**CNN-Based Road Sign Classification Project Report**

**Vidhu Govind**

**DA/DS November FN**

**1. Introduction**

This project involves developing a Convolutional Neural Network (CNN) to classify images of road signs into 30 distinct classes. The model is trained using augmented image data and further optimized using hyperparameter tuning. A Streamlit web application was also developed to allow real-time image prediction.

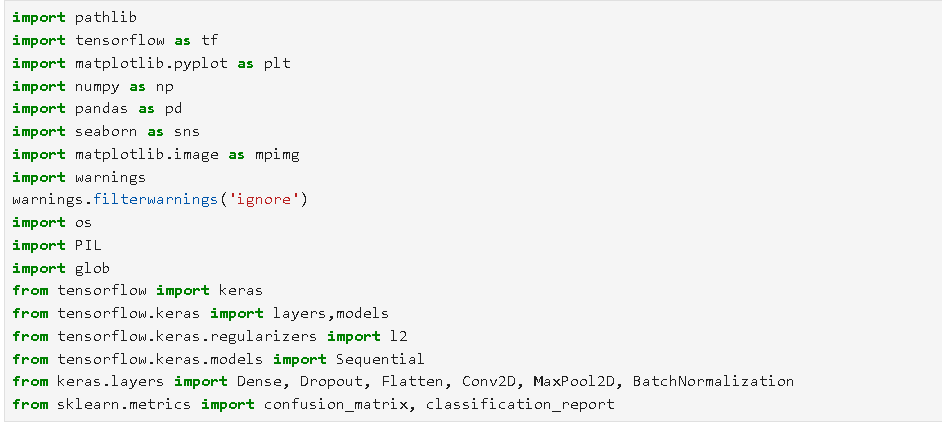
**2. Dataset Description**

The dataset consists of images categorized into 30 road sign classes such as stop signs, speed limits, and yield signs. Each image is labeled according to its class using a separate CSV file. The images are resized to 180x180 pixels and normalized during preprocessing.

**3. Data Preprocessing & Augmentation**

Preprocessing steps include resizing images, normalization (pixel values scaled to [0, 1]), and label encoding. Data augmentation techniques were applied using TensorFlow's Sequential API. These techniques included horizontal/vertical flips, random rotation, zoom, contrast adjustments, and translation.

