# DL Mini Project

**Srihari Wuntakal - Amro Imam**

**Github Code Submission**

## Introduction

The CIFAR-10 [Krizhevsky, Nair, and Hinton(2009)] data set contains 60,000 colored images, each of a size of 32x32 pixels. These images are equally distributed into 10 classes as shown in **Figure-1**.

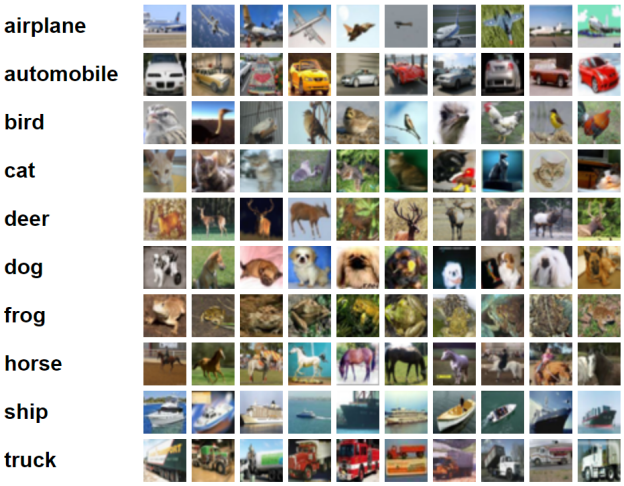



**Figure-1** The CIFAR-10 dataset, classified into 10 classes.

The task at hand is to make use of the data available from this compiled, labelled dataset to come up with a classifier that is accurate enough to perform general classifications for pictures within, and more importantly, outside the dataset. We will construct this classifier by creating a residual neural network (ResNet) [He et al.(2015)He, Zhang, Ren, and Sun], that uses the CIFAR-10 dataset for training, and then tweaking our architecture and model parameters (number of epochs, learning rate, number of layers, etc.) to optimize the accuracy of our predictions.



## Methodology

The basis of our model will be the ResNet-18; a network consisting of convolutional and pooling layers, skip connections, a fully connected layer and a softmax layer [Ramzan et al.(2019)Ramzan, Khan, Rehmat, Iqbal, Saba, Rehman, and Mehmood]. An example of the original architecture can be seen in **Figure-2**.

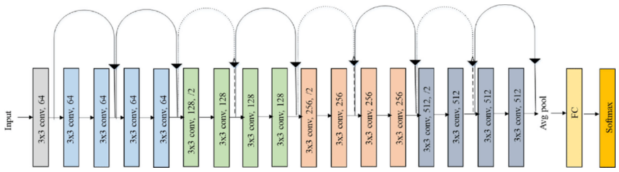


**Figure-2** The original ResNet-18 architecture.

This architecture using Python's PyTorch can be found in the following GitHub repository [Liu et al.(2020)Liu, Yang, Ducau, and Peiwen]. We trained this model for 200 epochs and were able to retrieve an accuracy of 95.46% with a hefty $11,183,582$ trainable parameters. The question then becomes: how can we do as well (almost), whilst keeping the number of trainable parameters under $5,000,000$?

There are multiple components of this architecture that we can tweak. In this attempt, we experimented with different learning rate values. number of epochs, dropout layers, batch size, pooling layers and optimizers. Some of these were altered in all of our attempts; we call these the global changes. Others were changed in one of our 5 attempts which we will outline in the local changes section. We have benefited immensely from the implementation of the Resnet using Keras from [JIANG(2021)].

### Global Changes

In this section, we list the components that were changed for each of the 5 attempts.

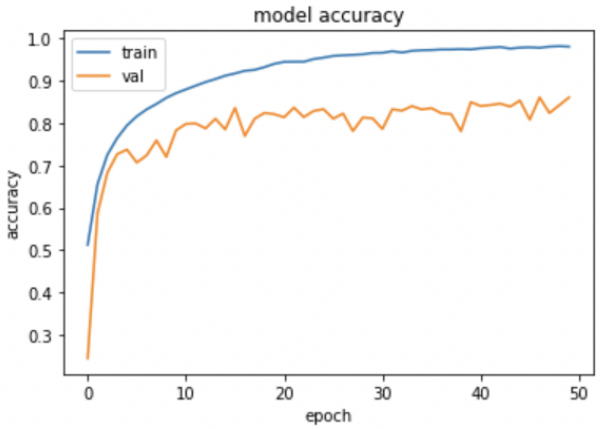
1. **Optimizer:** SGD Vs. Adam
   It was found that Adam gave us better accuracy predictions across the board.

