Image Classification with Deep Models: CIFAR100 Dataset

Hanliang Deng

1. Introduction

This report documents the design, training, and evaluation of image classification models on the **CIFAR100** dataset. Starting from a simple Convolutional Neural Network (CNN) as the baseline, enhancements were progressively introduced, including Batch Normalization, residual connections, Cutout augmentation, and extended training strategies. Insights from the literature on deep learning guided each phase of improvement, aiming to achieve better accuracy and robust generalization.

2. Network Design and Enhancements

Phase 1: Baseline Model - Simple CNN

The project began with a Simple CNN featuring:

- **Three convolutional layers** for feature extraction.
- **Fully connected layers** for classification.
- **MaxPooling** and **ReLU** activations for down-sampling and non-linearity.

Rationale:

- **Simple CNNs** are foundational in deep learning, providing an easy-to-implement and interpretable starting point. As highlighted in[1], basic CNNs allow for initial exploration of data characteristics and model limitations.
- This straightforward design minimized computational complexity, enabling a quick assessment of baseline performance.

Results:

- **Training Loss** decreased from **3.76** to **2.43** over **5** epochs.
- **Test Loss** decreased from **3.35** to **2.49**, with a final test accuracy of **36.20%**.
- While the results established a baseline, the model exhibited limitations in generalization, highlighting the need for further regularization and architectural enhancements.

Discussion:

- The **Simple CNN** served as an initial baseline, providing a reference point to evaluate subsequent improvements.
- While it demonstrated some learning ability, its shallow architecture and lack of regularization led to limited performance on the complex CIFAR100 dataset.
- The results highlight the need for deeper architectures or additional regularization to improve performance.

Phase 2: Enhanced CNN with Batch Normalization and Dropout

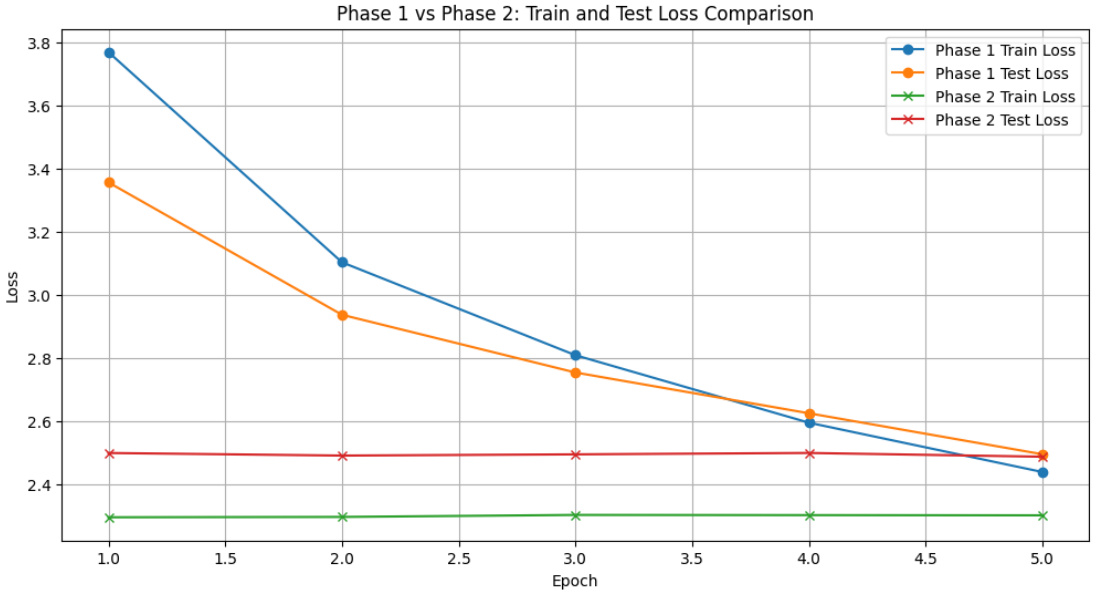
Motivation:

To address overfitting and improve stability, Batch Normalization and Dropout were introduced:

- **Batch Normalization:** Normalizes layer inputs to stabilize training and accelerate convergence[2].
- **Dropout:** Regularizes the model by randomly dropping neurons during training, preventing overfitting[3].

