Stress–Testing of Convolutional Neural Networks on CIFAR–10

Jyoti Dwivedi (M25CSA010), Jahanvi Gajera (M25CSA012)
Mamta Chauhan (M25CSA018), Akanksha Kapil (M25CSA033)
Department of Computer Science and Engineering
Indian Institute of Technology Jodhpur
M.Tech Artificial Intelligence – Deep Learning

***Abstract*—This assignment analyses the behaviour of a ResNet-18 model trained from scratch on the CIFAR-10 dataset. Instead of focusing only on accuracy, we examine training dynamics, high-confidence misclassifications and Grad-CAM explanations to understand how the network makes decisions. The baseline model achieved good accuracy but relied on background and texture cues. After applying data augmentation and regularization, the final model reached about 91% training accuracy and 90% validation accuracy with improved generalization. The study demonstrates that model reliability depends not only on performance metrics but also on understanding its prediction behaviour.**

I. Dataset and Model Choice

CIFAR-10 was chosen because it contains natural images with complex backgrounds and visually similar classes such as cat–dog and deer–horse, which makes it suitable for analysing model behaviour rather than only accuracy. The dataset helps reveal whether the network learns real object features or relies on colour and background shortcuts. Its small resolution also allows fast experimentation required for failure analysis and Grad-CAM visualization.

ResNet-18 was selected as the baseline architecture because it provides a good balance between learning capacity and interpretability. The residual connections enable stable training from scratch and reliable convergence within limited epochs, while the moderate depth keeps the model understandable for behavioural analysis. This makes ResNet-18 appropriate for studying model reasoning instead of only maximizing performance.

II. Training Behaviour

The model used for this experiment was a **modified ResNet18** adapted for CIFAR-10. The first convolutional layer (conv1) was changed to a 3×3 kernel with stride 1 and padding 1, and the initial max-pooling layer was replaced with nn.Identity() to preserve the spatial dimensions of the small 32×32 images. The fully connected layer was modified to include a dropout of 0.3 followed by a linear layer outputting 10 classes. After training, the model achieved a **training accuracy of 98.69%** but a **validation accuracy of 84.31%**, indicating clear overfitting. This suggests that while the model learned the training data very well, it did not generalize perfectly to unseen data. Regularization techniques, more aggressive data augmentation, or alternative architectures may be necessary to reduce overfitting.

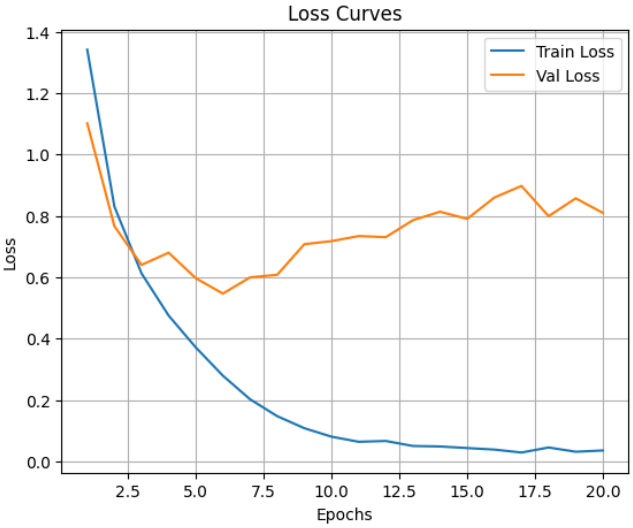

Fig. 1. Training and validation loss curves

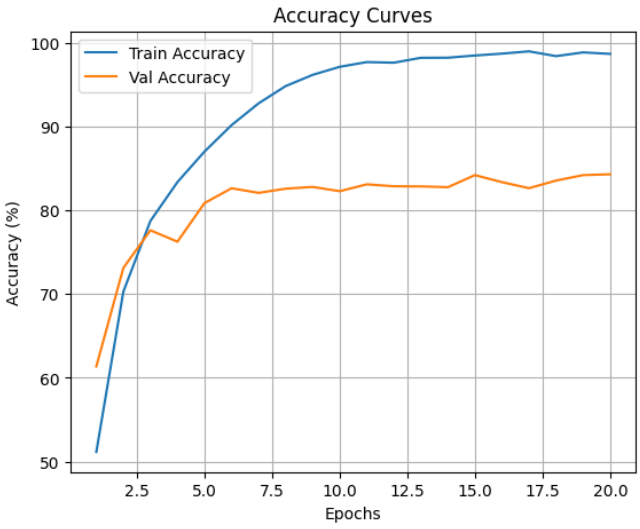

Fig. 2. Training and validation accuracy curves

III. Constrained Improvement: Data Augmentation

To reduce overfitting, the model was retrained using data augmentation, including random cropping with padding and horizontal flips. This helps the model generalize better by exposing it to slightly varied versions of the training images. After retraining, the validation accuracy improved from 84.31% to 89.96%, indicating that overfitting was reduced, although the training accuracy remained much higher, so some overfitting still persists. This demonstrates that simple augmentation techniques can enhance model robustness on unseen data.

