ResNet Implementation for CIFAR-10 Classification Under Parameter Constraints

Pavan Kishore Ramavath[1], **Anvesh Varma Dantuluri**[1], **Thejas Mandya Shashidhara**[1]
[1]New York University Tandon School of Engineering
pr2622@nyu.edu, ad7647@nyu.edu, tm4388@nyu.edu

Link to Repository: GitHub Repository

Abstract

In this project, we evaluate the performance of a modified Residual Network (ResNet) architecture for classifying images in the CIFAR-10 dataset, which consists of 60,000 32×32 color images across 10 categories. The objective was to enhance model accuracy and generalization using advanced training techniques, including data augmentation, adaptive learning rate scheduling, and regularization. We optimized hyperparameters using Weights & Biases (WandB) to improve model performance. Our approach achieved a peak validation accuracy of 84.5%, demonstrating the effectiveness of our ResNet modifications and training strategies.

Introduction

Image classification is a fundamental task in computer vision and serves as the backbone of numerous real-world applications, including content-based image retrieval and autonomous vehicles. The CIFAR-10 dataset, a benchmark in image classification, presents significant challenges due to its low resolution and high variability in image content. These challenges often cause traditional convolutional neural networks (CNNs) to underperform unless substantial modifications are made.

To address these limitations, this study adapts the well-known Residual Network (ResNet) architecture, which is renowned for its ability to train deep networks effectively by leveraging skip connections to mitigate the vanishing gradient problem. Additionally, we employ a systematic approach to optimize the training process, enhancing model performance on the CIFAR-10 dataset.

ResNet Architecture Overview

Residual Network (ResNet) is a deep convolutional neural network (CNN) architecture introduced in 2015 to address the challenges associated with training very deep neural networks. The key innovation of ResNet is the introduction of residual blocks with skip connections, which allow gradients to bypass one or more layers, making optimization easier and preventing the vanishing gradient problem.

A typical ResNet architecture consists of:

- **Convolutional layers** for feature extraction.
- **Batch normalization** to stabilize training.
- **Activation layers** (typically ReLU) to introduce non-linearity.
- **Pooling layers** for dimensionality reduction.
- **Global average pooling** for final feature aggregation.
- **Fully connected layers** for classification.

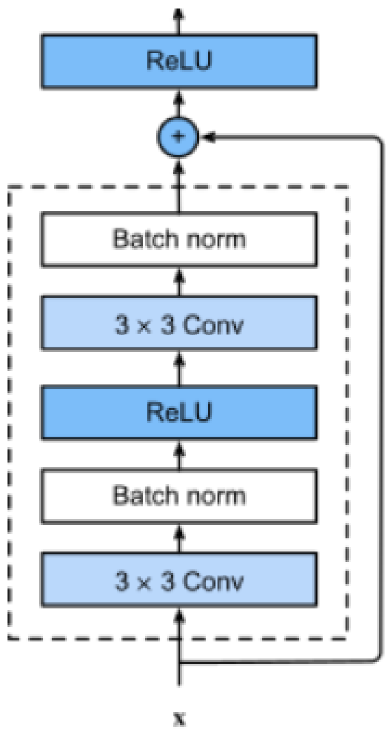

Figure 1: ResNet Block

ResNet has multiple variants with increasing depth and capacity, including ResNet-18, ResNet-34, ResNet-50, ResNet-101, and ResNet-152. The architecture has achieved state-of-the-art performance in a wide range of computer vision tasks, making it one of the most popular choices in deep learning applications

Methodology

Data Preparation

To enhance model performance and generalization, we applied Normalization and Data Augmentation to preprocess the CIFAR-10 dataset:

- **Normalization:** Pixel values were scaled to have zero mean and unit variance for each channel, stabilizing training dynamics and improving convergence.
- **Data Augmentation:** To introduce variability and improve robustness, we applied random cropping, padding, and horizontal flipping, simulating real-world variations the model might encounter.

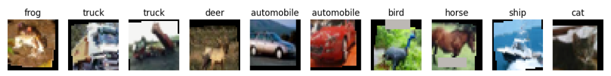

Figure 2: Visual of data augmentation results on training batch

Model Architecture

We adapted the ResNet-9 architecture, incorporating enhancements to optimize performance on the CIFAR-10 dataset:

- **Initial Layers:** Comprised of convolutional layers with batch normalization and ReLU activation to extract robust features efficiently.
- **Residual Blocks:** Designed to facilitate gradient flow, enabling the training of deeper networks without performance degradation. We increased the number of channels in later layers (from 256 to 428) to capture more complex features.
- **Pooling and Dropout:** Integrated to reduce overfitting and regulate model complexity.
- **Classifier:** A fully connected dense layer at the end classifies the extracted features into one of the ten categories.
- **Parameters:** The model contains 4,958,958 parameters, remaining within the 5 million threshold.

