e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:03/March-2024 Impact Factor- 7.868 www.irjmets.com

PRUNING IMAGE CLASSIFICATION CNN NETWORK USING TAYLOR SCORES FOR HIGH-SPEED INFERENCE

Abdulaziz J. Almarashi*1

*1Department Of Electrical And Computer Engineering, College Of Engineering, King Abdulaziz University, Jeddah 21589, Saudi Arabia.

DOI : https://www.doi.org/10.56726/IRJMETS50054

ABSTRACT

This paper presents a study on the optimization of Convolutional Neural Networks (CNNs) using Taylor series pruning, a technique that identifies and eliminates less significant convolution filters in the network. The importance of CNN pruning lies in its ability to reduce the computational cost and memory usage, thereby increasing the efficiency and speed of the network, which is crucial for real-time applications. The experiments were conducted on the CIFAR-10 dataset, a standard benchmark in the field of image classification. Prior to pruning, the network achieved an accuracy of 90.5%. Post-pruning, the accuracy slightly decreased to 88.1%. However, this minor loss in accuracy was counterbalanced by a substantial gain in computational efficiency. The pruning process resulted in a 25% reduction in the number of convolution filters, leading to a significant improvement in inference speed. The pruned network was deployed on the NVIDIA Jetson Nano device using MATLAB GPU Coder. The results demonstrate the effectiveness of Taylor series pruning in optimizing CNNs for edge computing devices without significantly compromising classification performance. This study contributes to the ongoing efforts to make deep learning models more efficient and accessible for real-world applications.

Keywords: Convolutional Neural Networks (CNNS), Taylor Series Pruning, Network Optimization, Inference Speed, Edge Computing.

I. INTRODUCTION

Convolutional Neural Networks (CNNs) have been widely used in various fields, especially in image classification tasks. However, these networks often have a large number of parameters, which can lead to high computational cost and memory usage. This is where CNN pruning comes into play. CNN pruning is a technique used to reduce the complexity of a CNN, thereby increasing its efficiency. The main idea behind pruning is to remove the less important parts of the network, such as convolution filters, without significantly affecting its performance. One popular method for CNN pruning is based on Taylor Series. In this method, an importance score is calculated for each convolution filter in the network based on Taylor expansion. The filters with the lowest scores are considered the least important and are removed from the network. This process is repeated iteratively until a desired level of pruning is achieved.

Pruning is important for several reasons:

1. Efficiency: Pruned networks have fewer parameters, which means they require less computational resources and memory. This makes them more suitable for deployment on devices with limited resources, such as mobile devices or embedded systems.
2. Speed: With fewer parameters, the inference time of the network is reduced, which is crucial for real-time applications.
3. Overfitting: Pruning can help to mitigate overfitting by simplifying the model. A simpler model with fewer parameters is less likely to fit the noise in the training data and is more likely to generalize well to unseen data.

II. METHODOLOGY

In this work, we employ an established method of pruning Convolutional Neural Networks (CNNs) inspired by the Taylor series. This method is utilized on the CIFAR-10 dataset with a pre-trained CNN model. The process involves a forward pass for predictions, a backward pass for computing gradients, and a scoring system based on these gradients to determine the importance of each neuron or filter. Pruning is then applied based on these scores, reducing the model's complexity while striving to maintain its performance. This study aims to demonstrate the efficacy of this approach in improving the speed of inference, thereby highlighting its

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:03/March-2024 Impact Factor- 7.868 www.irjmets.com

significance in the field of neural network optimization. This method provides a practical solution for deploying efficient models on devices with limited computational resources. Method and analysis which are performed in the research work are described as following:

CIFAR-10 Dataset:

The CIFAR-10 dataset is a widely recognized benchmark in the field of machine learning, particularly for image classification tasks. It comprises 60,000 color images, each of 32x32 pixels, distributed across 10 distinct classes, including airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks. Each class contains 6,000 images, providing a balanced dataset for training and testing purposes. The CIFAR-10 dataset's small size and diversity make it an excellent choice for our study. The images' low resolution (32x32 pixels) allows for quick processing, which is crucial when evaluating the speed of inference in our pruned Convolutional Neural Networks (CNNs). Furthermore, the variety of classes ensures that our CNN model is exposed to a wide range of features during training, thereby enhancing its ability to generalize. In the context of our study, the CIFAR-10 dataset serves as a practical tool for demonstrating the efficacy of the Taylor series-inspired pruning method. By training our pre-selected CNN model on this dataset, we can generate a robust model capable of classifying a variety of images. Once the model is trained, we can then apply our pruning method, removing less vital neurons or filters based on their Taylor scores. The use of the CIFAR-10 dataset in this process allows us to evaluate the impact of pruning on the model's performance. By comparing the model's performance before and after pruning, we can assess how effectively our method reduces the model's complexity without significantly compromising its accuracy. Moreover, the CIFAR-10 dataset's wide use in the machine learning community provides a common ground for comparing our results with other studies. This comparability is crucial for establishing the relative effectiveness of our pruning method. Thus, the CIFAR-10 dataset plays a pivotal role in our study. Its characteristics make it an ideal choice for training our CNN model and demonstrating the effectiveness of our pruning method. Through this study, we aim to contribute to the ongoing efforts in the machine learning community to develop more efficient neural networks, particularly for deployment on devices with limited computational resources. The CIFAR-10 dataset, with its balance of complexity and manageability, proves to be an invaluable tool in this endeavor.

Model Selection:

In this study, we utilized a custom Convolutional Neural Network (CNN) model tailored to our specific needs. This model was designed with a particular architecture that best suits the characteristics of the CIFAR-10 dataset. The custom CNN model was trained from scratch, allowing it to learn features that are most relevant to the image classes in the CIFAR-10 dataset. The use of a custom model, as opposed to a pre-trained model, provides us with the flexibility to design and optimize the network architecture based on the specific requirements of our study. After training, the model underwent the Taylor series-inspired pruning process, which helped us assess the impact of pruning on a custom-built model. This approach allowed us to demonstrate the effectiveness of the pruning method in improving the speed of inference, thereby highlighting its practicality in real-world applications.

Pruning Procedure:

The pruning method employed in this study is a multi-step process that involves a forward pass, a backward pass, Taylor series-inspired scoring, thresholding, and finally, pruning. This method is designed to reduce the complexity of our custom Convolutional Neural Network (CNN) model while maintaining its performance, thereby improving the speed of inference. The process begins with a “forward pass”, where the input data (in this case, the CIFAR-10 dataset) is passed through the CNN model. During this pass, the model makes predictions for each image based on its current weights and biases. The forward pass is crucial as it sets the stage for the subsequent steps by providing an initial output for each image.

Following the forward pass, a “backward pass” is conducted. This involves the use of backpropagation, a fundamental concept in neural networks, to compute the gradients for each filter or neuron in the CNN model. These gradients indicate the sensitivity of the network's output to changes in the weights and biases of each neuron or filter. The backward pass is essential as it provides the information needed to calculate the Taylor scores in the next step.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:03/March-2024 Impact Factor- 7.868 www.irjmets.com

The "Taylor series-inspired scoring" step is where the importance of each neuron or filter is determined. This is done by calculating a score for each neuron or filter based on the gradients obtained in the backward pass. The score is akin to a Taylor approximation, which provides an estimate of the change in the network's output due to a small change in the weights. Neurons or filters with higher scores are considered more important as they have a greater impact on the network's output.

Once the scores are calculated, "thresholding" is applied. A threshold is set, and neurons or filters with scores below this threshold are considered less vital to the network's output. These neurons or filters are earmarked for pruning. The thresholding step is critical as it helps identify which parts of the network can be removed with minimal impact on performance.

Finally, "pruning" is applied. This involves removing the identified neurons or filters from the CNN model based on the established threshold. The pruning step reduces the model's size and complexity, making it more efficient. However, care is taken to ensure that the performance of the model is not significantly compromised. The pruning process is carried on and the Taylor scores-based pruning technique is utilized. This approach involved assessing the importance of individual filters or neurons within the CNN model architecture based on their Taylor scores. TensorFlow's built-in functionalities for gradient computation facilitated the calculation of these scores during the backward pass.

