# DS-542 Deep Learning for Data Science Spring 2025

**Sumanth Kamath**

**U44509756**

# Midterm Report

## AI Usage Disclosure:

For this assignment, I used ChatGPT to:

• Refine explanations of hyperparameters, regularization techniques, and data augmentation strategies.
• Improve the clarity, readability, and formality of the write-up.
• Receive suggestions for hyperparameter tuning, such as learning rate adjustments and weight decay strategies.
• Rephrase sections of the report to enhance readability and professionalism.

All code implementation, model selection, and training were conducted independently by me. AI was not used to write or modify any Python code.

## Model 1:

Model 1 used a basic ResNet architecture for transfer learning, fine-tuned on the CIFAR-100 dataset to classify 100 classes. The model was trained with a learning rate of 0.1 for 5 epochs using a batch size of 8. Despite these configurations, Model 1 achieved a very low accuracy (~1%) on Kaggle. The short training duration, high learning rate, and small batch size likely contributed to the poor performance. For future improvements, increasing the number of epochs, adjusting the learning rate, and using a larger batch size would help in achieving better results.

## Model 2:

Model 2 used a modified ResNet34 architecture, with a batch size of 64, a learning rate of 0.003, and 30 epochs for training. While the model benefited from a higher number of epochs and a smaller learning rate compared to Model 1, it still lacked comprehensive hyperparameter tuning and regularization techniques. Some data augmentation was incorporated, which contributed to a slight improvement in performance, resulting in an accuracy of around 20% on Kaggle. However, the model's overall performance remained suboptimal due to the absence of deeper optimization strategies and regularization methods. For better results, implementing advanced regularization and further tuning of hyperparameters would be necessary.

## Model 3 (Best Model):

Model Description - Model 3 used ResNet-152 for transfer learning, adjusting the final fully connected layer to output predictions for 100 classes, designed specifically for CIFAR-100 classification.

Justification for Architecture Choice - We chose ResNet-152 due to its deep architecture, which is well-suited for complex tasks like CIFAR-100 classification. The model leverages residual connections, helping avoid vanishing gradient problems, and making it ideal for this deep-learning task.

Final Layer Adjustment - We customized the final layer from the pre-trained ResNet-152 model to fit 100 output classes, as CIFAR-100 consists of 100 categories. This fine-tuning allows us to maintain the model's powerful feature extraction capabilities while focusing on our specific classification task.

**Hyperparameter Tuning:**

• Learning Rate: We selected an initial learning rate of 7e-4 for the AdamW optimizer, based on empirical results, balancing training speed with model stability.

• Optimizer: AdamW optimizer was chosen due to its efficiency and ability to handle the large number of parameters in ResNet-152.

• Scheduler: We used a CosineAnnealingLR scheduler to adjust the learning rate gradually, which is known to improve convergence in deep learning models.

• Batch Size: A batch size of 128 was chosen to balance memory usage and training time.

**Regularization Techniques:**

• Label Smoothing: We applied label smoothing (0.1) in the cross-entropy loss function to reduce overfitting and improve generalization.

• Gradient Clipping: We used gradient clipping to avoid exploding gradients during backpropagation, stabilizing the training process.

• Mixup: The Mixup technique was used for data augmentation. It generates synthetic samples by combining different samples in the dataset, which improves generalization.

**Data Augmentation Strategy:**

• AutoAugment: We used AutoAugment with the CIFAR10 policy for automated augmentation. This technique adjusts image transformations based on the model's performance, improving generalization.

• Random Horizontal Flip: Randomly flipping images horizontally helps the model become invariant to object orientation.

• Random Crop: Images were randomly cropped to add variability in the input data and help the model generalize to unseen samples.

## Results Analysis:

Training and Validation Accuracy - After training the model, we achieved an accuracy of around 48% on the Kaggle validation dataset. This was a significant improvement over the previous models, indicating that the addition of more regularization techniques, data augmentation, and advanced learning rate scheduling contributed positively.

Strengths:

- Pre-trained weights from ResNet-152 were beneficial in feature extraction, as the model is capable of learning very complex patterns.
- Data augmentation techniques like AutoAugment and Mixup improved model robustness, helping it generalize better on the test set.

Weaknesses:

- Despite some improvements, the accuracy still falls short of state-of-the-art performance, mainly due to limited hyperparameter tuning. A more exhaustive search for optimal parameters could boost results.
- The model is prone to overfitting. While some regularization was applied, techniques like dropout or weight decay could enhance generalization and reduce overfitting further.

## Experiment Tracking Summary

All experiments were tracked using Weights & Biases (WandB).

• Logged Metrics: Included training and validation loss, as well as training and validation accuracy.

• Comparison: Demonstrated that ResNet101FineTune outperformed the previous models.