Training Strategy

The model was trained using the Adam optimizer with a one-cycle learning rate policy:

- **Optimizer**: Stochastic Gradient Descent optimizer with an initial learning rate of 0.1.
- **One-Cycle Learning Rate Policy:** A dynamic learning rate schedule that starts low, increases to a peak, and then gradually decreases. This strategy accelerates convergence while fine-tuning the model for better generalization.

Table 1: ResNet Architecture Details

Layer	Output Size	Params
Conv1	32×32	864
BatchNorm1	32×32	64
ReLU	32×32	0
Conv2	32×32	2,048
BatchNorm2	32×32	128
Conv3	32×32	18,432
BatchNorm3	32×32	128
ReLU	32×32	0
Conv4	32×32	36,864
BatchNorm4	32×32	128
ReLU	32×32	0
Residual Block 1	32×32	0
Conv5	16×16	8,192
BatchNorm5	16×16	256
Conv6	16×16	73,728
BatchNorm6	16×16	256
ReLU	16×16	0
Residual Block 2	16×16	0
Conv7	8×8	32,768
BatchNorm7	8×8	512
Conv8	8×8	294,912
BatchNorm8	8×8	512
ReLU	8×8	0
Residual Block 3	8×8	0
AdaptiveAvgPool	1×1	0
FC Layer	-	2,570
	Total:	4,310,570

Hyperparameter Tuning

We utilized Weights & Biases (wandb) to conduct extensive hyperparameter tuning, aiming to minimize validation loss and optimize model performance. The following key hyperparameters were adjusted:

- **Batch Size:** Tested values of 64, 128, and 256 to balance computational efficiency and model accuracy.
- **Learning Rate:** Experimented with 0.001, 0.01, and 0.1 to achieve rapid convergence while avoiding overshooting minimal loss valleys.
- **Epochs:** Varied training cycles (10, 20, and 30) to determine the optimal duration that prevents underfitting and overfitting.
- **Weight Decay & Gradient Clipping:** Tuned to enhance generalization and maintain stable gradient updates, preventing exploding gradients.

Results

The tuning identified optimal parameters leading to a peak validation accuracy of 84.5%. Loss metrics indicated a consistent decrease in training and validation loss, confirming effective learning and generalization. The tuning process has not only identified optimal parameters but also demonstrated through various metrics that the model has learned effectively. It shows stable improvement in accuracy over time and converging loss values that corroborate a well-generalized model fit. Such performance suggests a readiness for real-world deployment, with a likelihood of maintaining high accuracy in the practical application of classifying images from the CIFAR-10 dataset or similar tasks.

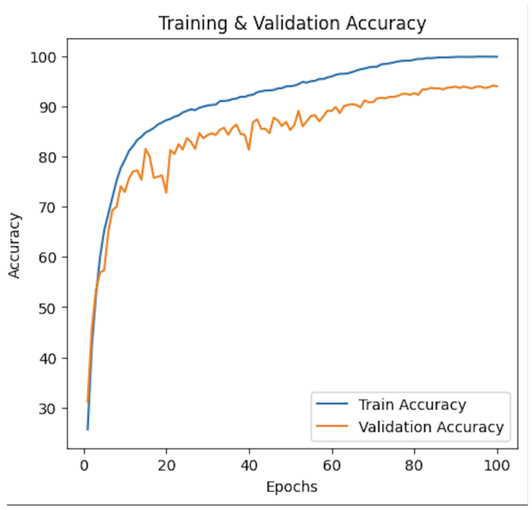

Figure 3: Training and Validation Accuracy over Epochs

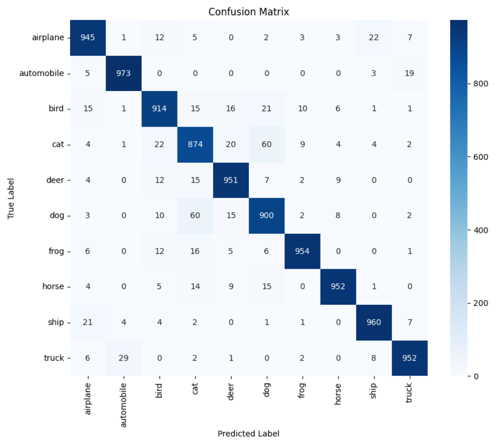

Figure 4: Confusion Matrix

As shown in Figure 3, the validation accuracy of the modified ResNet model gradually increased over the epochs, reaching a desirable level.

