inference time precision.

B. Pruning

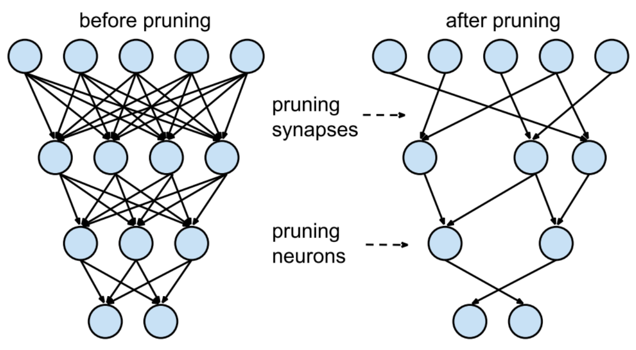

Fig. 4. Pruning illustration

Another technique mentioned in this paper is Pruning. Research carried out on applying simple pruning techniques has proven to be extremely beneficial for reducing inference costs of VGG-16 model by approximately 34% and ResNet-110 by approximately 38% on the CIFAR-10 dataset while maintaining substantial accuracy[6]. A comparison of pruned and original networks using the Singular Vector Canonical Correlation Analysis (SVCCA) has been significant in establishing the importance of Pruning on Neural Networks[7]. Compression techniques are important for integrating deep neural networks with edge computing devices, and algorithms such as clustering have also been used to extract filters and apply pruning to easy deployment of deep learning models in edge devices[8].

C. Knowledge Distillation

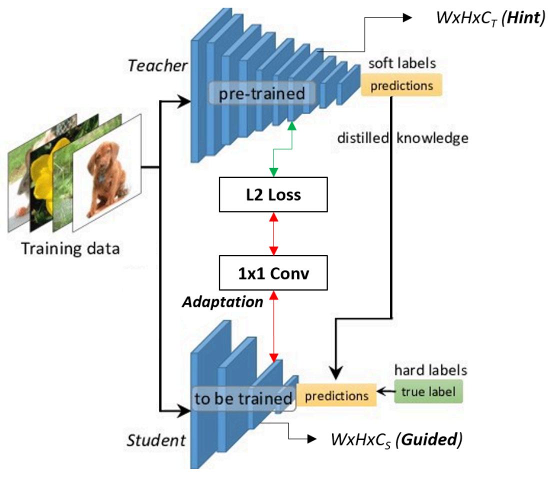

Fig. 5. Teacher-Student Knowledge Distillation illustration

Knowledge Distillation is a technique aimed at transferring knowledge from a larger, more complex model (teacher) to a smaller, more efficient model (student) without significant loss of performance. This process involves training the student model to mimic the output probabilities of the teacher model, effectively capturing the insights of a large network while operating with significantly fewer parameters and computational overhead. In our project, we utilized two forms of knowledge distillation: traditional knowledge distillation and attention transfer.

Traditional knowledge distillation works by softening the output logits of the teacher model using a temperature parameter, which helps in transferring more detailed information from the teacher to the student. This approach not only helps the student model achieve higher accuracy than training from scratch but also retains performance characteristics akin to the larger model.

Attention transfer, as another form of knowledge distillation, focuses on matching the attention maps of the teacher and student models. Attention maps indicate the regions within an input image that are most relevant for making a prediction. By aligning these maps between the teacher and student, the student model learns to focus on similar features as the teacher, enhancing its ability to generalize from limited data.

Both methods were applied in our study to evaluate their effectiveness in compressing the VGG-16 architecture for deployment on resource-constrained devices, such as mobile and edge computing platforms. The use of knowledge distillation proved particularly useful in maintaining close to baseline accuracies while significantly reducing the computational requirements and model size. This balance is critical for applications where both performance and efficiency are paramount.

