Project 1

● Graded

Group

Sahil Sarnaik
Pranav Tushar Pradhan
Rohan Gore
View or edit group

Total Points

14 / 15 pts

Question 1

Report

7 / 7 pts

✓ **– 0 pts** Correct

- **– 1 pt** Minor error in AAAI format.
- **– 2 pts** Major error in AAAI format.
- **– 1 pt** Missing/Insufficient summarization findings
- **– 0.5 pts** Results section not properly discussed.
- **– 1 pt** Methodology section not properly defined.
- **– 7 pts** Codebase missing or not clickable

Question 2

Codebase

Resolved **7** / 7 pts

✓ **– 0 pts** Correct

– 1 pt Training plots missing

– 1 pt Number of parameters missing

– 1 pt Final test accuracy missing

– 7 pts Codebase missing

– 1 pt For late submission

– 2 pts Data Leakage

Regrade Request **Submitted on: Apr 13**

Our 1 mark was deducted with the comment "Final Test accuracy missing" in the codebase. The final test accuracy, i.e. the final validation accuracy is already printed in the codebase, uploaded on github. The same is also mentioned under the metrics table in the report. The final accuracy is printed in the 13th cell in the jupyter notebook. Can you please take a look at it?

Hi,
I have updated your marks

Reviewed on: Apr 18

Question 3

Top 20% finish

0 / 1 pt

+ 1 pt Top 20%

– 1 pt Late Submission

✓ **+ 0 pts** Not in top 20%

Questions assigned to the following page: 3, 1, and 2

Deep Learning Project - 1

Group Name: Maharashtra Shasan
Rohan Gore (rmg9725), Pranav Pradhan(pp3051), Sahil Sarnaik(ss19100)

New York University
Github

Abstract

We present a modified ResNet architecture for CIFAR-10 image classification, designed to maximize accuracy within a 5 million parameter constraint. Our approach systematically explores the ResNet design space, focusing on the interplay between channel depth, block structure, and skip connection implementation. We used data augmentation, including random crops, horizontal flips, and color jittering, along with an optimized learning rate schedule using a multi-step decay. The resulting model, containing 4,985,390 parameters, achieves a validation accuracy of 94.80% and Kaggle competition score of 0.83680. This report provides a comprehensive analysis of our design methodology, highlighting the trade-offs between model complexity and performance, and offering insights for efficient deep learning model design.

Introduction

The CIFAR-10 dataset comprises 60,000 32x32 color images distributed across 10 mutually exclusive classes (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck), with 6,000 images per class. This dataset's relatively small image size and well-defined classes make it suitable for exploring novel network architectures and training strategies. However, achieving high accuracy on CIFAR-10 remains a challenging task, particularly under resource constraints.

The primary objective of this project is to develop a modified ResNet architecture that achieves high classification accuracy on CIFAR-10 while adhering to a strict limit of 5 million trainable parameters. This constraint necessitates a careful balance between model complexity (depth and width) and performance. We build upon the ResNet-44 architecture, as described in the original ResNet paper (He, Zhang, Ren, and Sun 2015), and systematically investigate modifications to its structure and training procedure.

The key contributions of this work are summarized as follows:

- Design and implementation of a customized, parameter-efficient ResNet architecture specifically tailored for the CIFAR-10 dataset, derived from a ResNet-44 baseline.
- Implementation of a comprehensive data augmentation pipeline, incorporating techniques such as random cropping, horizontal flipping, rotations, and color jittering, to improve model generalization and robustness.
- Exploration and evaluation of various optimization strategies, including the use of the Stochastic Gradient Descent (SGD) optimizer with momentum and a multi-step learning rate scheduler.
- Achievement of a **validation accuracy of 94.80%** with a model containing **4,985,390 parameters**, demonstrating the feasibility of high performance under tight resource constraints.
- Analysis of the impact of architectural choices and training hyperparameters on model performance, providing insights into efficient ResNet design for image classification.

Methodology

Our project aimed to develop a modified ResNet architecture for CIFAR-10 classification under the constraint of having fewer than 5 million parameters while achieving high test accuracy on a competitive Kaggle split. We first experimented with two alternative architectures before finalizing the model described below.

Our early experiments explored two designs. In the first experiment, the network began with 32 input channels, and each residual block had 32 channels in the first stage, followed by two additional stages with 7 blocks each containing 64 and 128 channels, respectively. This model had approximately 2.64 million parameters, achieved a validation accuracy of 93.4%, and a Kaggle score of 0.83961 but overfitted—as evidenced by a training accuracy of 99.96%. In a second iteration, we increased the initial channels to 64 and adopted a structure with residual stages of reduced depths—4 blocks with 64 channels, 4 blocks with 128 channels, and finally 3 blocks with 256 channels—totaling around 4.99 million parameters, as per the findings presented by (Thakur, Chauhan, and Gupta 2023). Although this model exhibited balanced training and validation accuracies (both near 92%), the Kaggle score dropped to approximately 0.82, prompting a design revision as suggested in the work by (He, Zhang, Ren, and Sun 2015). We also employed squeeze and excitation block (Hu et al. 2019) for regularization and look ahead (Zhang, Lucas, Hinton, and Ba 2019) with cosine annealing and cross entropy for optimization. We also performed hyperparameter tuning for this architecture but to no avail. (Snoek, Larochelle, and Adams

2012)

Network Architecture

The current network architecture is based on a custom residual architecture inspired by the ResNet family (He, Zhang, Ren, and Sun 2015). The core building block is a residual unit (or BasicBlock) that performs two successive convolution operations with batch normalization and ReLU activations. A shortcut connection either directly passes the input forward (when dimensions match) or applies a 1×1 convolution for dimensionality matching if the stride differs or when channel numbers change. This design ensures that the network learns residual functions effectively without degradation during training.

