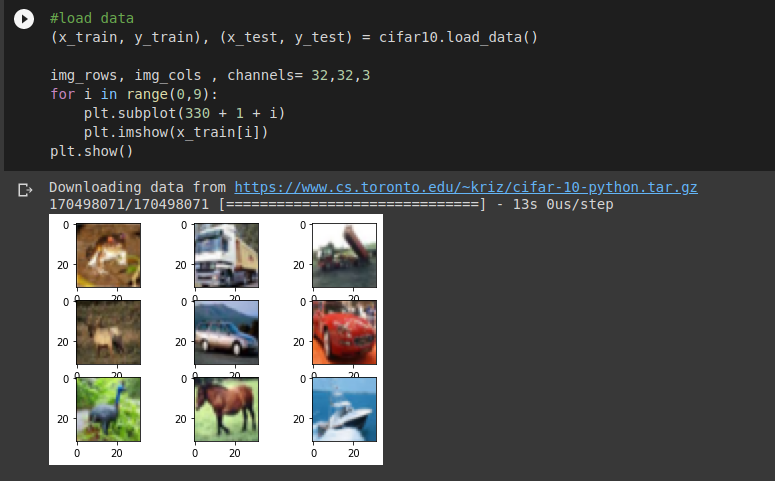
FCC Lab Assessment 02

Manoj Kumar D. - MS22048400

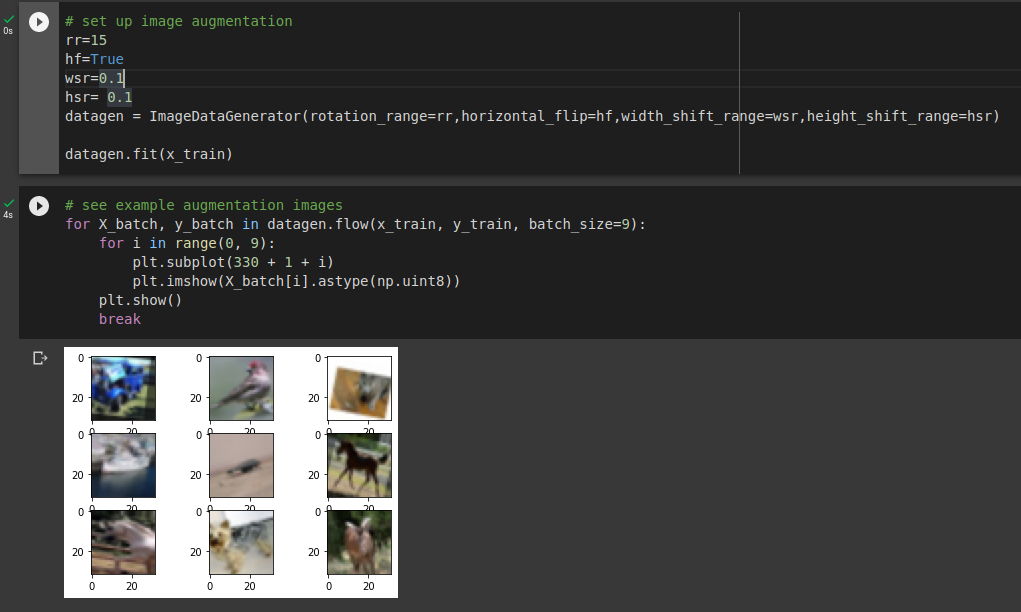
## Introduction

* One of the challenges in image classification is the need for large amounts of data to train models. The CIFAR-10 dataset is a popular benchmark dataset for image classification, consisting of 60,000 32x32 colour images in 10 classes. However, the limited size of the dataset can limit the performance of models trained on it. One way to overcome this limitation is through image augmentation, which involves generating new images by applying various transformations to existing ones. In this report, we explore the impact of image augmentation on image classification using the CIFAR-10 dataset. Specifically, we assess the performance of various machine learning algorithms in classifying augmented images and compare them to their performance on the original images. This report aims to provide insights into the effectiveness of image augmentation as a technique to improve image classification accuracy on the CIFAR-10 dataset.

## Data Preparation

* The dataset needs to be properly preprocessed before it can be used for training and evaluation. I have also applied image augmentation techniques, such as rotation, flipping, and zooming, to increase the diversity of the dataset and reduce the risk of overfitting. Additionally, we split the dataset into training, validation, and testing sets to ensure that the performance of our machine learning models is evaluated on unseen data.
* As it is visible in the above image, the dataset is loaded, divided into training and validation sets and some sample images are plotted to show some contents of the dataset.

### Augmentation:



The code snippet defines an instance of the ImageDataGenerator class from the Keras library, which will perform image augmentation on the input data. Image augmentation is a technique used to artificially increase the size of a dataset by creating modified versions of existing images. In this code, the augmentation parameters are specified, including a rotation range of 15 degrees, horizontal flipping, and random horizontal and vertical shifts of up to 10% of the image size. These parameters are designed to introduce variability into the training data and prevent overfitting of the model to the specific training images. Overall, the ImageDataGenerator class provides a powerful tool for improving the performance of deep learning models on image classification tasks.