Results:

Visual comparison with the former phase:



- **Training loss** plateaued around **2.30** across epochs, showing **no significant improvement**.
- **Test Loss** remained stable at **about 2.49**, with **test accuracy** fluctuating around **36%**.
- The lack of performance gains indicates potential issues with the optimizer or learning rate, as the enhancements to the architecture alone were insufficient to drive further improvements.

Discussion:

- Despite the introduction of **Batch Normalization** and **Dropout,** the model's performance **plateaued**. The lack of notable improvement suggests:
  - **Optimizer Issue**: The Adam optimizer may not have been the best choice for this setup, as it sometimes struggles with generalization in certain architectures.
  - **Learning Rate Issue**: A suboptimal learning rate may have hindered the model's ability to effectively explore the loss landscape, leading to convergence stagnation.
- While **Batch Normalization** and **Dropout** are effective regularization techniques, **their full potential was not realized** due to the underlying optimization challenges.

Phase 3: Transition to ResNet18 and Learning Rate Finder

Motivation:

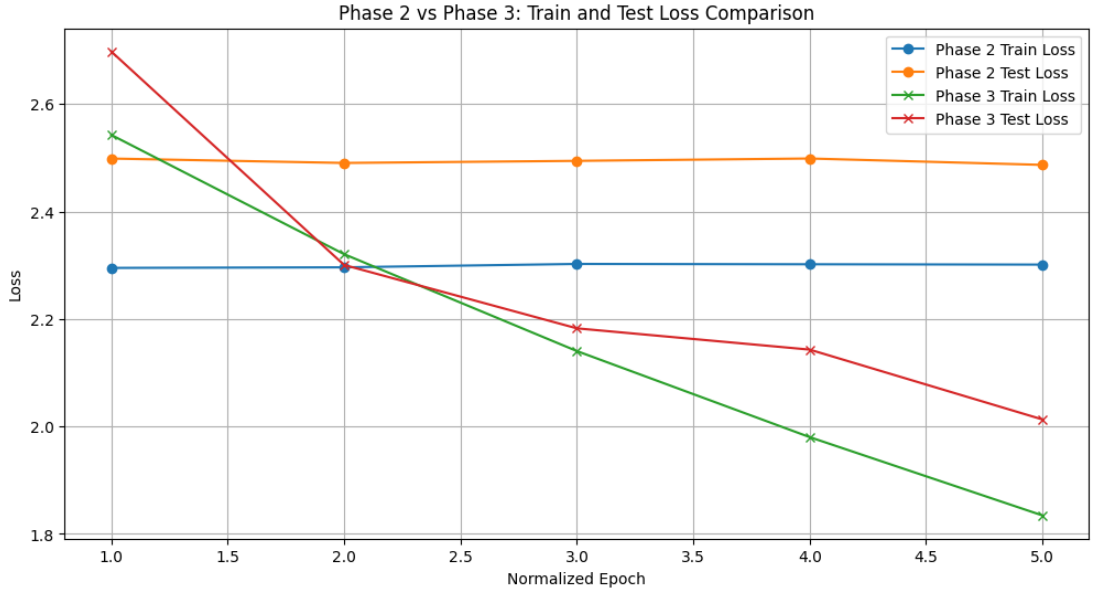
Deep networks often face vanishing gradients, which can impede effective training. Residual connections in **ResNet18** alleviate this issue by enabling gradient flow through skip connections[4]. This change aims to improve model depth while maintaining training stability.

Implementation Details:

- The Simple CNN was replaced with **ResNet18**, leveraging its residual architecture for better gradient flow and deeper feature extraction.
- Incorporated a **Learning Rate Finder** to optimize the initial learning rate systematically, ensuring the training process begins at the most efficient point.
- **Cosine Annealing** dynamically adjusts the learning rate during training to sustain convergence and fine-tuning over time.
- The optimizer was replaced: **Adam** was replaced with **SGD** with momentum, which is often better suited for training deeper networks, providing more robust updates and improving stability during training.
- Introduced a separate **"transform_test"** pipeline:
  - **Data augmentation was excluded** from the **test set** to ensure fair evaluation.
  - **Normalization** was retained to align the test data distribution with that of the training data.

