CNN-Based Image Classification on CIFAR-10 Dataset

Abdullah Farooq, Executive Member, ACM, Ammar Ahmad, *Undergrad Junior, GIKI,* and
Bilal Malik, *Undergrad Junior, GIKI*

Abstract:

This report presents a comprehensive approach to implementing a Convolutional Neural Network (CNN) for image classification using Python's Keras library. It describes the process of loading, visualizing, and preprocessing the CIFAR-10 dataset, followed by the design and training of multiple variants of a CNN model. Each model is evaluated on various performance metrics, with comparisons based on performance scores.

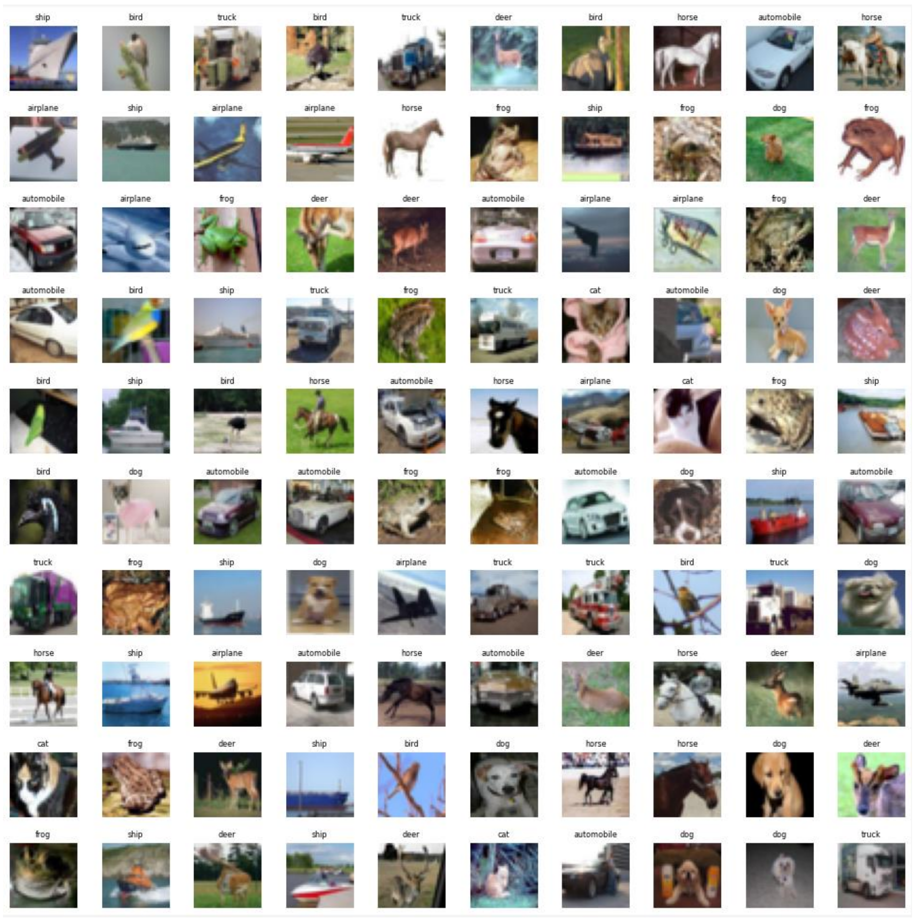
Introduction:

The CIFAR-10 dataset comprises 60,000 color images of size 32x32 pixels, evenly distributed across ten distinct classes. The main goal is to develop an accurate multi-classification model to classify these images effectively.

Methodology:

1. ***Data Loading & Visualization:***

   Initially, the CIFAR-10 dataset is loaded using the Keras library, followed by some preliminary visualization of sample images and class distribution.