V. Experimental Setup

All the experiments and implementation of this project, involving the implementation of the three compression techniques on MNIST and CIFAR10 datasets using VGG16 architecture were carried out on ASU's Sol Supercomputer with the following configurations:

- Type of GPU: a100
- Number of GPUs: 1
- Memory Allocated by GPU: 40GiB
- Number of Cores Allocated: 24

The utilization of Sol's powerful resources enabled us to carry out all the experiments successfully, and minimum memory challenges were encountered throughout the implementation

VI. Results and Observations

The efficiency of the compressed models as compared t the baseline model was tested using the following metrics:

Accuracy: The validation accuracy of the baseline model and compressed model have been compared

Model Size: The size of the compressed model was compared with the baseline model to check the efficiency of compression techniques.

Inference Time: The inference time of the baseline model is compared with the models compressed using various compression techniques. This is to check that the models not only run effectively but also optimally when compression techniques are applied to them.

A. Implementation of Compression Techniques on MNIST Dataset

Compression Technique	**Evaluation Metrics**		
	Model Size	***Validation Accuracy***	***Inference Time (in seconds)***
BaseModel	70.5 MB	98.2%	10s
BaseModel+ Quantization	16.1 MB	97.8%	12s
BaseModel+Pruning	26.3 MB	96.7%	15s
BaseModel+ Knowledge Distillation	30.0 MB	98.1%	18s

Table 1. Comparison of compression techniques on MNIST Dataset

- Applying quantization to the base model significantly reduces its size from 70.5 MB to 16.1 MB, while maintaining a relatively high accuracy of 97.8%. However, the inference time slightly increases from 10s to 12s.
- Pruning the base model reduces its size to 26.3 MB, with a slight decrease in accuracy to 96.7%. The inference time also increases to 15 seconds compared to the base model.
- Employing knowledge distillation on the base model increases its size to 30.0 MB, with a similar accuracy of 98.1%. However, the inference time further increases to 18s compared to the other techniques.

B. Implementation of Compression Techniques on the CIFAR10 Dataset

Compression Technique	**Evaluation Metrics**		
	Model Size	***Validation Accuracy***	***Inference Time (in seconds)***
BaseModel	70.8 MB	86.7%	18s
BaseModel+ Quantization	30.1 MB	82.5%	23s
BaseModel+ Knowledge Distillation	30.0 MB	81.1%	28s

Table 2. Comparison of compression techniques on CIFAR10 Dataset

- Applying quantization to the base model reduces its size to 30.1 MB, with a decrease in accuracy to 82.5%. However, the inference time increases to 23s compared to the base model.
- Employing knowledge distillation on the base model maintains a similar size of 30.0 MB, but there is a decrease in accuracy to 81.1%. Additionally, the inference time further increases to 28s compared to both the base model and the quantized model.

Overall, the following observations can be made for the compression techniques:

- It can be inferred that quantization significantly reduces the model size in both datasets while maintaining a close to baseline model accuracy. It can also be observed that the quantized models have increased inference times as compared to the baseline models. This is due to the conversion of images from float32 to int8, thus increasing inference time.

- We observed that the overall accuracies are higher for the MNIST Dataset as compared to the CIFAR10 dataset. The difference in accuracies between MNIST and CIFAR10 can also be attributed to the fact that MNIST has black and white images, whereas CIFAR10 has color images.

VI. Conclusion

In this study, we delved into the realm of deep neural network (DNN) compression techniques, specifically targeting the VGG-16 architecture. Our aim was to address the escalating challenges posed by the expanding size and computational intricacies of modern DNNs. Through a meticulous exploration of compression methods such as quantization, pruning, and knowledge distillation, we sought to ascertain their efficacy in reducing model size without compromising performance, focusing on the MNIST and CIFAR-10 datasets. Our investigation unearthed several significant insights. Quantization emerged as a potent method for significantly reducing model size while preserving a commendable level of accuracy. However, the conversion from float32 to int8 led to increased inference times, a trade-off worth considering in deployment scenarios.