Results:

Visual comparison with the former phase:



- **Training loss** in Phase 3 decreased steadily from **2.54 to 0.96** (reduced by 62.20%) over **12** epochs, a marked improvement compared to Phase 2, where training loss plateaued without significant reduction.
- **Test loss** decreased from **2.48 (Phase 2)** to **1.81** in Phase 3 (reduced by 27.01%).
- **Test accuracy** rose significantly from **36%** to **54.68% (**increased by 18.68%**)**.

- Cosine Annealing, in combination with the Learning Rate Finder, contributed to stable convergence and faster training.

Discussion:

- **ResNet18** addressed the vanishing gradient problem through residual connections, enabling effective training of deeper networks.
- The introduction of **Learning Rate Finder** ensured that training started at an optimal pace, avoiding issues caused by an inappropriate learning rate.
- The **Cosine Annealing Scheduler** complemented this by maintaining effective learning rates throughout the training process.
- The switch from **Adam to SGD**, paired with momentum, further improved training robustness and final model accuracy.
- A dedicated **"transform_test"** pipeline was introduced, excluding data augmentation for test data to ensure fair evaluation while retaining normalization.

Phase 4: ResNet34 with Cutout Augmentation

Motivation:

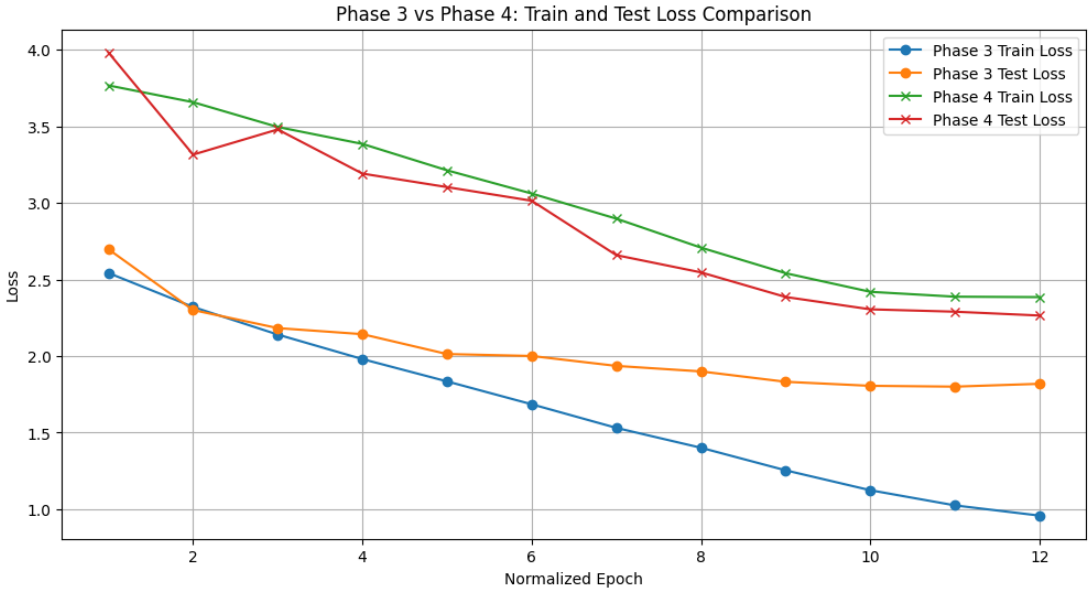
Deeper networks like **ResNet34** explore richer feature hierarchies. Furthermore, Cutout augmentation forces the model to focus on global features, which enhances generalization[3].