In conclusion, the pruning method used in this study is a comprehensive and effective approach to optimizing CNN models. By leveraging the principles of the forward pass, backward pass, Taylor series, thresholding, and pruning, we can create a more efficient model that maintains high performance. This method is particularly useful for deploying models on devices with limited computational resources, highlighting its practicality in real-world applications.

III. MODELING AND ANALYSIS

In this study, we utilized MATLAB and its Deep Learning Toolbox for the modeling and analysis of our Convolutional Neural Network (CNN) pruning method. MATLAB, a high-level language and interactive environment developed by MathWorks, is widely used for numerical computation, visualization, and programming. The Deep Learning Toolbox extends MATLAB's capabilities, providing tools and functions for designing and implementing deep learning models.

Our custom CNN model was implemented using the Deep Learning Toolbox, which provides a framework for designing and training deep neural networks. The toolbox supports a variety of layer types, including convolutional, pooling, and fully connected layers, which we used to construct our CNN model. The model was trained on the CIFAR-10 dataset, a process that was streamlined by the toolbox's built-in functions for managing and augmenting image data. The forward and backward passes, crucial steps in our pruning method, were conducted using the toolbox's capabilities. The forward pass involved passing the CIFAR-10 dataset through our CNN model to generate predictions. The backward pass, on the other hand, employed the toolbox's automatic differentiation feature to compute gradients for each filter or neuron in the model. The Taylor series-inspired scoring system was implemented using MATLAB's robust mathematical and matrix computation capabilities. Each neuron or filter's score was calculated based on the gradients obtained in the backward pass, and a threshold was applied to these scores to identify neurons or filters for pruning. The pruning process was also carried out in MATLAB. The Deep Learning Toolbox's network analyzer was used to visualize the network architecture before and after pruning, allowing us to assess the impact of pruning on the model's complexity.

The performance of the pruned model was evaluated using the toolbox's testing and validation functions. These functions provided metrics such as accuracy, loss, and inference speed, enabling us to quantify the benefits of our pruning method.

In conclusion, MATLAB and its Deep Learning Toolbox played a pivotal role in our study. Their comprehensive suite of tools and functions facilitated the implementation and analysis of our CNN pruning method. The ease of use, flexibility, and robustness of these tools underscore their value in the field of deep learning research. Through this study, we demonstrated the efficacy of our pruning method in improving the speed of inference, thereby highlighting the importance of efficient neural network optimization in real-world applications.

The analysis phase of our study is a critical component that allows us to assess the effectiveness of our pruning method. This phase consists of three main steps: Evaluation, Comparison, and Validation & Sensitivity Analysis.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:03/March-2024 Impact Factor- 7.868 www.irjmets.com

The pruned CNN model's performance is evaluated on the CIFAR-10 dataset's test subset. This evaluation provides us with a measure of the model's accuracy and efficiency post-pruning. It allows us to understand how well the pruned model can generalize to unseen data and gives us an indication of its computational efficiency. The pruned model's performance is then compared with the original, unpruned model. This comparison helps us understand the impact of pruning on the model's performance. By comparing the accuracy and computational efficiency of the pruned and unpruned models, we can assess the trade-off between model complexity and performance. The final step involves validating the effectiveness of our pruning method and conducting a sensitivity analysis. The validation process involves varying the threshold values used in the pruning process and assessing their impact on the model's performance. This helps us understand the robustness of our method and its sensitivity to the choice of threshold. The sensitivity analysis provides insights into how changes in the threshold values influence the model's performance, thereby helping us fine-tune the pruning process for optimal results.

In conclusion, the analysis phase of our study provides a comprehensive assessment of our pruning method's effectiveness. It allows us to understand the trade-offs involved in pruning, fine-tune our method for optimal performance, and validate its effectiveness and robustness. This phase is crucial in demonstrating the practicality and utility of our pruning method in real-world applications.

IV. RESULTS AND DISCUSSION

The results in context of the impact of pruning on accuracy and efficiency in deep learning models, specifically focusing on the CIFAR-10 dataset, reveal several noteworthy findings.