2. **Learning Rate:** [0.0001, 0.001, 0.005, 0.05, 0.1] It was found that a learning rate = 0.001 was optimal.
3. **Data Augmentation** In effort to train the model using a larger number of examples, we augmented our data by performing horizontal flips on all of the images to create an extra 60,000 images for our training epochs.
4. **Removal of the final 512-channel layer.** Although possibly a compromise on the accuracy achieved with our model, but given that a lower number of parameters is desirable in this implementation, removal of this residual layers substantially decreased the number of our trainable parameters.
5. **Batch Normalization** A batch normalization of size 128 instead of the suggested 256 was used after the first 64-channel convolutional layer.

## Local Changes

In this section, keeping the global changes that we have aforementioned in mind, we list the specific alterations for each of our attempts, the test-accuracy along with the resulting model accuracy and loss plots.

**Attempt 1**
A MaxPooling layer of pool size 2x2, stride=2 and same-padding was used after the Batch-Norm layer.



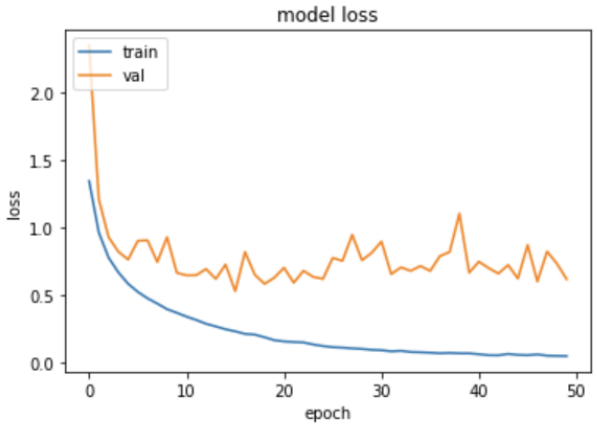
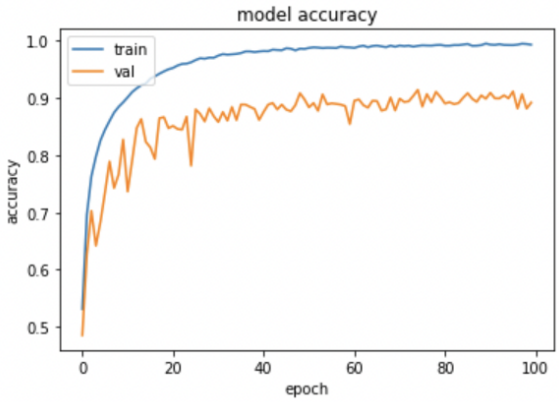




**Figure-3** Model accuracy & loss for attempt 1.

Epochs: 50
Trainable Parameters: 2,787,594
Model Accuracy: 76.09%

**Attempt 2**
The MaxPooling was removed, and epoch size was increased to 100



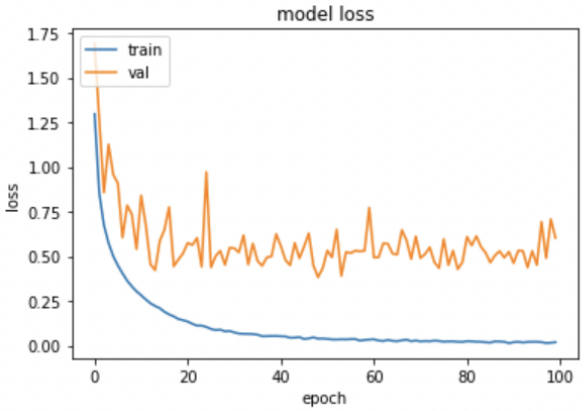




**Figure-4** Model accuracy & loss for attempt 2.

Epochs: 100
Trainable Parameters: 2,779,914
Model Accuracy: 88.57%

**Attempt 3**
The final AveragePooling layer was removed, and a MaxPooling layer was used instead. This was also followed by dropout layer set to 0.25. Dropout was used for regularization to prevent overfitting. Dropout was initially set to 0.5 but that proved counterproductive.

