Assignment – 1

Stress-Testing of Convolutional Neural Networks

Submitted By-

Satendra Singh(M25AIR006)

Shashank Kumar Gautam(M25AIR007)

Shivansh Yadav(M25AIR008)

Sunil Pargi(M25AIR010)

1.Introduction - The goal of this is to critically examine the behaviour, strength, and failure of Convolutional Neural Network (CNN) that has been trained on the CIFAR-10 dataset. Although high classification accuracy is a primary measure, this does not just examine the standard performance evaluation but delves into the reasons that the model malfunctions and how confident it is to make its erroneous predictions. We also use a ResNet-18 architecture, modified and implemented in PyTorch, to create a baseline and introduce a failure analysis to it.

We are going to use a baseline model that is trained without the process of pre-training as we are using firm architectural constraints. We have selected ResNet-18 architecture in particular because it is a compromise between depth and computational efficiency. Nonetheless, due to the original design of ResNet to high-resolution ImageNet data, we made certain architectural adjustments to make it fit into the low-spatial-dimension of CIFAR-10 data. This report describes our methodology and analyses the training dynamics, visualizes particular cases of our failure when the model is confidently wrong, and reflects on the implications of our results on the real-world trust.

2. Methodology –

2.1 Dataset Preparation-

CIFAR-10 dataset is comprised of 60,000 color images that have been categorized into 10 classes, which are airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck.

We adhered strictly to the official split

Training Set: 50,000 images.

Test Set: 10,000 images.

Deep neural networks are also sensitive to the magnitude of input data. We converted the images to PyTorch tensors (scaling pixel values to the [0, 1] range) and standard normalized them. We used the mean (0.4914, 0.4822, 0.4465) and standard deviation (0.2023, 0.1994, 0.2010) as the means and the standard deviation of the RGB channels respectively. This concentrates the data which makes converging in the optimization faster.

2.2 Model Architecture: Modified ResNet-18

We used the ResNet-18 architecture which incorporated residual connections (skip connections) to help in overcoming the vanishing gradient issue with deep networks. Traditional Resnet-18 models (trained on 224x224 images) employ a violent internal downsampling technique, employing a 7x7 stride-2 convolution layer, and an directly succeeding MaxPool. When applied to CIFAR-10 (32x32), this will decrease the number of features present at the 8x8 block before the residual block, which will then cause valuable spatial features to be lost.

In order to overcome this, we varied the input layers as follows:

Input Convolution - The original 7*7 convolution (stride 2) was substituted with a 3*3 convolution (stride 1). This maintains the full 32*32 spatial resolution that it uses in the original residual block.

Pooling Removal - The first MaxPooling was dropped and instead an Identity mapping was applied, which once again avoided information loss at an early stage.

Classification Head - The last fully connected layer was changed to produce 10 logits instead of 1000.

It is a 4-block residual layers model (with two convolutional layers on each block with skip connections), which is then followed by average pooling and the linear classifier.

2.3 Training Configuration

To make repeated experiments reproducible, a set random number (SEED=42) was always set in PyTorch and NumPy for all stochastic operations.

Optimizer- Stochastic Gradient Descent (SGD)

Momentum - 0.9

Weight Decay**:** $5*10^{-4}$ (L2 Regularization)

Learning rate Scheduler - Cosine Annealing was used. Learning rate was set to 0.1 with a decay taking a cosine form towards close to zero after 35 epochs. This schedule enables the model to jump out of local minima during the initial stages of the training and narrow the gap towards the latter half.

Batch Size- 128

Epochs- 35

3. Baseline Experimental Results

3.1 Training Dynamics

The convergence in the training process was stable. Figure 1 shows that the training loss monotonically decreased and this shows that the loss is optimized well.

Final Training Accuracy- 100.0%

Final Test Accuracy- 86.9%

The fact that the training accuracy reaches 100% is the confirmation that the modified ResNet-18 can memorize the distribution of the training. The fact that training and test accuracy are not the same (the difference is about 13 percent) implies some overfitting, which is understandable since we did not use heavy data augmentation at the baseline phase (such as Cutout or Mixup).