## Model Architecture

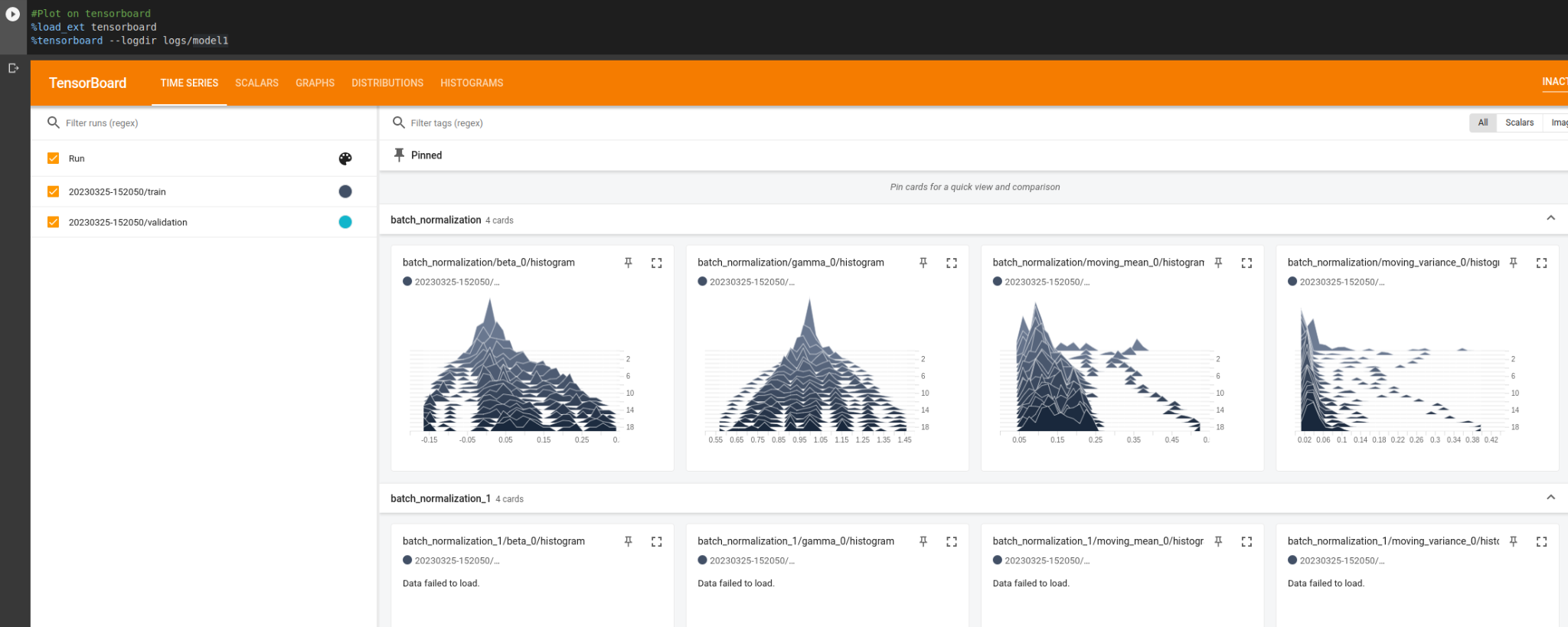
* The model architecture is a convolutional neural network (CNN) that is used for image classification. The model consists of several layers that perform different functions. Firstly, the model adds a convolutional layer with 32 filters, each having a 3x3 kernel size, and the activation function used is Rectified Linear Unit (ReLU). The padding is set to 'same' to ensure the output size is the same as the input size. Batch normalisation is applied to normalise the output of the previous layer across the mini-batch. This is followed by another convolutional layer with the same configuration as the previous one, except that the number of filters remains the same, and a max-pooling layer with a 2x2 pool size is added to reduce the spatial dimensions of the output. Dropout is also applied to the convolutional output to prevent overfitting.
* The next set of layers consists of two convolutional layers with 2 times the number of filters (64 filters in this case), followed by a max-pooling layer, and a dropout layer. Similarly, two more convolutional layers are added, this time with 4 times the number of filters (128 filters in this case), followed by a max-pooling layer and a dropout layer. The max-pooling layers are used to reduce the spatial dimensions of the output, and the dropout layers help prevent overfitting.
* Finally, the output of the convolutional layers is flattened and passed through two fully connected dense layers with 512 and 10 neurons, respectively. Batch normalisation is applied to the first dense layer, and dropout is applied to both dense layers. The activation function used for the last dense layer is softmax, and the loss function used for training is categorical cross-entropy. The model is optimised using the Adam optimizer with a learning rate of 0.001, a decay rate of 0, and default beta\_1, beta\_2, and epsilon values. The goal of this model is to classify images from the CIFAR-10 dataset using image augmentation techniques.