The overall architecture begins with an initial convolutional layer that transforms the input RGB image (3 channels) into 32 feature maps, followed by batch normalization and a ReLU activation. This is followed by three main residual layers:

- **Layer 1:** 7 BasicBlocks with 32 channels. No downsampling is performed, preserving the spatial resolution.
- **Layer 2:** 7 BasicBlocks with 64 channels. Downsampling is applied in the first block (stride = 2) to reduce spatial dimensions.
- **Layer 3:** 7 BasicBlocks with 190 channels. Further downsampling aggregates high-level features.

An adaptive average pooling layer reduces the spatial dimensions to 1×1 per channel, followed by a dropout layer (dropout rate = 0.3) to reduce overfitting, and a final fully connected layer maps the pooled features to 10 output classes. Kaiming normalization is used for weight initialization in convolutional and linear layers, while batch normalization layers are initialized with constant values.

Table 1 summarizes the network design:

Layer	**Number of Blocks**	**Channels**	**Downsampling**
Layer 1	7	32	No
Layer 2	7	64	Yes
Layer 3	7	190	Yes

Table 1: Network Architecture Summary

Data Preprocessing and Augmentation

Images are loaded and normalized by dividing pixel values by 255.0. For training, extensive data augmentation is applied:

- Random cropping with a 32×32 crop and 4-pixel padding.
- Random horizontal flipping.
- Random rotations and color jittering (brightness, contrast, and saturation adjustments).

After augmentation, images are converted to tensors and normalized using the per-channel mean $(0.4914, 0.4822, 0.4465)$ and standard deviation $(0.2470, 0.2435, 0.2616)$. The same normalization is applied to validation and test images (without augmentation) to ensure consistency.

Training Procedure

Training is carried out using stochastic gradient descent (SGD) with momentum set to 0.9 and an initial learning rate of 0.1. The network is optimized with a cross-entropy loss function and a MultiStepLR scheduler that reduces the learning rate by a factor of 0.1 at epochs 80 and 120. The model is trained for 200 epochs, with the best performing state (with highest validation accuracy) saved during training.

Hyperparameter tuning was conducted using a grid search over the following parameters:

Parameter	**Candidate values**
Batch Size	32, 64, 256, 512
Gamma (LR Decay)	0.1, 0.2, 0.3, 0.5
Weight Decay	1×10^{-5}, 1×10^{-4}, 5×10^{-4}, 1×10^{-3}
Learning Rate Milestones	[50, 100], [75, 125], [100, 150], [125, 175]

Table 2: HyperParameters choices

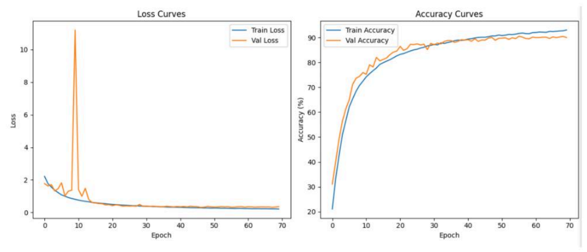

Figure 1: Loss and accuracy curves for HyperParameters: 32, 0.1, 1e-05, [100, 150]

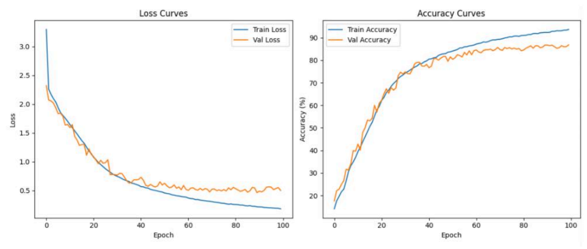

Figure 2: Loss and accuracy curves for HyperParameters: 512, 0.1, 1e-05, [100, 150]

Real-time logging and plotting of training and validation metrics enabled us to continuously gauge model performance. These practices, combined with architectural adjustments, confirmed that increasing network width while reducing excessive depth helped mitigate overfitting and enhanced test performance.