2. ***Data Preprocessing:***

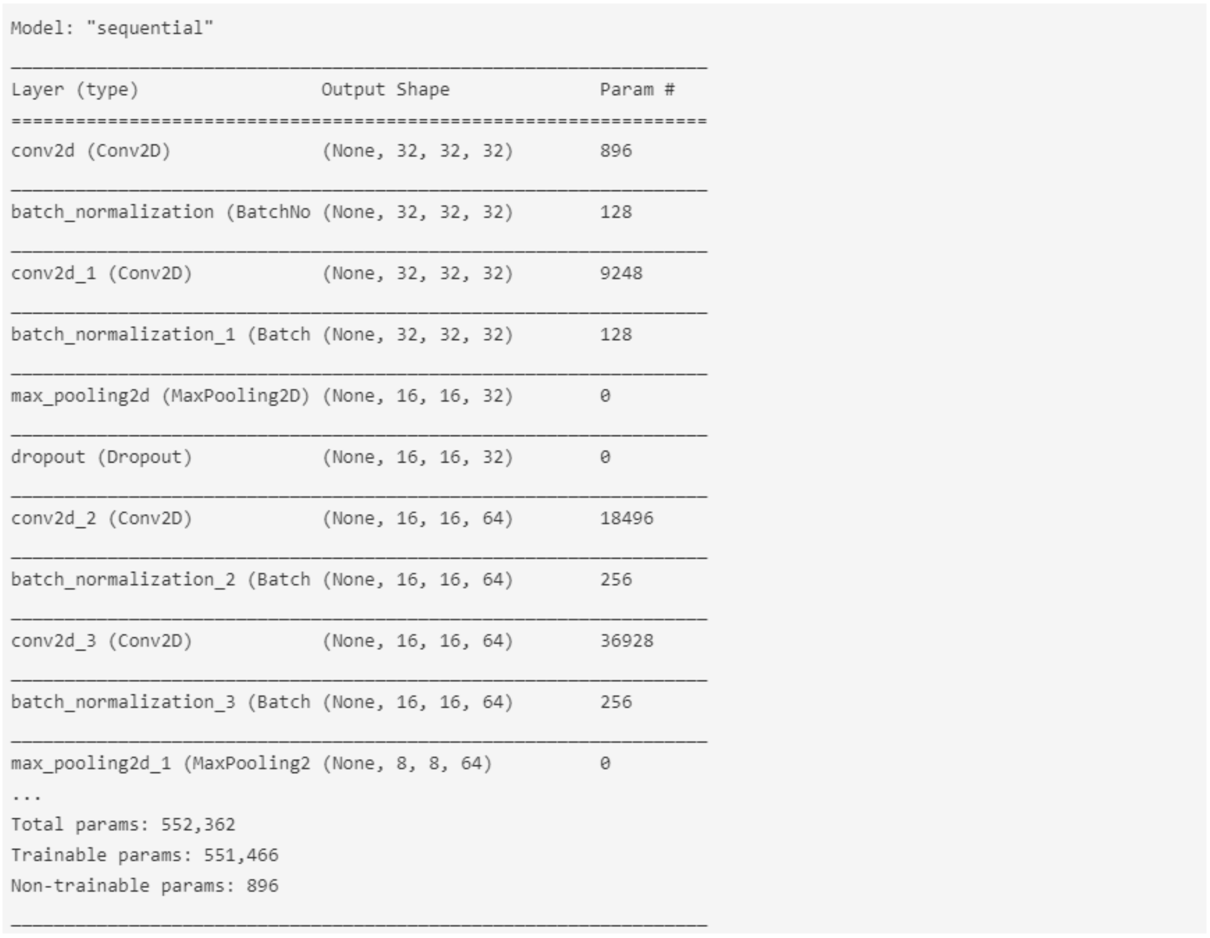
The raw images are preprocessed by normalizing pixel values and implementing one-hot encoding on target labels.

3. ***Model Building & Training:***

Five distinct models are built to compare the effects of different architectural decisions and techniques.

1. A simple Neural Network model.
2. A primary Convolutional Neural Network (CNN) model.
3. A CNN model with Dropout layers to prevent overfitting.
4. A CNN model with BatchNormalization layers to accelerate learning and further prevent overfitting.
5. An augmented CNN model with BatchNormalization and Dropout layers, trained using Adam optimizer and data augmentation techniques.

The models are trained for 15 epochs using a batch size of 32 and a training-validation split of 80-20.



