Comparative Analysis of ResNet-152 and VGG-19 Architectures on CIFAR-10 Dataset: Benchmarking and Parameter Optimization

Sally Yu
s8yu@ucsd.edu

I. Abstract

This project is focused on comparing two different existing architectures, ResNet152 and VGG19, using the popular image classification benchmarking dataset CIFAR-10. The aim is to assess how various hyperparameters influence model performance enhancement. The hyperparameters being examined includes batch normalization, image augmentations, pooling functions, activation functions, optimization strategies, and learning rate schedulers. Through comprehensive testing, both models demonstrated over a 4% increase in accuracy, with the VGG-19 model notably achieving a 9% improvement after hyperparameter optimization.

II. Introduction

The field of deep learning and AI has been a hot topic in the past few decades, especially receiving the general public's attention in the past few years after the emergence of Transformers and Large Language Models (LLMs). Innovations in algorithms and architectures are continuously being built every year, setting new benchmarks for state-of-the-art performance in every subfield.

Image classification remains as one of the key areas of focus for researchers. Central to this subfield is the concept of convolutional neural networks (CNNs). Architectures that achieve high performance in image classification benchmarks are mostly CNNs [1]. ResNet-152 and VGG-19 are two of the exemplar models in this context, each demonstrating unique strengths in handling complex image data to achieve high accuracy.

VGG-19, developed by the Visual Geometry Group at the University of Oxford, exemplifies a simple yet deep architecture characterized by its consistent use of 3x3 convolutional filters and a depth of 19 layers [2]. It has been widely adopted for various image processing tasks due to its uniform and simple architecture.

On the other hand, ResNet-152, introduced by Kaiming He et al. [3] a year after the release of VGG-19, represents a leap in architectural design through the introduction of residual blocks. This network consists of a total of 152 layers, much deeper than the 19 layers in VGG-19. It employs skip connections that allow it to train very deep networks by addressing the vanishing gradient problem, which was a significant hurdle for training deep architectures in the past. The residual connections enable the network to learn identity functions, ensuring that the deeper layers can perform as well as shallower ones, without compromising training performance. As a result, ResNet-152 is known for achieving high accuracy results on challenging datasets with much more efficient convergence than other deep networks.

In this project, I aim to conduct a detailed comparative analysis of the ResNet-152 and VGG-19 architectures on the CIFAR-10 dataset. By examining their baseline performances as well as exploring potential enhancements through hyperparameter optimization, the goal is to identify key factors that most significantly enhance accuracy and generalization capabilities. The analysis is intended to provide insights into the further development and advancement of CNNs for more sophisticated image classification and processing tasks.

III. Data

CIFAR-10 is a popular open-source dataset commonly used for testing and benchmarking classification algorithms. Comprised of 60,000 rgb images of size 32 by 32 pixels with 10 classes of objects: airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks. Its size and complexity makes it a great resource for testing new image-classification algorithms.

IV. Methods

A comparison of baseline performance for both ResNet 152 and VGG 19 is first evaluated by training both architectures with the same set of parameters for 10 epochs, freezing all layers except the last fully-connected layer. The default settings are { activation_function=ReLu, pooling_function=MaxPool, no batch normalization, criterion=CrossEntropyLoss, learning_rate=0.001, optimizer=Adam, batch_size=16 }.

The second phase is focused on hyperparameter tuning of the two baseline models. A wide range of hyperparameters are tested individually on ResNet-152 and VGG-19. These parameters includes image augmentation techniques (Rotation, Color Jitter, Random Crop, and Random Flip), Activation functions (Rectified Linera Unit, Leaky Rectified Linera Unit, Parametric Rectified Linera Unit, Exponential Linear Unit), pooling functions (max pooling, average pooling), batch normalization, optimization functions (Adam, Adam Weight Decay, Stochastic Gradient Descent), learning rate scheduler.