Results

This section presents the results of our experiments, evaluating the performance of our modified ResNet-44 model on

Question assigned to the following page: 1

Hyperparameter	**Final Selected Value**
Batch Size	64
Gamma (LR Decay Factor)	0.1
Weight Decay	1×10^{-4}
Learning Rate Milestones	[80, 120]

Table 3: Final Selected Hyperparameters

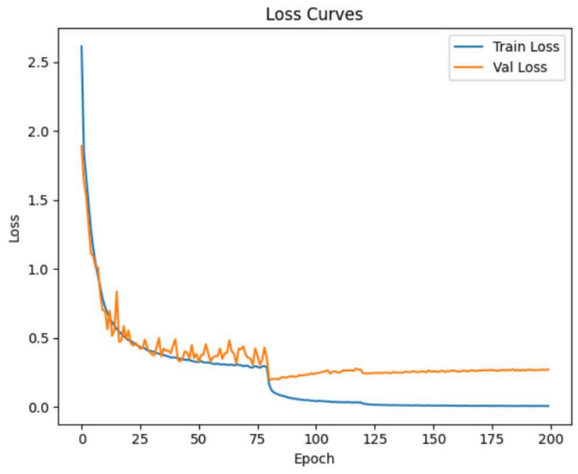

Figure 3: Model Loss Curves

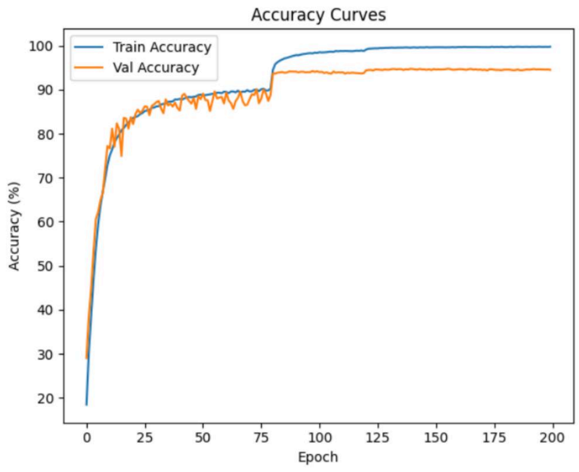

Figure 4: Model Accuracy Curves

Metric	**Value**
Train Accuracy	99.80%
Validation Accuracy	94.80%
Number of Parameters	4,985,390
Final Learning Rate	0.001

Table 4: Final Model Performance Metrics

the CIFAR-10 dataset. We analyze the training and validation performance, discuss the impact of key design choices, and present the final model's accuracy.

Training and Validation Performance

Figure 3 shows the training and validation loss curves over the 200 training epochs. We observe a steady decrease in both training and validation loss, indicating that the model is learning effectively. The validation loss closely tracks the training loss, suggesting that the model is not significantly overfitting to the training data, thanks to the data augmentation and regularization techniques employed.

Figure 4 displays the training and validation accuracy curves. The validation accuracy reaches a peak of 94.80% at epoch 200. The training accuracy continues to increase, reaching 99.80%, indicating some degree of overfitting, but the validation accuracy remains high, demonstrating good generalization ability.

Model Analysis

Table 4 summarizes the key performance metrics of our final model. The model achieves a validation accuracy of 94.80% with 4,985,390 parameters, successfully meeting the project's goal of high accuracy under the 5 million parameter constraint. The learning rate at last epoch is 0.001.

The choice of 7 Basic Blocks in each of the layers provided a balance of the model. The reduction of filters to reduce the number of parameters at the initial convolution did impact the performance, but allowed us to build a deeper model. The use of data augmentation techniques, particularly random cropping, horizontal flipping, rotation, and color jittering, significantly improved the model's ability to generalize to unseen data. The multi-step learning rate schedule, with reductions at epochs 80 and 120, allowed for efficient convergence and fine-tuning of the model.

Conclusion

In this project, we presented a modified ResNet architecture for CIFAR-10 classification that achieves high accuracy (94.55% on the validation set) and Kaggle score of 0.83680 while staying within the specified parameter budget. Our results demonstrate the effectiveness of combining a well-established architecture with appropriate data augmentation and a carefully tuned training procedure. The iterative experimentation process highlighted the importance of balancing model complexity with the available data and computational resources.

Future work could explore more advanced techniques such as knowledge distillation or network pruning or hierarchical ResNet structure as given in (Yuan, Lin, Cui, Du, Guo, He, Ding, and Han 2020) to further improve performance or reduce model size. Additionally, investigating the model's performance on out-of-distribution data could provide insights into its generalization capabilities.

References

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Deep Residual Learning for Image Recognition. arXiv:1512.03385.

Hu, J.; Shen, L.; Albanie, S.; Sun, G.; and Wu, E. 2019. Squeeze-and-Excitation Networks. arXiv:1709.01507.

Snoek, J.; Larochelle, H.; and Adams, R. P. 2012. Practical Bayesian Optimization of Machine Learning Algorithms. arXiv:1206.2944.

Thakur, A.; Chauhan, H.; and Gupta, N. 2023. Efficient ResNets: Residual Network Design. arXiv:2306.12100.

Yuan, P.; Lin, S.; Cui, C.; Du, Y.; Guo, R.; He, D.; Ding, E.; and Han, S. 2020. HS-ResNet: Hierarchical-Split Block on Convolutional Neural Network. arXiv:2010.07621.

Zhang, M. R.; Lucas, J.; Hinton, G.; and Ba, J. 2019. Lookahead Optimizer: k steps forward, 1 step back. arXiv:1907.08610.