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 32, 32, 32)        896
_________________________________________________________________
batch_normalization (BatchNo (None, 32, 32, 32)        128
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 32, 32, 32)        9248
_________________________________________________________________
batch_normalization_1 (Batch (None, 32, 32, 32)        128
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 16, 16, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 16, 16, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 16, 16, 64)        18496
_________________________________________________________________
batch_normalization_2 (Batch (None, 16, 16, 64)        256
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 16, 16, 64)        36928
_________________________________________________________________
batch_normalization_3 (Batch (None, 16, 16, 64)        256
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 8, 8, 64)          0
...
Total params: 552,362
Trainable params: 551,466
Non-trainable params: 896
_________________________________________________________________
```

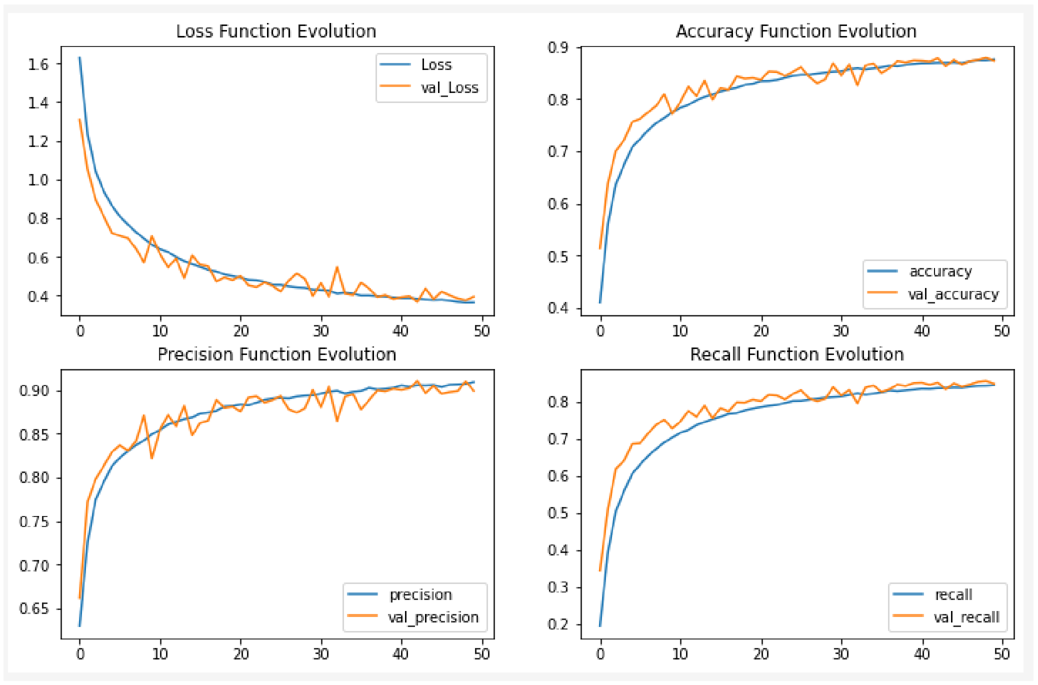
Model Summary