While pruning showcased promising reductions in model size, it also introduced a slight accuracy decrement and increased inference times compared to the base model. Nonetheless, its potential for inference cost reduction remains a compelling factor for certain applications. Although knowledge distillation maintained comparable model sizes, it incurred higher inference times and exhibited a slight accuracy decrease relative to other techniques. Our findings underscore the intricate balance between model size reduction and performance preservation inherent in compression techniques, providing invaluable insights for guiding practitioners and researchers in navigating the complexities of DNN deployment, especially in resource-constrained environments. In essence, our study contributes to the advancement of efficient and sustainable deep learning solutions by shedding light on the nuanced interplay between compression techniques and model performance, fostering innovation and accessibility in the field of artificial intelligence.

References

[1] Pang, B., Nijkamp, E., & Wu, Y. N. (2020). Deep learning with TensorFlow: A review. *Journal of Educational and Behavioral Statistics*, *45*(2), 227-248.

[2] Razani, R., Morin, G., Sari, E., & Nia, V. P. (2021). Adaptive binary-ternary quantization. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 4613-4618).

[3] Han, S., Mao, H., & Dally, W. J. (2015). Deep compression: Compressing deep neural networks with pruning, trained quantization, and Huffman coding. arXiv preprint arXiv:1510.00149.

[4] Zhou, A., Yao, A., Guo, Y., Xu, L., & Chen, Y. (2017). Incremental network quantization: Towards lossless cnns with low-precision weights. arXiv preprint arXiv:1702.03044.

[5] Imambi, S., Prakash, K. B., & Kanagachidambaresan, G. R. (2021). PyTorch. *Programming with TensorFlow: Solution for Edge Computing Applications*, 87-104.

[6] Cohen, G., Afshar, S., Tapson, J., & Van Schaik, A. (2017, May). EMNIST: Extending MNIST to handwritten letters. In *2017 International Joint Conference on neural networks (IJCNN)* (pp. 2921-2926). IEEE.

[7] Tammina, S. (2019). Transfer learning using vgg-16 with deep convolutional neural network for classifying images. *International Journal of Scientific and Research Publications (IJSRP)*, *9*(10), 143-150.

[8] Li, S., Jiao, J., Han, Y., & Weissman, T. (2016). Demystifying resnet. *arXiv preprint arXiv:1611.01186*.

[9] Li, H., Kadav, A., Durdanovic, I., Samet, H., & Graf, H. P. (2016). Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*.a

[10] Blakeney, C., Yan, Y., & Zong, Z. (2020). Is pruning compression?: Investigating pruning via network layer similarity. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (pp. 914-922).

[11] L. Li, J. Zhu and M. -T. Sun, *"Deep Learning Based Method for Pruning Deep Neural Networks," 2019 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), Shanghai, China, 2019, pp. 312-317, doi: 10.1109/ICMEW.2019.00-68. keywords: {Network pruning, filter-level, deep learning},*

[12] Rajendiran, U. M., Ajith, K., & Vedak, O. S. (2021). *A Comparison of Model Compression Techniques for VGG'16.*

[13] Shah, J., & Hatgadkar, R. (2020). *Improving Performance for Distributed SGD using Ray.*

[14] Barhate, P., GD, S., & Das, A. (2020). *Partitioning and Co-location in Apache Spark.*

[15] Gupta, K., Kamat, V., & Sridhar, V. (2021). *Performance Analysis of Stand-Alone and Distributed Machine Learning Systems on Different Workloads.*

[16] Hubens, N., Mancas, M., Decombas, M., Preda, M., Zaharia, T., Gosselin, B., & Dutoit, T. (2020, January). An experimental study of the impact of pre-training on the pruning of a convolutional neural network. *In Proceedings of the 3rd International Conference on Applications of Intelligent Systems (pp. 1-6).*

[17] Gao, M., Shen, Y., Li, Q., Yan, J., Wan, L., Lin, D., ... & Tang, X. (2018). *An embarrassingly simple approach for knowledge distillation. arXiv preprint arXiv:1812.01819.*

[18] Li, T., Li, J., Liu, Z., & Zhang, C. (2020). Few sample knowledge distillation for efficient network compression. *In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 14639-14647).*