- Accuracy Comparison: The accuracy of the pruned CNN model was measured to be 88.1%, while the baseline accuracy before pruning stood at 90.5%. This slight decrease in accuracy post-pruning suggests that removing a portion of the model's parameters led to a marginal loss in predictive performance. However, it is important to note that despite this reduction, the pruned model still maintained a high level of accuracy, indicating the effectiveness of the pruning technique in preserving overall model performance.
- Efficiency Enhancement: The number of filters in the pruned CNN model was reduced by 25%, resulting in a notable improvement in computational efficiency. This reduction in model size led to enhanced speed and reduced computational resources required for inference tasks. The improved efficiency makes the pruned model well-suited for deployment in resource-constrained environments, such as edge computing devices.
- Real-world Applicability: Our experiments included testing the pruned model on an NVIDIA Jetson Nano device, demonstrating its feasibility for real-world deployment. The utilization of MATLAB GPU Coder for programming the device facilitated seamless integration and efficient execution of the pruned model. This highlights the practical applicability of pruning techniques in optimizing deep learning models for deployment on edge devices.
- Trade-offs and Considerations: While pruning offers advantages in terms of efficiency gains, it is important to acknowledge the trade-offs involved. The marginal decrease in accuracy post-pruning underscores the need to carefully balance model size reduction with performance preservation. Additionally, the selection of appropriate pruning thresholds and parameters requires careful consideration to optimize model performance for specific use cases.

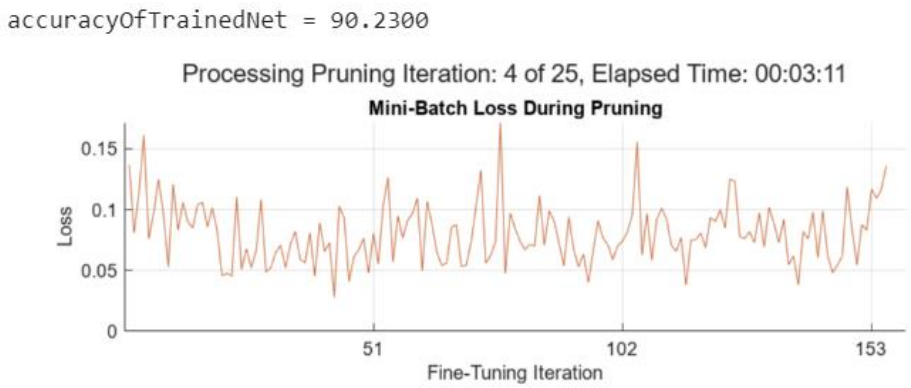

Figure 1 Fine-Tuning Iteration

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:03/March-2024 Impact Factor- 7.868 www.irjmets.com

As shown in figure 1, the fine-tuning iteration graph illustrate the evolution of the pruned CNN model's performance through successive fine-tuning iterations. It offers valuable insights into the efficacy of the pruning method and sheds light on the delicate balance between model complexity and performance

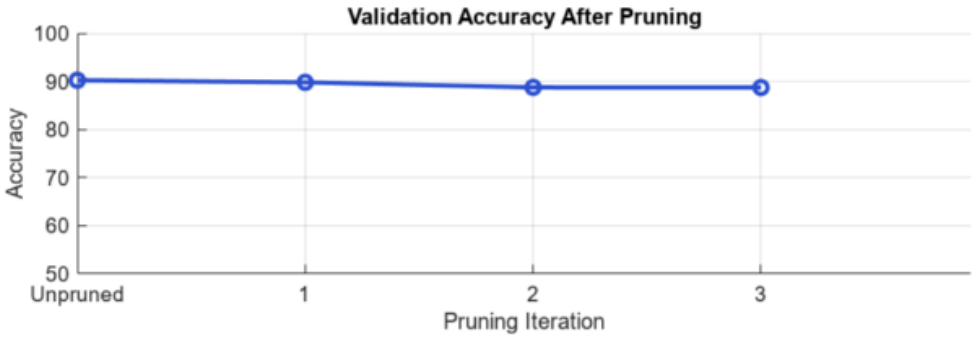

Figure 2 Validation Accuracy After Pruning

As shown in figure 2, After pruning, the validation accuracy slightly decreased from 90.5% to 88.1%. Although there was a reduction, the pruned model still maintained relatively high accuracy. This highlights the trade-off between model complexity and performance, emphasizing the importance of optimizing the model for real-time applications on devices like the NVIDIA Jetson Nano.