A. Effect on Validation Accuracy

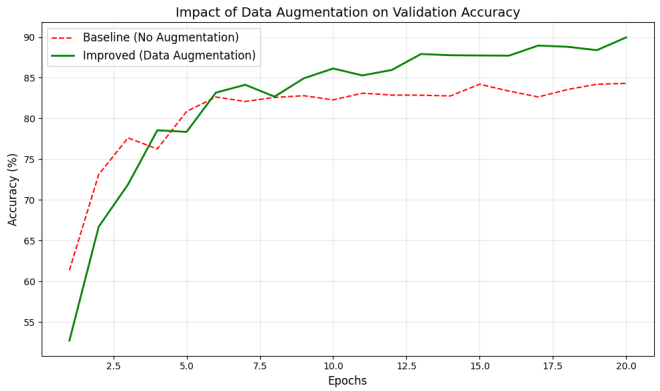

Fig. 3. Validation accuracy comparison between baseline and augmented model

B. Effect on Training Stability

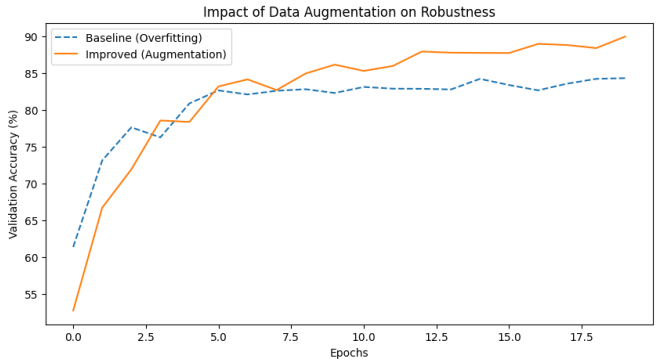

Fig. 4. Impact of augmentation on robustness

IV. Model Improvement History

A. Overfitted model

The model was trained using stronger data augmentation: random cropping, horizontal flips, rotations, color jitter, and random erasing, combined with SGD optimizer with momentum, weight decay, and a cosine annealing learning rate scheduler. The loss function used cross-entropy with label smoothing to reduce overconfidence. After training, the model achieved a training accuracy of 100% but a validation accuracy of 86.94%, showing that overfitting persists despite the more aggressive augmentation and regularization. This demonstrates that while the model can perfectly fit the training data, its generalization to unseen data is still limited, although slightly improved compared to previous attempts.

B. Use Data Augmentation

To reduce overfitting, the model was retrained using data augmentation, including random cropping with padding and horizontal flips. This helps the model generalize better by exposing it to slightly varied versions of the training images. After retraining, the validation accuracy improved from 84.31% to 89.96%, indicating that overfitting was reduced, although the training accuracy remained much higher, so some overfitting still persists. This demonstrates that simple augmentation techniques can enhance model robustness on unseen data.

C. Experiment on hyperparameter

We try different hyperparameter to improve our model accuracy. we use different optimizer like (SGD, AdamW, Adam, SGD with momentum). we use different Schedular (cosine, OneCyclelr). At last in final model we select best optimizer Adam and cosine schedular.