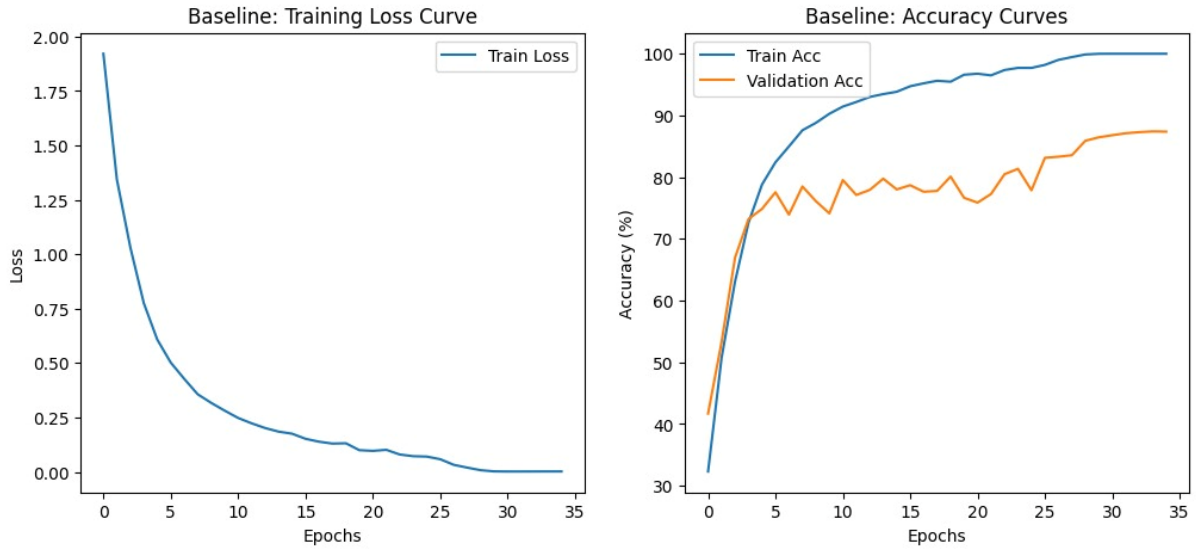

Figure 1

3.2 Analysis of Convergence – The Cosine Annealing scheduler then played an important role. We saw that the greatest improvements in the accuracy of the decision were in the first 15 epochs, and then the model refined the boundaries of the decision. The elimination of the first pooling layer proved to be effective because the model was able to learn how to make the distinction between fine-grained features (e.g., distinguishing "Cat" from "Dog") which would have been lost with the aggressive use of downsampling.

4. Failure Analysis: High Confidence Errors

An important demand of the trustworthy AI is calibration: when a model is correct, it should be confident. In order to assess this we selected a subset of the test set which contained only the instances of "Dangerous Failures" the ones in which the model made the incorrect prediction with a probability (confidence) value of > 0.90

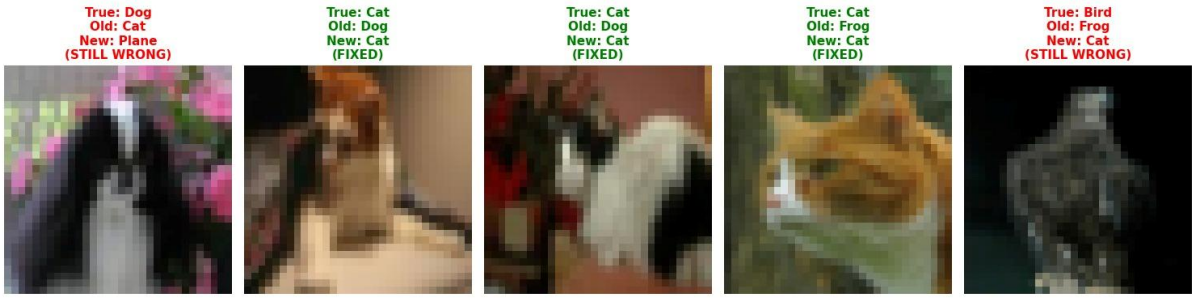

Figure -2

We have seen that photos on blue backgrounds were usually mixed up with pictures of birds or airplanes. To give an example, a ship on a blue ocean may be mistakenly identified as an airplane when the texture of the sky/water is prevalent in the convolutional filters..Animals (Cats, Dogs, Deer) were exchanged quite often. This implies that this model is based on local textural characteristics (fur patterns) and not overall shape characteristics (ear shape, snout

length). The fact that there are $99percent confident errors implies that this model is uncalibrated. The practical case, e.g. a self-driving car that identifies a stop sign as such, where the system is highly confident that it has made a wrong prediction, does not allow the human operator to regain control of the car, which is extremely unsafe.

5. Stress Testing: Robustness to Noise –

In order to test the reliability of the model further, we subjected the model to a stress test through the introduction of synthetic perturbation. It is known that deep learning models are sensitive to high-frequency noise which is usually invisible to the human eye.