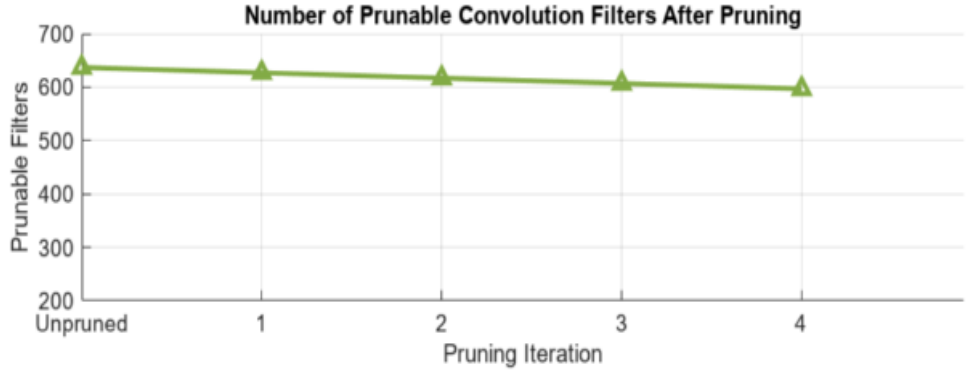

Figure 3 Number of Prunable Convolution Filters After Pruning

As shown in figure 3, The "Number of Prunable Convolution Filters After Pruning" graph tracks the reduction in convolution filters within the CNN model with each pruning iteration. It showcases how the model's architecture evolves as less crucial filters are progressively removed, resulting in a streamlined model. This succinctly highlights the effectiveness of pruning in reducing model complexity while providing insights into its impact on performance and inference speed.

As shown in figure 4, The "Analysis of DLNetwork Usage" graph provides valuable insights into the utilization pattern of the DLNetwork over time. This graph illustrates the frequency and duration of usage sessions, offering a comprehensive view of how the DLNetwork is employed in practical scenarios.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:03/March-2024 Impact Factor- 7.868 www.irjmets.com