V. Experiments

A. *Training Process*

Both ResNet-152 and VGG-19 models are trained on the training set of 50,000 images with the same default parameters for 10 epochs. The training loss curve is shown in figure 1. Both models exhibit a consistent decline in loss, indicative of effective throughout the epochs. In comparison, ResNet-152 baseline outperforms VGG-19 by 2.46% accuracy – 82.32% for ResNet-152 and 79.86% for VGG-19.

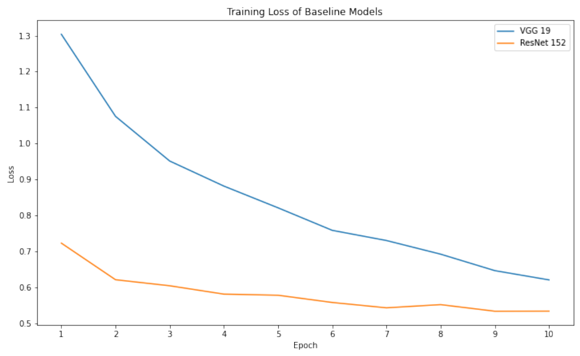


Fig. 1: Baseline Models Training Process

B. Hyperparameter Tuning

Different hyperparameters are sequentially applied to each model to assess their impact on performance enhancement. For ResNet-152, a thorough hyperparameter tuning on random rotation (image augmentation) applied to the training data is tested. As shown in figure 2, random rotation of up to 15 degrees yielded the highest accuracy of 86.95%. With additional evaluation done on ResNet-152 with different optimization functions: default Adam [4], AdamW with weight decay of 0.1, and Stochastic Gradient Descent (SGD) as well as other augmentation techniques including Random Crop, Color Jitter, Batch Normalization, results still show that the default Adam optimizer combined with 15 degrees of rotation remains as the highest performing model with accuracy of 86.95%.

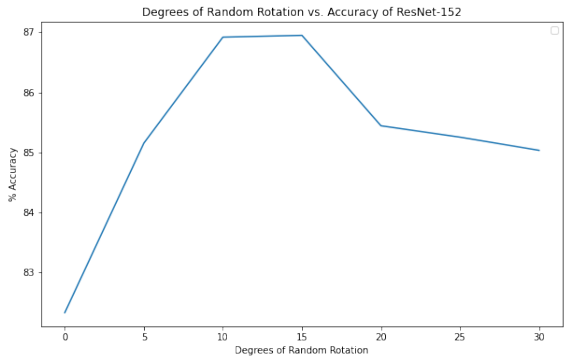


Fig. 2: Image Augmentation - Random Rotation

For VGG-19, various activation functions were tested to replace the standard ReLU in the final fully connected layer in the VGG-19 architecture. Among ReLU, LeakyReLU, PReLU, and ELU, PReLU was most effective, enhancing the accuracy of baseline model by 2% from 82.99% to 84.99%. For pooling functions, average pooling was used to replace the default max pooling in the last layer, and performance did not surpass the baseline with max pooling. Additionally, The same set of image augmentations applied to ResNet-152 were evaluated on VGG-19. The combination of Random Horizontal Flip and Color Jitter is found to have the best performance boost, reaching an accuracy of 87.43%. During this phase, batch size was also periodically increased to reduce training time, which accounted for some of the improvements in accuracy score. Further refinement involved optimizing batch normalization parameters from the conventional values to using the dataset's mean and standard deviation. Coupled with the implementation of a Step learning rate scheduler, these adjustments culminated in the VGG-19 model achieving an overall accuracy of 89.02%.