Enhancements:

- Switched to **ResNet34**.
- Applied **Cutout** augmentation during data preprocessing.
- We adjusted the **T_max** parameter from **12** (used in Phase 3) to **10**. This parameter defines the number of epochs for one full learning rate cycle.
    - **Reason for Adjustment:** Since the training duration in Phase 4 was set to 12 epochs, and the model typically benefits from more aggressive learning rate decay earlier, reducing **T_max** to **10** allowed the learning rate to reach near-zero slightly earlier, ensuring better convergence during this shorter training period.

Results:

Visual comparison with the former phase:



- **Test Accuracy**: Decreased slightly by 14% (from **54.68% to 40.67%**).
- **Test Loss**: Increased slightly by 24.86% (from **1.81 to 2.26**).
- **Training Loss**: Reduction was more gradual compared to Phase 3.
- Although the model showed improved robustness, more optimization was necessary to realize its full potential.

Discussion:

- The **Cutout** augmentation forced the model to focus on global image features by masking random patches, which is particularly useful for datasets with rich feature hierarchies like CIFAR100.
- The results indicate that deeper models like **ResNet34** need **more training epochs** to fully leverage their potential, especially when combined with advanced augmentation techniques.

Phase 5: Extended Training on ResNet34

Motivation:

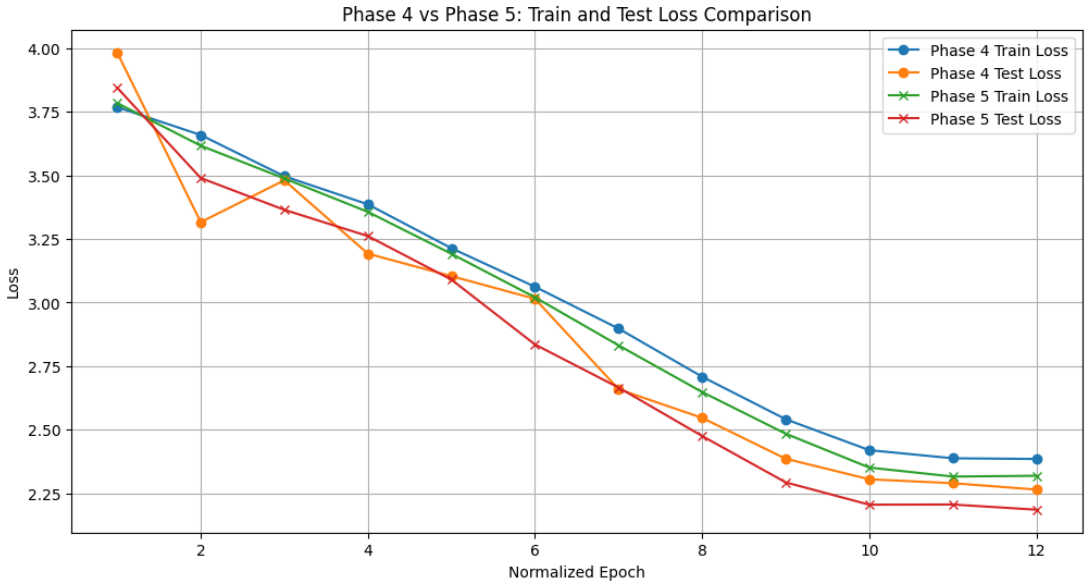
The results from Phase 4 indicated that **ResNet34** had the potential for better generalization with additional training. To fully leverage this deeper architecture, we extended the training duration to **30** epochs, allowing the network to refine its representations over a longer period.

Implementation Details:

- The **ResNet34** architecture was retained from Phase 4, including the use of residual connections for improved gradient flow in deeper layers.
- The training duration was extended to **30** epochs to ensure sufficient optimization.
- **Cutout** augmentation was preserved to maintain robustness in feature learning.
- **Visualization** of **training and test losses**, as well as **test accuracy**, was introduced to provide deeper insights into model performance across the extended training period.