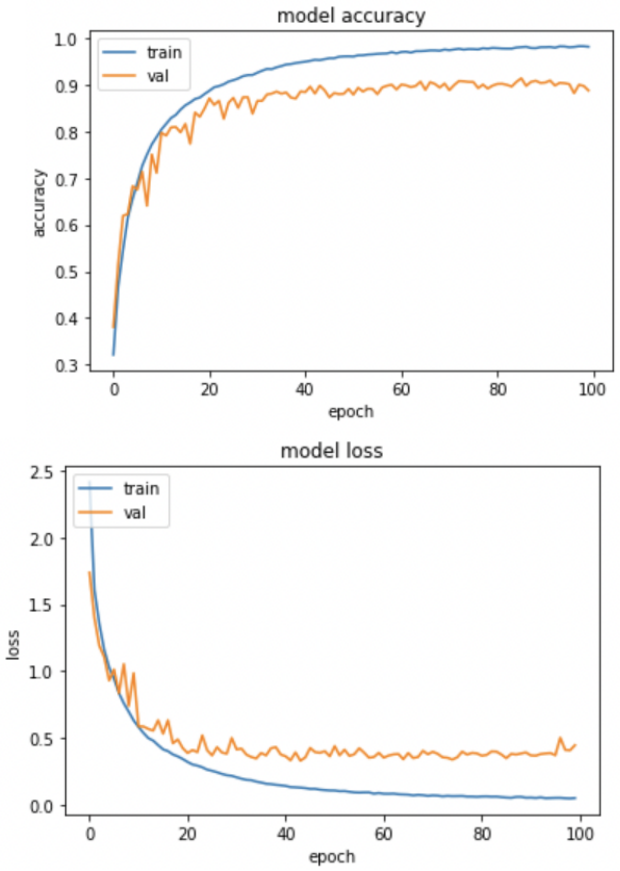
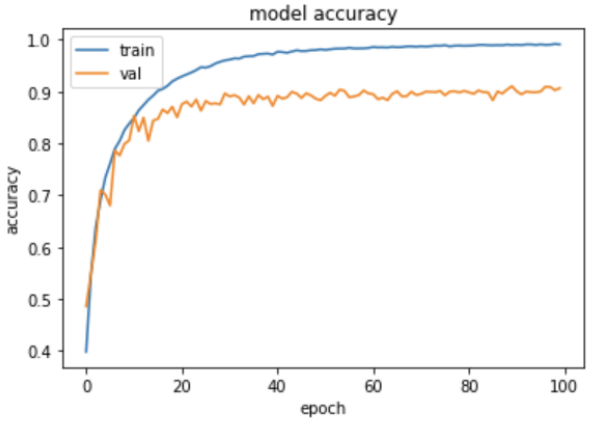



**Figure-5** Model accuracy & loss for attempt 3.

Epochs: 100
Trainable Parameters: 2,818,314
Model Accuracy: 88.95%

**Attempt 4**
One more drop-out layer with value set to 0.1 was added before the aforementioned drop-out layer in Attempt 3.