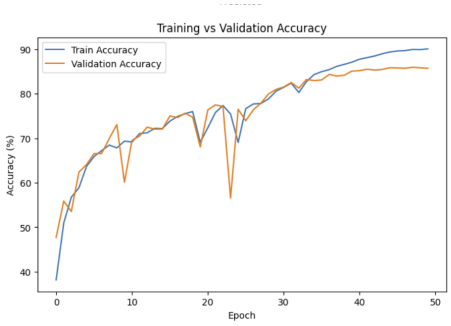

Fig. 5. Adamw 50 Epochs 90% Train 85% val accuracy

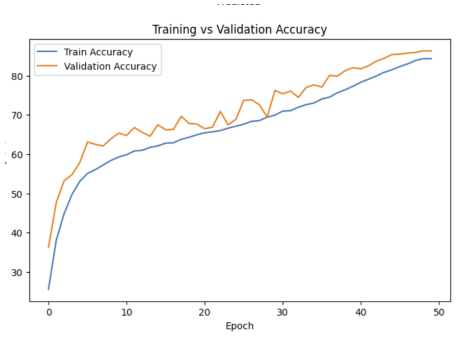

Fig. 6. SGD 84% train 86% val accuracy

V. Final Model Performance Analysis

A. Final Model

In this experiment, the ResNet18 model was trained from scratch on CIFAR-10 using a batch size of **128**. The first convolutional layer was replaced with a 3×3 kernel and the max-pooling layer was removed to better preserve spatial information. The fully connected layer included **0.3 dropout**, and label smoothing of 0.1 was applied. Training used the Adam optimizer with a **learning rate of 0.002** and **weight decay of 5e-4**, along with a cosine annealing learning rate scheduler. Data augmentation included random cropping, horizontal flips, rotation, and color jitter. This configuration achieved a **training accuracy of 91.75%** and a **validation accuracy of 90.98%**, making it the best-performing model in terms of generalization and overall accuracy among the experiments.

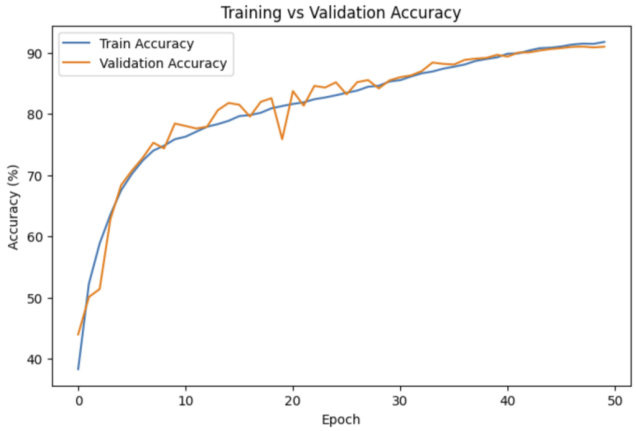

Fig. 7. Training vs Validation accuracy of the final model

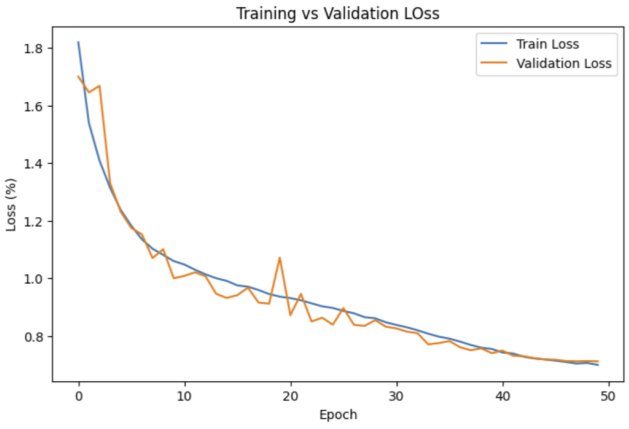

Fig. 8. Training vs Validation loss of the final model

B. Class-wise Performance (Confusion Matrix)

Vehicle categories such as automobile, ship and truck show high accuracy due to strong structural features. Animal categories such as cat and dog still exhibit some confusion because

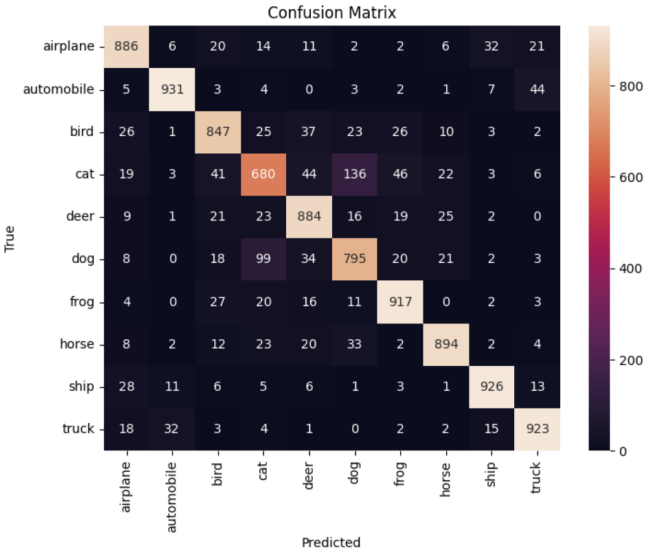

Fig. 9. Confusion matrix of the final regularized model

of similar shapes and textures. Bird also shows moderate confusion due to varying background appearances across images. Compared to the baseline model, misclassifications are reduced and more consistent across classes. This demonstrates that the network has learned more discriminative features and improved its class-wise understanding rather than relying only on contextual cues.