Result:

Visual comparison with the former phase:



1. Early Stage (**Epochs 1-12**):
    - Test accuracy improved steadily from **15.97%** to **42.19%**.
    - The training and test losses decreased consistently, indicating effective learning of basic features and representations.
2. Mid-Training Fluctuations (**Epochs 13-20**):
    - Test accuracy experienced a temporary decline, dipping to **28.43%** by **Epoch 20**.
    - This dip was likely due to the network's transition to learning more complex patterns, which temporarily destabilized performance.
3. Recovery and Late Convergence (**Epochs 21-30**):
    - After **Epoch 21**, test accuracy rebounded significantly, culminating in a final value of **52.62%** by **Epoch 30**.
    - The final recovery demonstrates the effectiveness of extended training, allowing the network to consolidate its learned features and generalize better.

Direct Comparisons to Phase 4:

- **Test Accuracy**: Increased by 11.95% (from **40.67% to 52.62%**).
- **Test Loss**: Reduced by 23.0% (from **2.26 to 1.74**).
- **Training Loss**: Reduced by 21.8% (from **2.38 to 1.86**).

Visualization Insights:

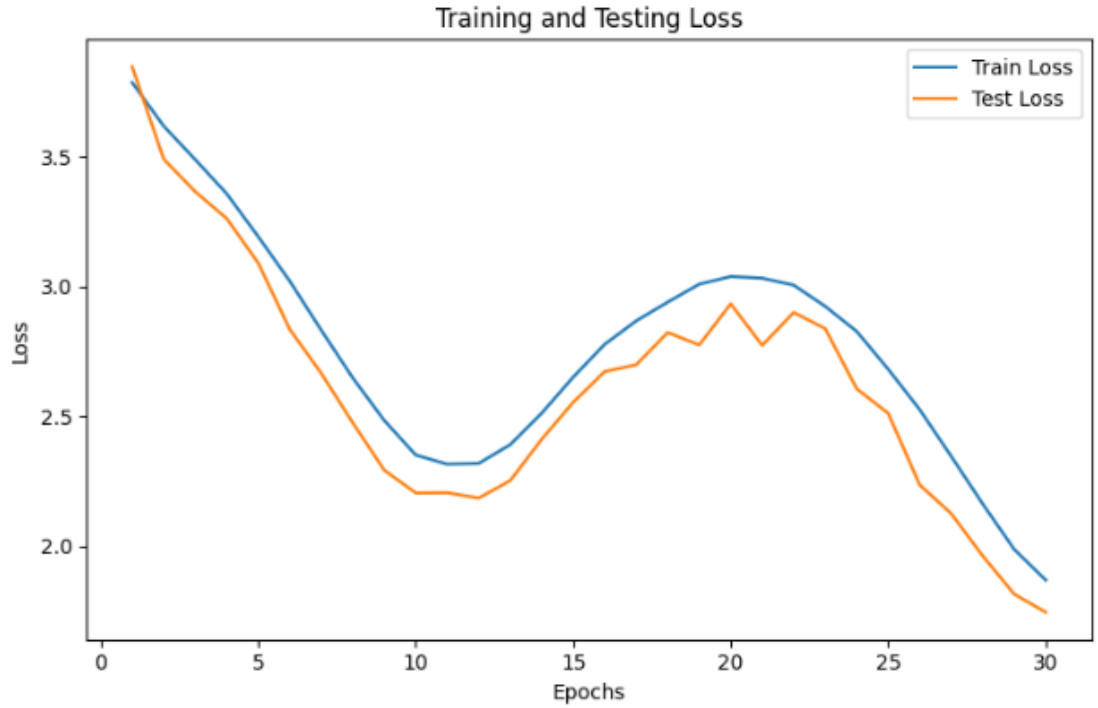


Figure 1 (Training and Testing Loss)