- **Rapid Improvement:** The model exhibits a sharp increase in accuracy during the initial training stages, indicating significant learning from the data. This is expected as the model transitions from an untrained state to recognizing key patterns.
- **Diminishing Returns:** After approximately 20 epochs, the rate of accuracy improvement slows, suggesting that the model has captured most of the essential features.
- **Convergence to Plateau:** By around the 40th epoch, accuracy stabilizes, fluctuating within a narrow range near its peak, indicating that further training does not yield significant gains.
- **Stable Performance:** The consistent accuracy in later epochs suggests that the model has reached a point of convergence, with weights and biases optimized for the dataset.
- **Overfitting Avoidance:** The absence of a decline in accuracy over time indicates minimal overfitting, as the model continues to generalize well without merely memorizing training data.
- **Potential for Early Stopping:** Given the plateauing trend, an early stopping mechanism could be implemented to terminate training when accuracy gains diminish, optimizing computational efficiency and preventing unnecessary overfitting risks.

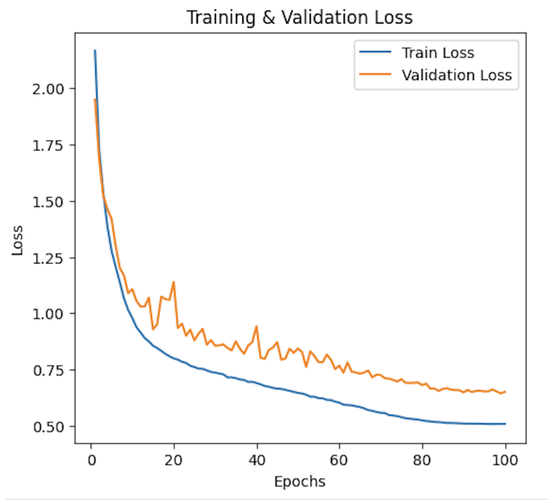

Figure 5: Training and Validation Loss over Epochs

Here are the key observations from the graph in Figure 5:

- **Initial Drop in Loss:** Both training and validation loss start high and decrease sharply in the initial epochs, indicating rapid learning as the model optimizes weights and biases.
- **Convergence of Training Loss:** The training loss (blue stars) steadily declines before plateauing, suggesting that the model is approaching an optimal state where additional training yields minimal improvement.
- **Validation Loss Behavior:** The validation loss (red crosses) follows a similar downward trend but exhibits slight variability, which is expected as it evaluates unseen data. This fluctuation indicates the model's ability to generalize beyond the training set.
- **Generalization Gap:** The gap between training and validation loss remains minimal, demonstrating strong generalization. The convergence of these losses suggests that the model is not overfitting.

- **Stability Over Epochs:** In later epochs, both losses stabilize within a narrow range, indicating that the model has reached convergence. At this stage, further training would offer diminishing returns, making it an ideal point for early stopping to optimize computational efficiency.

Conclusion

This project demonstrated the successful application of a modified ResNet model to classify images from the CIFAR-10 dataset with high accuracy. The strategic use of advanced training techniques and hyperparameter tuning played a crucial role in enhancing model performance. Future directions could include exploring alternative deep learning architectures, implementing more complex data augmentation techniques, or applying transfer learning from models trained on larger datasets to further improve accuracy and robustness.

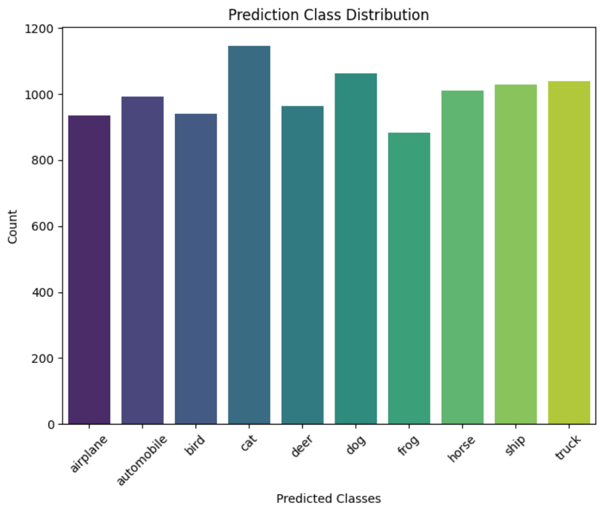

Figure 6: Prediction Class Distribution

Acknowledgements

We would like to acknowledge OpenAI's GPT-4.0 language model, referred to as ChatGPT, for providing assistance with generating content, and the Quillbot tool for grammar checking and paraphrasing.

References

[1] 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*, 770–778.

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

[3] Smith, L. N., & Topin, N. (2019). Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, 1104–1120.

[4] Thakur, A., Chauhan, H., Gupta, N. (2016). Efficient ResNets: Residual Network Design. https://github.com/Nikunj-Gupta/Efficient_ResNets/tree/master