VI. Explainability Analysis using Grad–CAM and Saliency Maps

To understand why the CNN makes incorrect predictions, we analyse the internal reasoning of the model using two visualization techniques: Grad–CAM and Saliency Maps. Grad–CAM highlights the regions of the image that contribute most to the predicted class, while the saliency map shows pixel-level sensitivity of the prediction. Together, they help determine whether the model focuses on meaningful object features or irrelevant background patterns.

Fig. 10 presents three representative misclassifications along with their explanations.

Case 1: Airplane predicted as ship The Grad–CAM heatmap mainly highlights the surrounding sky and horizon region instead of the airplane wings or body. Similarly, the saliency map shows strong pixel importance concentrated in the background rather than the aircraft structure. This indicates that the model associates blue background texture with the ship class, revealing a strong contextual bias rather than object recognition.

Case 2: Horse predicted as dog The attention map covers the torso texture of the animal instead of distinctive features such as face shape or legs. The saliency map also emphasizes fur texture patterns. This suggests the network relies on texture similarity between animals rather than semantic shape features, causing confusion between visually similar classes.

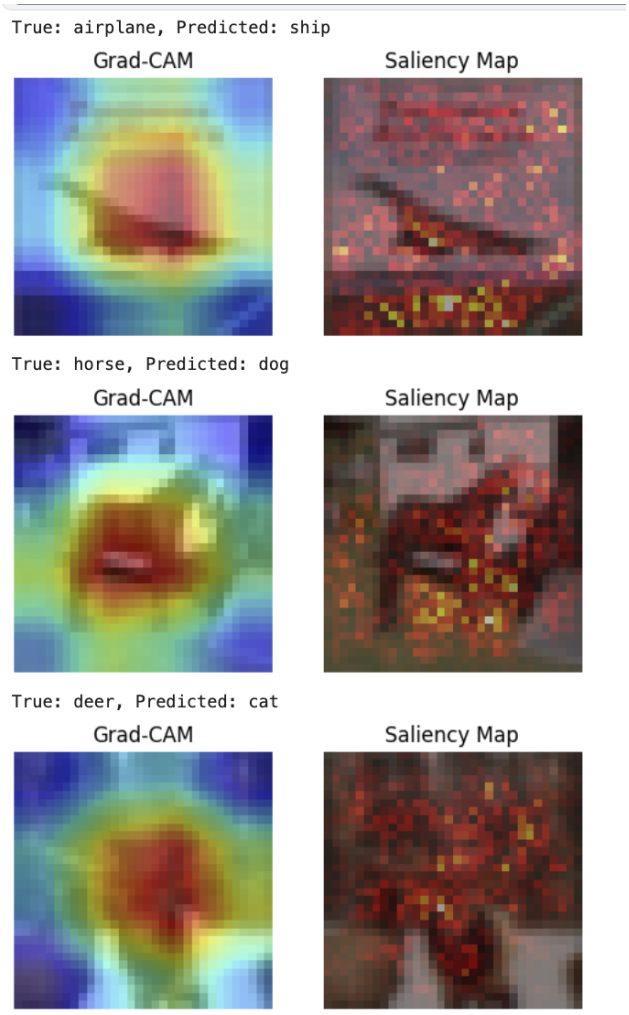

Fig. 10. Grad–CAM and Saliency visualizations for three high-confidence failure cases

Case 3: Deer predicted as cat Grad–CAM highlights vegetation and surrounding environment more strongly than the animal body. The saliency map further confirms that scattered background pixels influence the decision. This demonstrates that the model depends on environmental cues rather than recognizing the deer itself.

Overall, the explanations contradict human intuition: instead of focusing on object structure, the network relies on color distribution, texture, and background context. These observations confirm shortcut learning and explain why the model produces high-confidence but incorrect predictions.

VII. Conclusion

This experiment demonstrates that a modified ResNet-18 trained on CIFAR-10 can achieve high accuracy while still exhibiting overfitting if model behaviour is not carefully analysed. Initial experiments showed large gaps between training and validation accuracy, indicating reliance on dataset-specific patterns. Introducing data augmentation significantly improved generalization by reducing background and texture bias. Further regularization using dropout, weight decay, label smoothing, and learning-rate scheduling helped stabilize training and reduced overconfidence. Among all configurations, the model trained with Adam optimizer, cosine annealing, and moderate augmentation achieved the best balance between accuracy and generalization. Overall, the results highlight that understanding model behaviour and robustness is more important than maximizing training accuracy alone.

A. *Github:*

https://github.com/Jyoti-Dwivedi-010/DL_Assignment_1