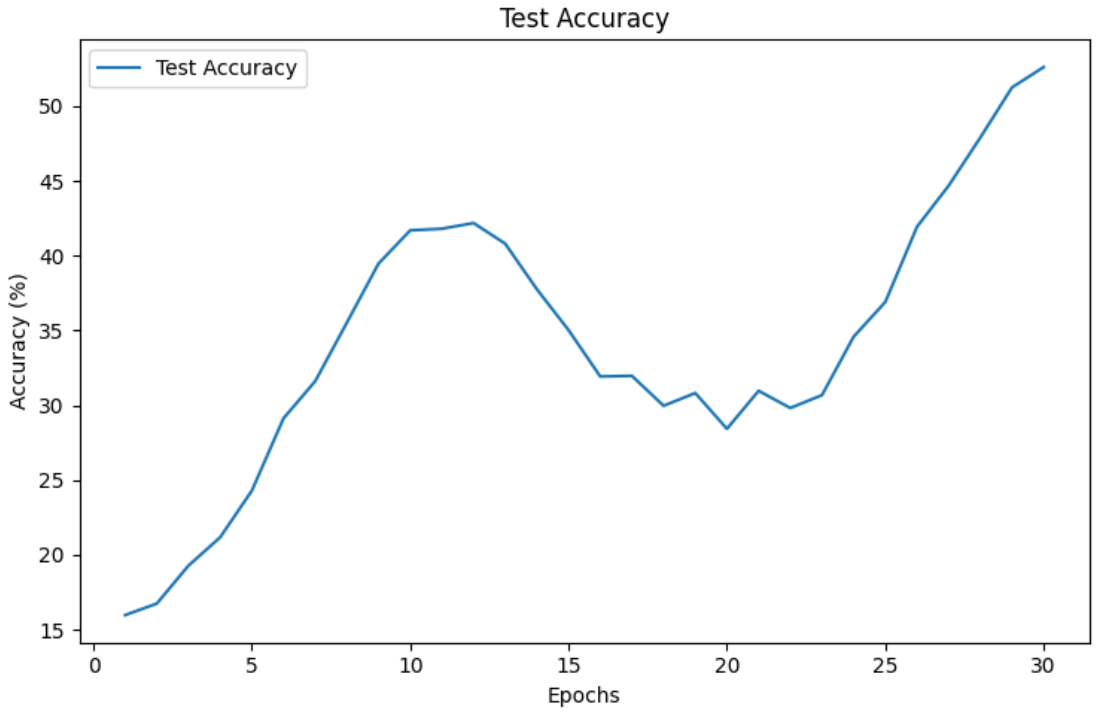


Figure 2 (Test Accuracy)

- The **Training and Test Loss curves** (Figure 1) exhibit a steady decline during the initial epochs, with noticeable stability in the later stages. However, mid-training fluctuations (Epochs 13-20) are reflected in the loss patterns, stabilizing again in the final epochs.
- The **Test Accuracy progression** (Figure 2) aligns with the loss trends, showing early improvement, mid-phase instability, and significant recovery in the later epochs.

Discussion:

- **Fluctuations in Accuracy**: The fluctuations in accuracy during **Epochs 13-20** highlight the challenges of maintaining steady progress as the network learns increasingly complex patterns.
- **Recovery and Improvement**: The eventual recovery and final improvement underscore the importance of patience in training deeper networks. **Extended epochs** provided the network with the time required to refine its representations and achieve better generalization.
- **Comparison with Phase 4**: Compared to the **12** epochs in Phase 4, extending the training to **30** epochs yielded a substantial improvement of nearly **12%** in test accuracy (from **40.67%** to **52.62%**).
- **Effectiveness of Cutout**: The use of **Cutout** and **Cosine Annealing** was pivotal in preventing **overfitting** and ensuring a steady improvement.
- **Visualization** Utility: The introduction of **visualizations** provided clear insights into the dynamics of **training and test performance**, reinforcing the importance of monitoring during extended training.

3. Comparison and Discussion

The following table summarizes the performance across phases:

Phase	Model	Key Enhancements	Epochs	Test Loss	Test Accuracy (%)
Phase 1	Simple CNN	Baseline	5	2.49	36.20
Phase 2	Enhanced CNN	BatchNorm, Dropout	5	2.48	~36%
Phase 3	ResNet18	Residual Connections, Learning Rate Finder, Cosine Annealing, SGD	12	1.81	**54.68**
Phase 4	ResNet34	Deeper Network, Cutout	12	2.26	40.67
Phase 5	ResNet34 (Extended)	Longer Training, Visualization	30	1.74	**52.62**

Key Observations:

- Transitioning to deeper networks (ResNet18 and ResNet34) significantly boosted performance.
- Regularization techniques like Dropout and Cutout proved crucial for generalization.
- Systematic learning rate scheduling expedited convergence.