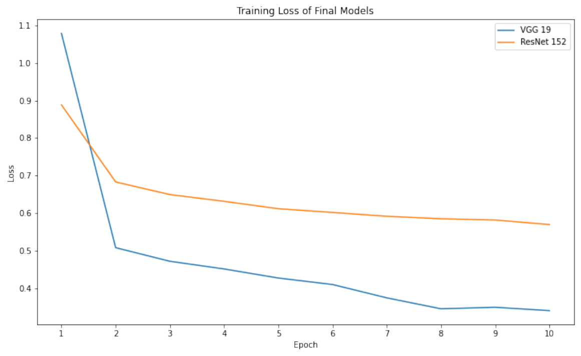
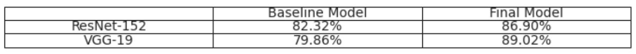


Fig. 3: Final Models Training Process

C. Results



	Baseline Model	Final Model
ResNet-152	82.32%	86.90%
VGG-19	79.86%	89.02%

Fig. 4: Model Performance Comparison Table

The final ResNet-152 and VGG-19 models achieved accuracy of 86.90% and 89.02% respectively. Baseline and final model accuracies for both architectures are shown in figure 4. In comparison, VGG-19 model observed a large improvement of around 9% while ResNet-152 was only able to improve by a 4% after hyperparameter optimization.

As observed in the training loss plots (figure 3), ResNet-152 initiates training with a relatively lower loss, but experiences only marginal declines throughout the training epochs. On the other hand, the final VGG-19 model begins with greater loss but exhibits significant loss attenuation during the convergence process, especially in the first 2 epochs, allowing it to outperform the final ResNet-152 model despite lower performance on the baseline.

This is a little unexpected due to the notion that the deeper a network, the larger its capacity to learn more complex data representations, hence in general yielding better performance. When comparing between ResNet-152 and VGG-19, VGG-19 has fewer layers and is much simpler in structure. However, it's able to reach a higher level of accuracy on the CIFAR-10 dataset than the ResNet-152 architecture after hyperparameter tuning. This further highlights that depth and complexity of CNNs do not always correlate with enhanced performance, and that certain architectures are suited for specific problems than others.

The final model configurations for both architectures are shown in table 1 and table 2. Different set of hyperparameters were found to have the most effect on increasing model performance, which suggests that a straightforward transfer of optimal parameters between models does not guarantee improvement in performance and a more tailored hyperparameter tuning approach need to be taken in order to unlock the full potential of each model due to the intricate nature of convolutional neural networks.

ResNet-152 Configuration	
Pooling	Max Pooling
Activation	ReLU
Optimizer	Adam (learning rate = 0.001)
Scheduler	StepLR
Augmentation	Random Rotation (15 degrees)

TABLE I: ResNet-152 Model Configuration

VGG-19 Configuration	
Pooling	Max Pooling
Activation	PReLU
Optimizer	Adam (learning rate = 0.001)
Scheduler	StepLR
Augmentation	ColorJitter, RandomHorizontalFlip
Normalization	Mean & Standard Deviation

TABLE II: VGG-19 Model Configuration

VI. CONCLUSIONS

After an in-depth examination of hyperparameter tuning for the ResNet-152 and VGG-19 architectures on the CIFAR-10 dataset, the key takeaway is the distinct preferences for specific parameters and values by different architectures. While certain parameters, such as random rotation augmentation, noticeably enhanced the accuracy of ResNet-152, applying the same adjustments to VGG-19 did not yield similar improvements, and visce versa. In addition, specific parameters may enhance model performance by itself, but when applied in combination with other parameters, the outcomes may be suboptimal; therefore, a combined hyperparameter search needs to be done to ensure most optimal combinations. These findings underscore the nuanced nature of CNN's response to hyperparameter adjustments and emphasize the importance of testing and tailoring specific optimization strategies for each distinct model.

REFERENCES

[1] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," 2015.
[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.
[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.
[4] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

APPENDIX

Below are the links to the resources and documentation referenced in this paper:

- CIFAR-10 and CIFAR-100 datasets: https://www.cs.toronto.edu/~kriz/cifar.html
- ResNet152 Documentation: https://pytorch.org/vision/main/models/generated/torchvision.models.resnet152.html
- VGG19 Documentation: https://pytorch.org/vision/main/models/generated/torchvision.models.vgg19.html