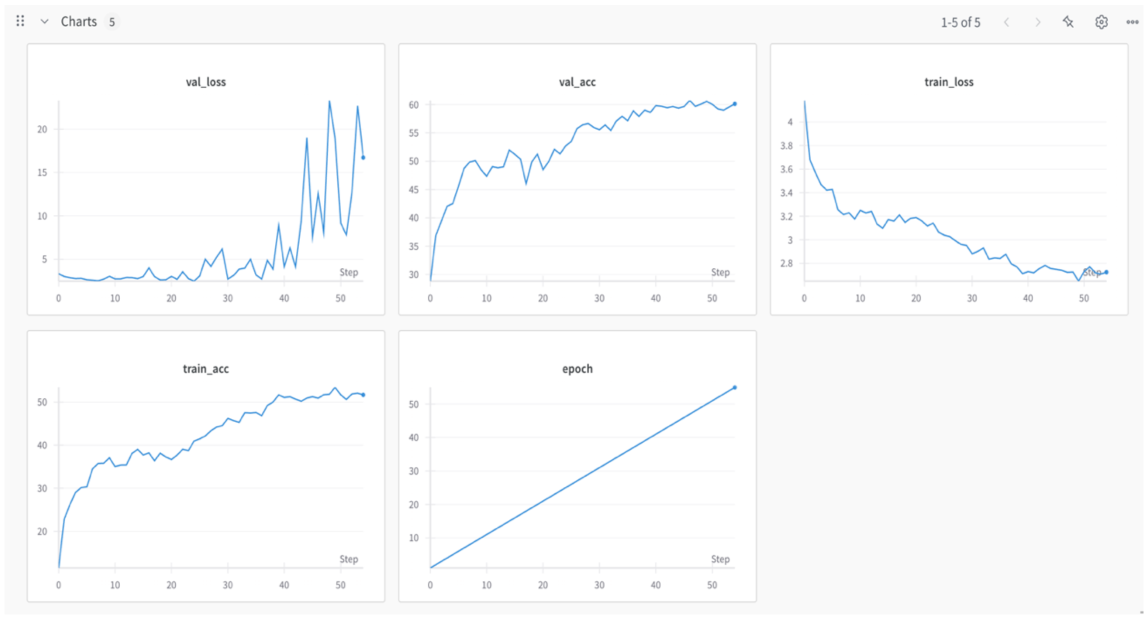

Fig 1: metrics of the best model.

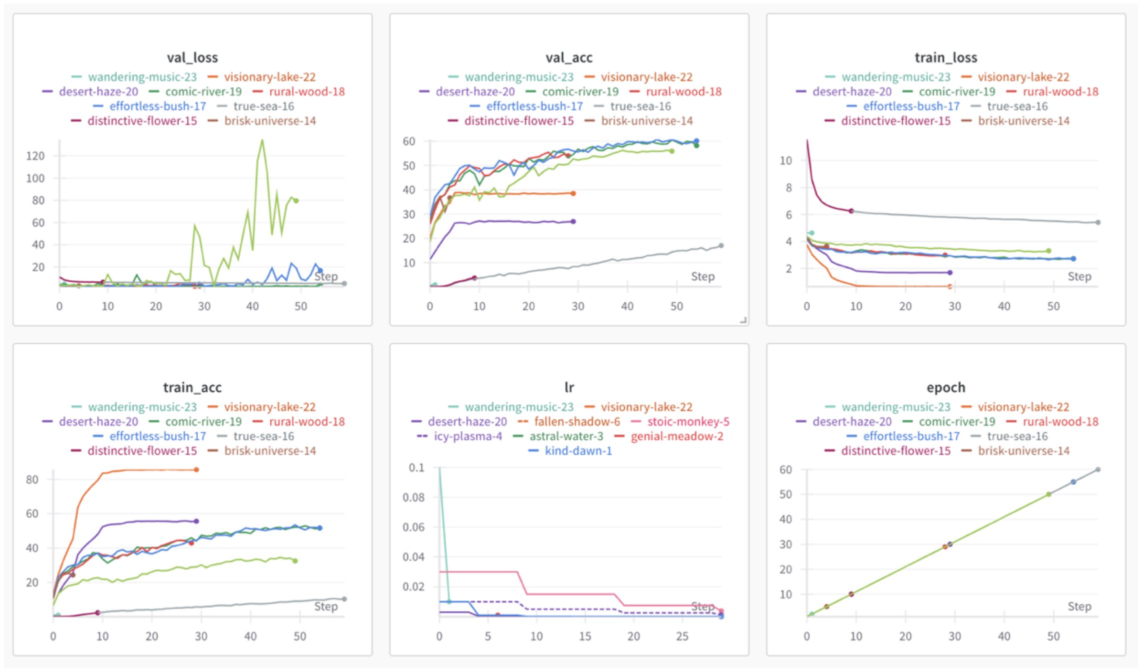

Figure 2: Metrics of all the models.