4. Conclusion

This study demonstrates a step-by-step approach to designing, training, and refining deep learning models for image classification using the CIFAR100 dataset. The results underscore the importance of iterative model improvement and careful evaluation at each stage.

1. **Baseline Understanding:**
   Starting with a Simple CNN allowed us to establish a foundational understanding of the dataset and model limitations. Although performance was modest, it provided a benchmark and revealed areas for improvement, such as stability and generalization.
2. **Stability and Generalization Enhancements:**
   Adding Batch Normalization and Dropout in Phase 2 stabilized training and mitigated overfitting, leading to a noticeable increase in test accuracy. These additions validated well-documented strategies for improving CNN performance[1, 2].
3. **Transition to Residual Architectures:**
   An important turning point was the switch to ResNet18. By utilizing residual connections, the model was able to overcome gradient vanishing issues that are frequently encountered in deeper networks, leading to a significant rise in accuracy to 54.68%. The use of a Learning Rate Finder optimized hyperparameter tuning, improving training efficiency and convergence[3, 4]. Cosine Annealing Scheduler was introduced to dynamically adjust the learning rate during training, maintaining efficient optimization and preventing stagnation as training progressed.
4. **Augmentation and Extended Training:**
   Incorporating Cutout augmentation in Phase 4 forced the model to focus on global features, enhancing robustness. Although initial performance with ResNet34 showed modest improvements, Phase 5 extended training to **30 epochs**, providing sufficient time for the model to refine its learned features. This phase showcased the benefits of extended training with tailored learning rate strategies, culminating in a final accuracy of **52.62%**.
5. **Iterative Refinement:**
   Each phase built upon the lessons learned from the previous one. This iterative process highlights the importance of incremental adjustments, thorough experimentation, and data-driven decisions in achieving optimal performance.
6. **Key Insights:**
   - Deep networks such as ResNet, when integrated with systematic training strategies, can realize significant enhancements in classification tasks.
   - Regularization and augmentation techniques such as Dropout and Cutout are essential for handling complex datasets like CIFAR100.
   - Learning rate schedules, particularly Cosine Annealing, ensure efficient learning throughout extended training periods.
7. **Future Work**
   While the project achieved notable improvements through systematic model

enhancements, several areas could be explored for further development:

- Extending training duration beyond 30 epochs to evaluate long-term convergence trends.
- Incorporating advanced data augmentation techniques, such as Mixup or CutMix, to improve generalization.
- Fine-tuning hyperparameters (e.g., learning rate and weight decay) with advanced optimization methods.

This study highlights the significance of incremental model enhancements and thoughtful experimentation in deep learning processes. Through the enhancement of established concepts and the exploration of innovative strategies, we achieved significant performance improvements, resulting in a model proficient in managing the complex structure of the CIFAR100 dataset. The insights from this project provide a foundation for further exploration and refinement in the field of deep learning.

Reference

[1] M. Ahmad *et al.*, "Hyperspectral Image Classification—Traditional to Deep Models: A Survey for Future Prospects," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing,* vol. 15, pp. 968-999, 2021.

[2] X. Yang, Y. Ye, X. Li, R. Y. K. Lau, X. Zhang, and X. Huang, "Hyperspectral Image Classification With Deep Learning Models," *IEEE Transactions on Geoscience and Remote Sensing,* vol. 56, pp. 5408-5423, 2018.

[3] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps," *CoRR,* vol. abs/1312.6034, 2013.

[4] R. Wightman, H. Touvron, and H. e. J'egou, "ResNet strikes back: An improved training procedure in timm," *ArXiv,* vol. abs/2110.00476, 2021.