```
1562/1562 [==============================] - 38s 20ms/step - loss: 1.6282 - accuracy: 0.4095 - precision: 0.6303
Epoch 2/50
1562/1562 [==============================] - 30s 19ms/step - loss: 1.2323 - accuracy: 0.5607 - precision: 0.7258
Epoch 3/50
1562/1562 [==============================] - 30s 19ms/step - loss: 1.0423 - accuracy: 0.6369 - precision: 0.7746
Epoch 4/50
1562/1562 [==============================] - 31s 20ms/step - loss: 0.9402 - accuracy: 0.6734 - precision: 0.7946
Epoch 5/50
1562/1562 [==============================] - 30s 19ms/step - loss: 0.8658 - accuracy: 0.7073 - precision: 0.8127
Epoch 6/50
1562/1562 [==============================] - 30s 19ms/step - loss: 0.8097 - accuracy: 0.7230 - precision: 0.8225
Epoch 7/50
1562/1562 [==============================] - 29s 18ms/step - loss: 0.7682 - accuracy: 0.7402 - precision: 0.8301
Epoch 8/50
1562/1562 [==============================] - 29s 19ms/step - loss: 0.7284 - accuracy: 0.7535 - precision: 0.8370
Epoch 9/50
1562/1562 [==============================] - 30s 19ms/step - loss: 0.6954 - accuracy: 0.7633 - precision: 0.8422
Epoch 10/50
1562/1562 [==============================] - 29s 19ms/step - loss: 0.6631 - accuracy: 0.7744 - precision: 0.8495
Epoch 11/50
1562/1562 [==============================] - 30s 19ms/step - loss: 0.6415 - accuracy: 0.7828 - precision: 0.8544
Epoch 12/50
1562/1562 [==============================] - 29s 19ms/step - loss: 0.6253 - accuracy: 0.7884 - precision: 0.8607
Epoch 13/50
1562/1562 [==============================] - 29s 19ms/step - loss: 0.5998 - accuracy: 0.7972 - precision: 0.8636
...
Epoch 49/50
1562/1562 [==============================] - 30s 19ms/step - loss: 0.3646 - accuracy: 0.8738 - precision: 0.9070
Epoch 50/50
1562/1562 [==============================] - 30s 19ms/step - loss: 0.3649 - accuracy: 0.8755 - precision: 0.9091
```

Model Training

4. *Model Evaluation:*

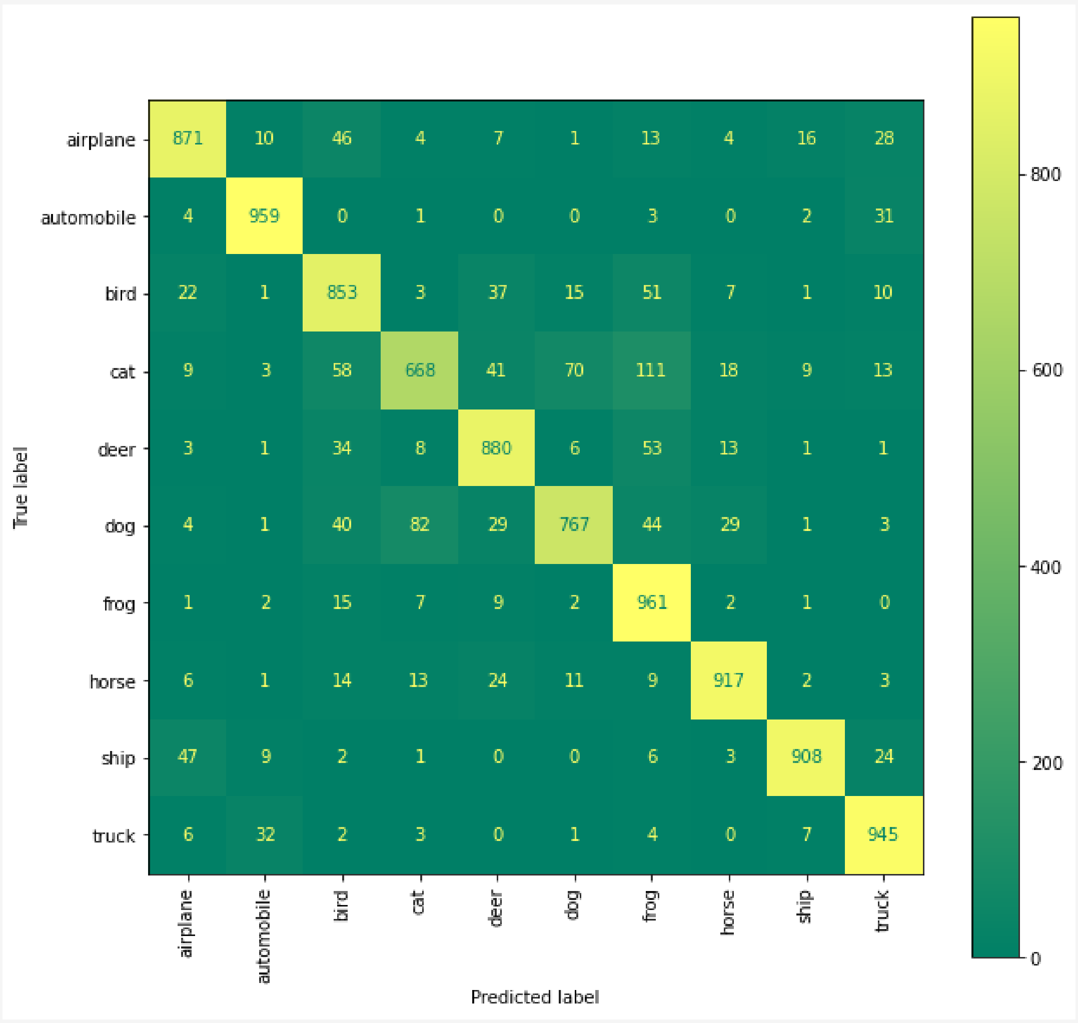
After training, the models' performances are evaluated using accuracy as a metric on the test dataset. The training and validation loss curves are plotted to analyze the models' learning over the epochs.



Results:

The model evaluation results are tabulated, demonstrating how each model variant performed in terms of accuracy on the test dataset. Further, critical insights derived from the loss curves of each model are discussed.

```
313/313 [==============================] - 1s 4ms/step - loss: 0.3949 - accuracy: 0.8729 -
Test Accuracy : 87.29%
```



True label \ Predicted label	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	871	10	46	4	7	1	13	4	16	28
automobile	4	959	0	1	0	0	3	0	2	31
bird	22	1	853	3	37	15	51	7	1	10
cat	9	3	58	668	41	70	111	18	9	13
deer	3	1	34	8	880	6	53	13	1	1
dog	4	1	40	82	29	767	44	29	1	3
frog	1	2	15	7	9	2	961	2	1	0
horse	6	1	14	13	24	11	9	917	2	3
ship	47	9	2	1	0	0	6	3	908	24
truck	6	32	2	3	0	1	4	0	7	945

Conclusion:

This report showcased the development of a CNN-based image classification model using the CIFAR-10 dataset. The model-building process underlined the significant roles of data preprocessing, careful architectural decisions, and regularization techniques in achieving a robust and accurate model. The comparison of different models' performance provided vital insights into the benefits of various CNN enhancements, ultimately leading to the selection of the optimal model configuration.

Reference(s):

- https://www.cs.toronto.edu/~kriz/cifar.html