5.1 Methodology-

We created a Noisy Test Set by adding Gaussian Noise to the normalized images. We then evaluated the pre-trained baseline model on this perturbed dataset without retraining.

5.2 Analysis – The show of accuracy greatly declining emphasizes the vulnerability of features learned. It is plausible that the model has been trained to use the precise pixel-level correlations perturbed by noise. This proves that even though the model can easily be seen as generalizing to clean data of the same distribution, it does not have a strong, semantic concept of the objects. It is rather a texture-matcher than a shape-recognizer.

6. Explainability (Grad-CAM) - In order to ascertain the reasons behind the model making certain choices, we have used Gradient-weighted Class Activation Mapping (Grad-CAM). It is a visualization method that shows the gradients into the last convolutional layer, creating a heatmap that shows the areas of the image that the model considered when making its prediction.

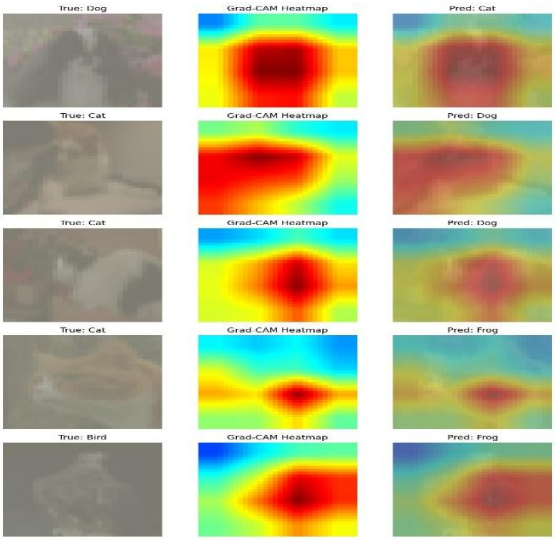

Figure – 3

7. Model Improvements and Comparative Analysis – After the critical analysis of the baseline model, we found the overfitting and sensitivity to noise as the major shortcomings. The baseline reached 100 percent training accuracy, but after that the test accuracy reached its

peak at around 87 percent; this implies that the model was not learning but memorising the training data. To address this, we implemented a second iteration of the model ("Improved Model") with specific regularization and data augmentation strategies. The Improved model had a large decrease in the generalization gap (the difference between the accuracy of training and test).

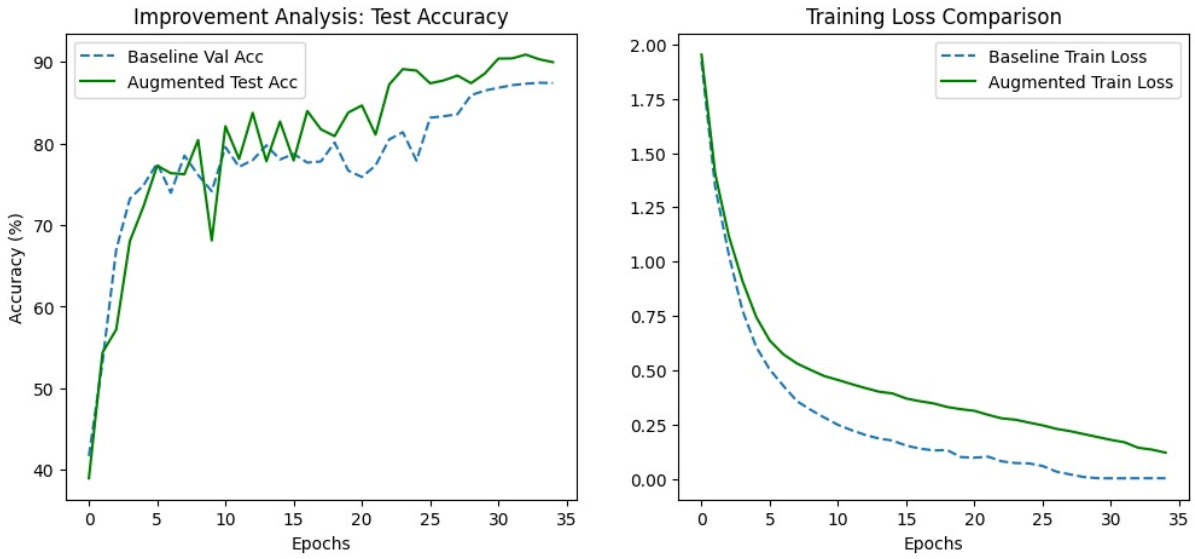

Figure - 4