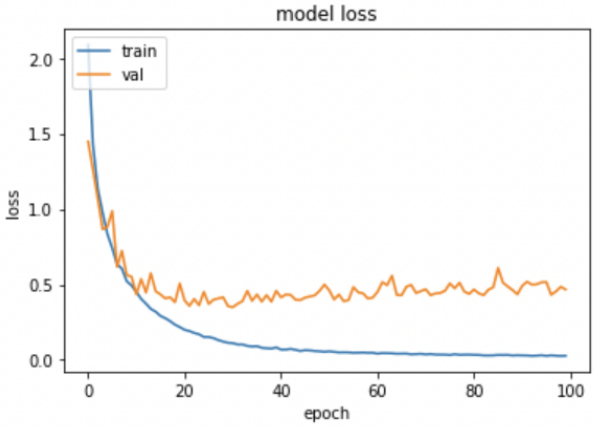




**Figure-6** Model accuracy & loss for attempt 4.

Epochs: 100
Trainable Parameters: 2,818,314
Model Accuracy: 89.90%

**Attempt 5**
The number of epochs was increased from 100 to 150.

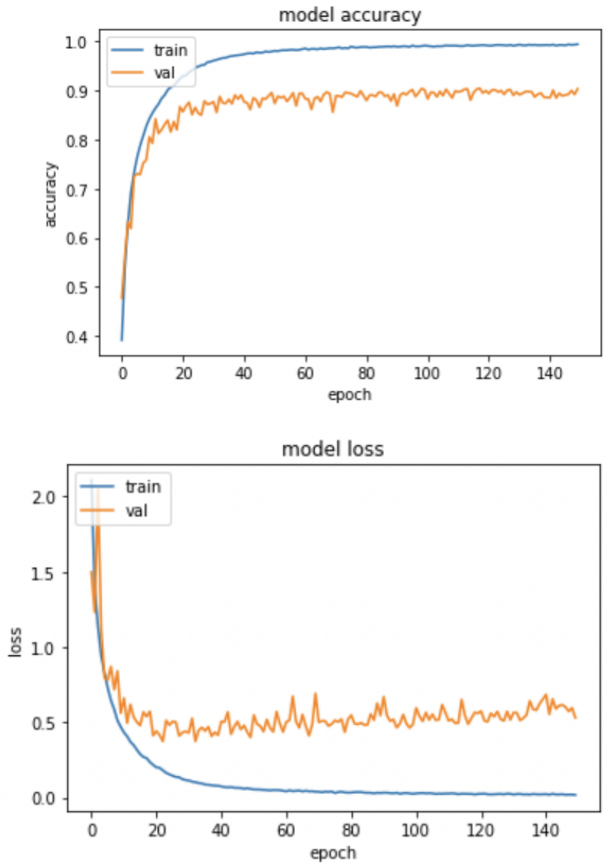



**Figure-7** Model accuracy & loss for attempt 5.

Epochs: 150
Trainable Parameters: 2,818,314
Model Accuracy: 90.88%

## Results

The following architecture shown in **Figure-8** was arrived to by adding batch normalization after our first convolutional layer, removal of the final 512-channel layer, substituting the AveragePooling layer for a MaxPooling layer and finally, adding dropout. These are the pertinent characteristics of our optimized architecture:
Trainable Parameters: $2,818,314$
Model Accuracy: 90.88%

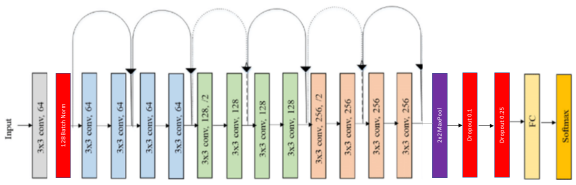



**Figure-8** The optimized architecture. Red blocks denote added processes, and purple denotes added layers.

Future endeavours to further optimize this architecture can include experimenting with different optimizers. Momentum seems like a potential candidate for better results [Choi et al.(2019)Choi, Shallue, Nado, Lee, Maddison, and Dahl].

## References


[Choi et al.(2019)Choi, Shallue, Nado, Lee, Maddison, and Dahl] Choi, D.; Shallue, C. J.; Nado, Z.; Lee, J.; Maddison, C. J.; and Dahl, G. E. 2019. On Empirical Comparisons of Optimizers for Deep Learning.

[He et al.(2015)He, Zhang, Ren, and Sun] He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Deep Residual Learning for Image Recognition. *CoRR*, abs/1512.03385.

[JIANG(2021)] JIANG, R. 2021. Implementing resnet-18 using keras.

[Krizhevsky, Nair, and Hinton(2009)] Krizhevsky, A.; Nair, V.; and Hinton, G. 2009. CIFAR-10 (Canadian Institute for Advanced Research).

[Liu et al.(2020)Liu, Yang, Ducau, and Peiwen] Liu, K.; Yang, W.; Ducau, F.; and Peiwen, Y. 2020. Train CIFAR10 with PyTorch.

[Ramzan et al.(2019)Ramzan, Khan, Rehmat, Iqbal, Saba, Rehmar Ramzan, F.; Khan, M. U.; Rehmat, A.; Iqbal, S.; Saba, T.; Rehman, A.; and Mehmood, Z. 2019. A Deep Learning Approach for Automated Diagnosis and Multi-Class Classification of Alzheimer's Disease Stages Using Resting-State fMRI and Residual Neural Networks. *Journal of Medical Systems*, 44.