## Model Training

* The code is training the model using the fit method of the model object. The training data x\_train and their corresponding labels y\_train are passed as arguments. The training process will iterate through the entire dataset epochs number of times, with each iteration consisting of mini-batches of size batch\_size. During training, the performance of the model is evaluated on the validation set specified by validation\_data, which consists of x\_test and their corresponding labels y\_test. The basemodel\_tb\_callback is a callback function that is executed at the end of each epoch, which can be used to perform various tasks such as logging the training progress, saving the model, or stopping early.
* The hyperparameters used are:
  + batch\_size: 128
  + epochs: 20
* The performance metrics are recorded in the history object, which is returned by the fit method. The metrics recorded are the loss and accuracy of the model on both the training and validation sets for each epoch. These metrics can be accessed using the history.history attribute.

## Results

### Training without Image Augmentation

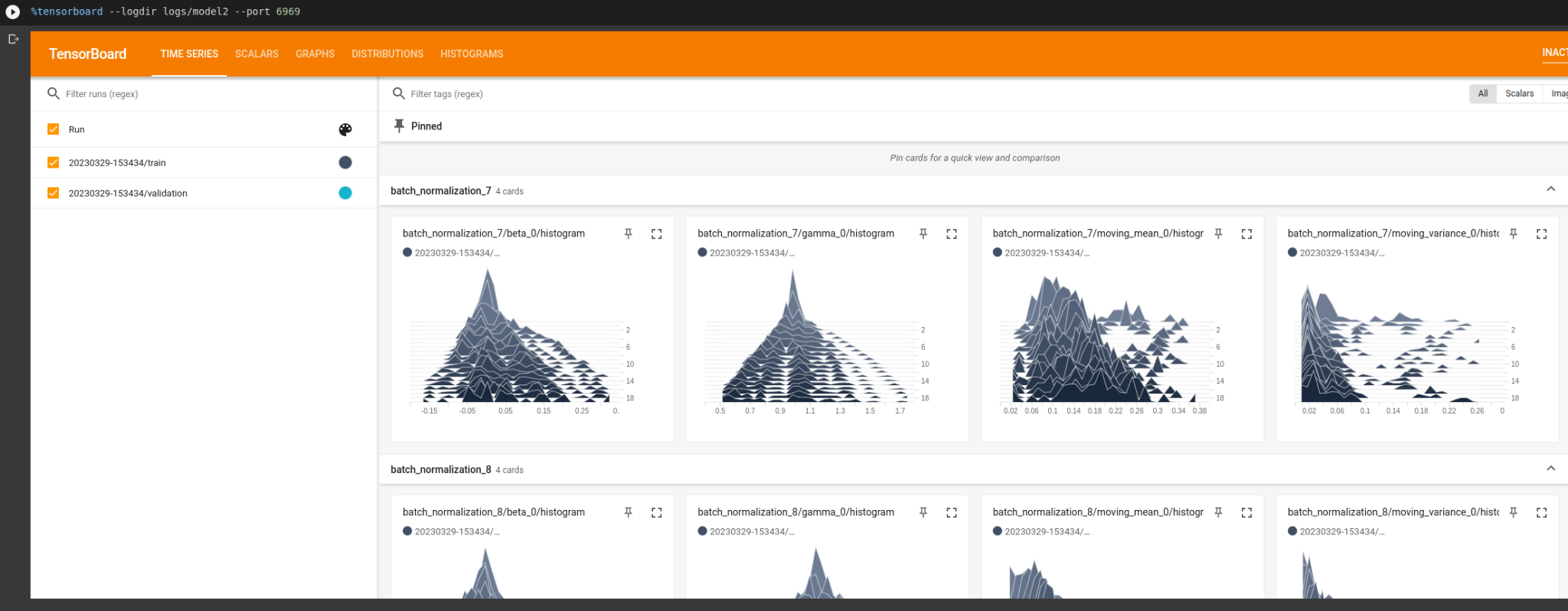


loss: 0.0642 - accuracy: 0.9778

Accuracy



### Training with Image Augmentation



loss: 0.3487 - accuracy: 0.8802

Accuracy:

## Model Interpretation

Based on the evaluation metrics of the two models and the above graphs, the initial model without image augmentation has high accuracy with low loss. This suggests that the model is performing well in classifying the images from the CIFAR-10 dataset. On the other hand, the second model, which includes image augmentation, has relatively low accuracy with relatively high loss. While this may seem concerning, it is not necessarily unexpected as image augmentation can sometimes lead to a decrease in accuracy and increase in loss. It is important to note that these metrics should be interpreted in the context of the specific problem and dataset being analysed.

## Conclusion

The report discusses the use of image augmentation in improving image classification accuracy on the CIFAR-10 dataset. The dataset is preprocessed, and various image augmentation techniques are applied to increase the dataset's diversity and reduce the risk of overfitting. A convolutional neural network (CNN) is used for image classification, consisting of several layers performing different functions. The model is trained and evaluated using the fit method, and the performance metrics are recorded in the history object. The results show that the model without image augmentation has high accuracy with low loss, while the model with image augmentation has relatively low accuracy with relatively high loss. The report concludes that image augmentation can improve the performance of deep learning models on image classification tasks.