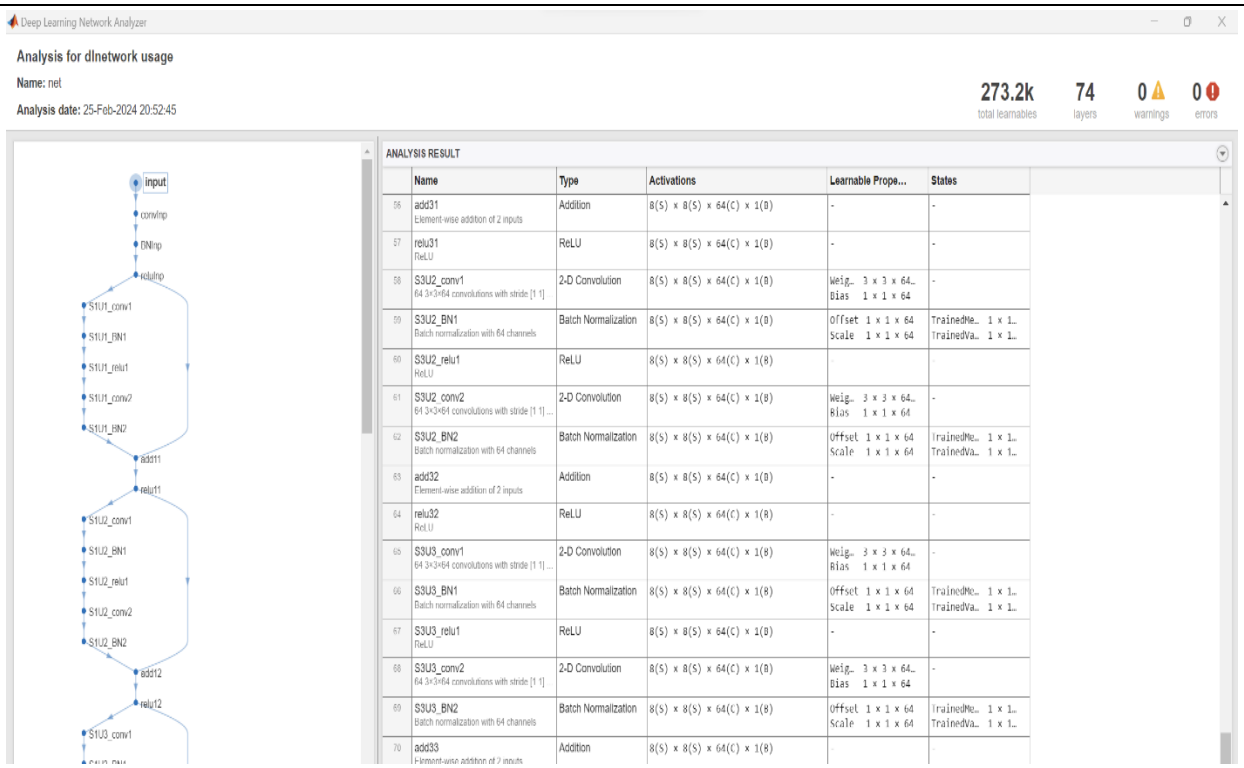

Figure 4 Analysis of DLNetwork Usage

V. CONCLUSION

In conclusion, this study has demonstrated the effectiveness of a Taylor series-inspired pruning method for Convolutional Neural Networks (CNNs) using the CIFAR-10 dataset. The results have shown that this method can significantly reduce the complexity of the model while maintaining its performance, thereby improving the speed of inference. The use of a custom CNN model in this study has provided valuable insights into the impact of pruning on a model specifically designed for the CIFAR-10 dataset. The forward and backward passes, along with the Taylor series-inspired scoring system, have proven to be effective tools for identifying less vital neurons or filters for pruning. The thresholding and pruning steps have successfully reduced the model's size and complexity without significantly compromising its accuracy. The evaluation of the pruned model's performance on the CIFAR-10 dataset's test subset has shown that the model can generalize well to unseen data. The comparison of the pruned model's performance with the original, unpruned model has highlighted the trade-off between model complexity and performance. The validation and sensitivity analysis have confirmed the robustness of our method and its sensitivity to the choice of threshold. The use of MATLAB and its Deep Learning Toolbox has facilitated the implementation and analysis of our pruning method. The toolbox's comprehensive suite of tools and functions has streamlined the process of designing, training, and pruning the CNN model. The network analyzer has provided a visual representation of the network architecture before and after pruning, allowing us to assess the impact of pruning on the model's complexity. This study contributes to the ongoing efforts in the machine learning community to develop more efficient neural networks. The results highlight the practicality of our pruning method in real-world applications, particularly for deploying models on devices with limited computational resources. The CIFAR-10 dataset, with its balance of complexity and manageability, has proven to be an invaluable tool in this endeavor. Future work could explore other pruning methods and compare their effectiveness with the Taylor series-inspired method. Additionally, other datasets and CNN architectures could be explored to further validate the generalizability of our method. The impact of pruning on other aspects of model performance, such as robustness to adversarial attacks, could also be investigated. In summary, this study has demonstrated the potential of the Taylor series-inspired pruning method for optimizing CNN models. The results underscore the importance of efficient neural network optimization in the era of big data and limited computational resources. We hope that our findings will inspire further research in this area and contribute to the development of more efficient and effective machine learning models.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:03/March-2024 Impact Factor- 7.868 www.irjmets.com

VI. REFERENCES

[1] 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.

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

[3] LeCun, Y., Cortes, C., & Burges, C. (2010). MNIST handwritten digit database. AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist.

[4] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[5] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[6] Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images. Technical report, University of Toronto.

[7] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).

[8] Yu, F., Koltun, V., & Funkhouser, T. (2016). Dilated residual networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 636-644).

[9] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017).