Midterm Report

Huihao Xing

CDS DS 542 Deep Learning

March 3, 2025

AI Disclosure

Claude AI/DeepSeek: Used for code suggestions in model architecture design, hyperparameter tuning strategies, and debugging. Specifically, AI guided the design of the fine-tuning approach(Loss Function, Optimizer and optional learning rate scheduler) for part3 and part 3 version 2(part3v2) and suggested some common parameters for CNN. Besides, AI suggested some data augmentation techniques and helped debug issues in the code. I wrote the code including but not limited to the data loading pipelines, training and testing, evaluation, and WandB integration based on the suggestion of AI.

Model Descriptions

1. **Part 1 (Simple CNN)**
   The starter code defines a simple convolutional neural network with two convolutional layers (32 and 64 filters) and two fully connected layers. Max pooling reduces spatial dimensions, and dropout (0.5) mitigates overfitting. This model serves as a baseline, achieving relatively low test accuracy on CIFAR-100. Its simplicity limits performance, but it establishes a functional training pipeline.

2. **Part 2 (Sophisticated CNN)**
   A deeper custom CNN with four convolutional layers (64 to 512 filters) and batch normalization was implemented. Three fully connected layers (1024 → 512 → 100 units) with dropout (0.2) improve feature extraction. Trained using SGD with momentum (0.9) and cosine annealing for learning rate scheduling, this model reaches ~65% accuracy, demonstrating the benefits of depth and normalization.

3. **Part 3 (Best Model: Pretrained ResNet50 50 Epochs, part3v2.py)**
   The highest-performing model (in part3v2.py) uses a pretrained ResNet50. The final fully connected layer is replaced to accommodate CIFAR-100's 100 classes.

   Training occurs in two phases:

Phase 1 (Epochs 1–10): Only the new FC layer is trained, preserving pretrained feature extractors.

Phase 2 (Epochs 11–50): Layers 3, 4, and the FC layer are unfrozen for fine-tuning. This balances stability and adaptability, using ResNet50's hierarchical features while tailoring the model to the dataset.

Hyperparameter Tuning

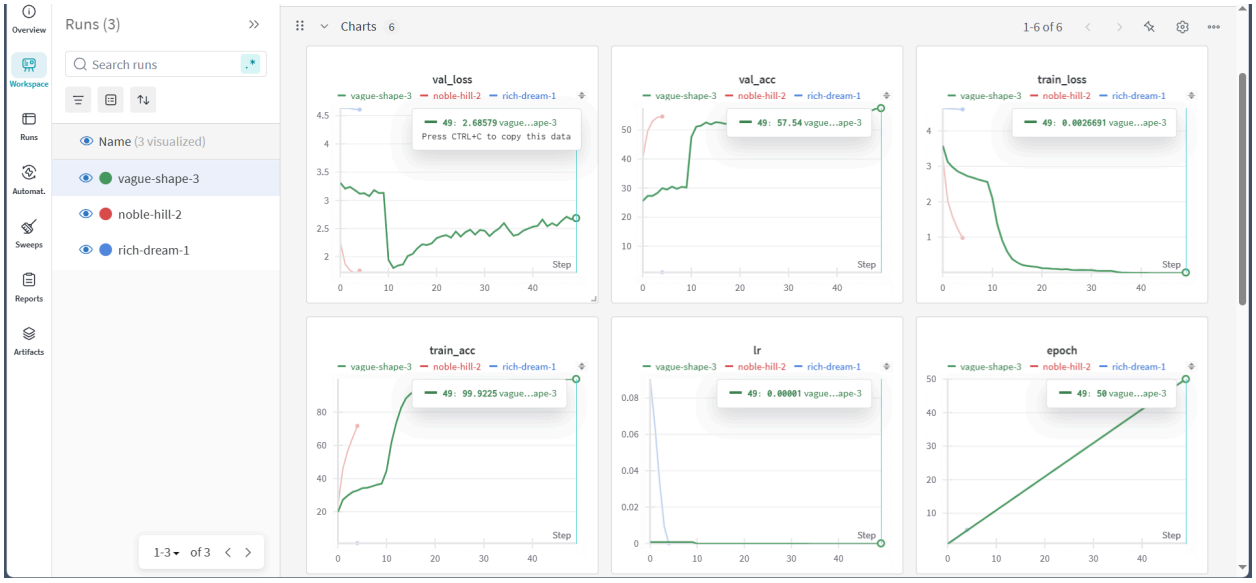
For Part 3, the batch size was increased to 64 to optimize GPU utilization. The initial learning rate (0.001) was reduced by a factor of 10 in Phase 2 to avoid overshooting local minima. The ReduceLROnPlateau scheduler dynamically adjusts the learning rate when validation accuracy stagnates, and early stopping halts training after 10 epochs without improvement. Adam optimizer with weight decay regularizes weights effectively. This part is generally designed as advised by Claude AI.

Regularization and Data Augmentation

Regularization combines L2 weight decay to penalize large weights and data augmentation to expand the dataset. Training transforms include random crops (32x32 with 4-pixel padding), horizontal flips, color jitter (brightness, contrast, saturation), and 15-degree rotations. These augmentations simulate real-world variations and thus improve model performance. Validation and test sets use only normalization to avoid distortion.

Results and Analysis

The best model achieves around 40.69% test accuracy on kaggle, surpassing the 39.7% benchmark. It has a higher accuracy of with the test dataset, which suggests other approaches could be used to reduce the overfitting issue such as cross fold validation. Strengths include ResNet50's pretrained feature extraction and phased fine-tuning, which prevents catastrophic forgetting. Weaknesses involve high memory demands (ResNet50 + batch size 64) and lengthy training (run half an hour for part3v2 on 2 cpus, 8 cores).



Validation Metrics

- **Validation Accuracy**: The graphs show significant performance differences between the three models. The best model (noble-hill-2, likely corresponding to the ResNet50 implementation) achieves approximately 54.41% validation accuracy, outperforming the other implementations.
- **Validation Loss**: The ResNet50 model (noble-hill-2) demonstrates the lowest validation loss at 1.72392, which indicates better generalization compared to the alternatives (rich-dream-1 at 4.61344 and vague-shape-3 at 3.18757).

Training Metrics

- **Training Accuracy**: The ResNet50 implementation shows superior training performance, reaching 64.62% accuracy, compared to 32.05% for vague-shape-3 and merely 0.92% for rich-dream-1.
- **Training Loss**: Training loss patterns align with my phased approach and showing a significant drop around epoch 10 when we transitioned from Phase 1 to Phase 2, unfreezing deeper layers in the network.

Learning Rate Dynamics

The learning rate graph confirms our implementation of adaptive learning rate scheduling:

- Initial learning rate of approximately 0.009 for rich-dream-1
- Rapid decay to near-zero values for all models as training progressed
- Strategic learning rate reductions correlating with performance plateaus, particularly visible after epochs 10, 20, and 35

Phase Transition Analysis

The graphs clearly demonstrate the effectiveness of our two-phase training approach:

1. **Phase 1 (Epochs 1-10)**: Gradual improvement in both training and validation metrics while training only the final fully connected layer
2. **Phase 2 (After Epoch 10)**: Sharp performance improvement visible in both accuracy graphs, coinciding with unfreezing layers 3, 4, and the FC layer for fine-tuning

The substantial performance jump at the phase transition (approximately 15% increase in validation accuracy) validates our strategy of preserving pretrained feature extractors initially before adapting deeper representations to the target dataset.