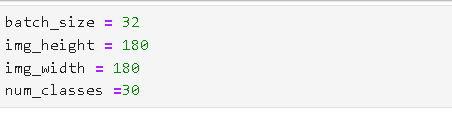
Libraries and Tools



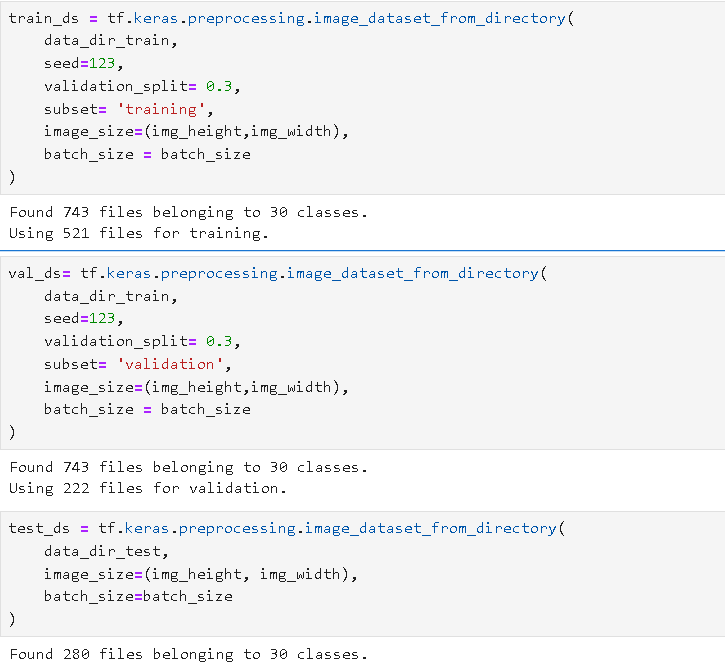
**Data Loading**



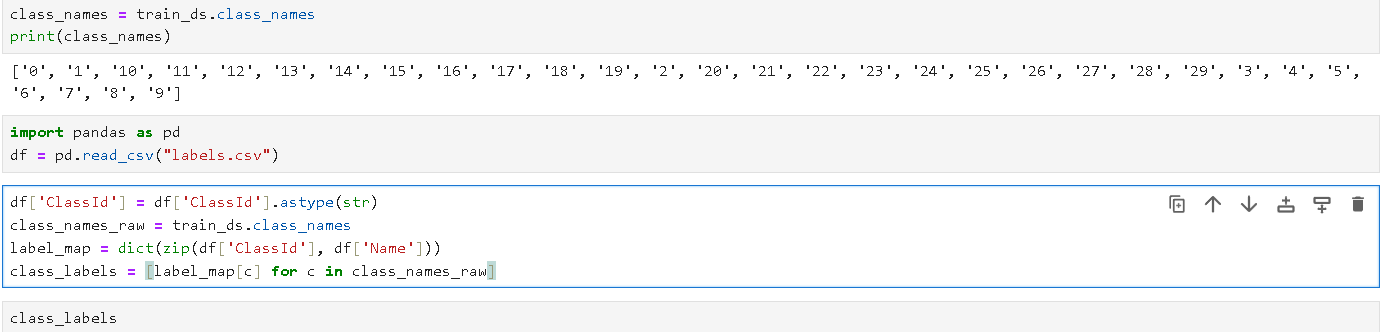
**Preprocessing:**



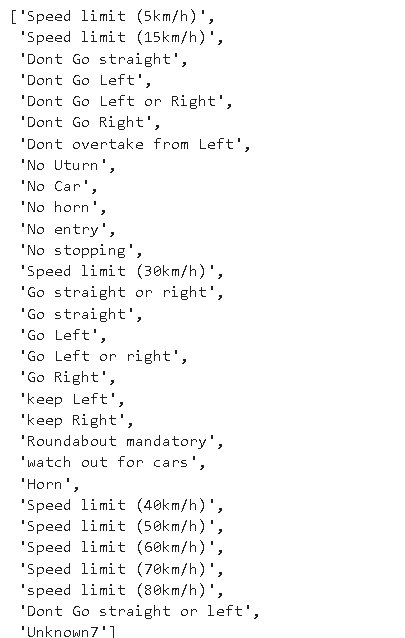
**Data Splitting:**



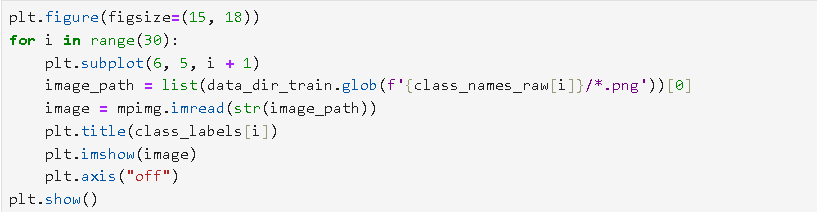
**Class Names -labelling:**



**Labelled Class Names:**



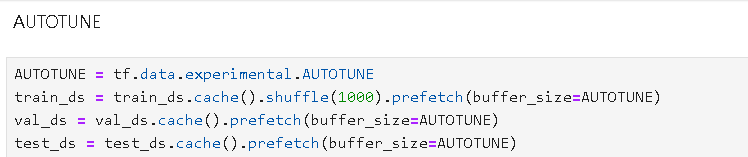
**Images and Classes:**



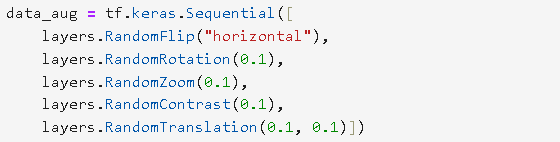




**Autotune:**



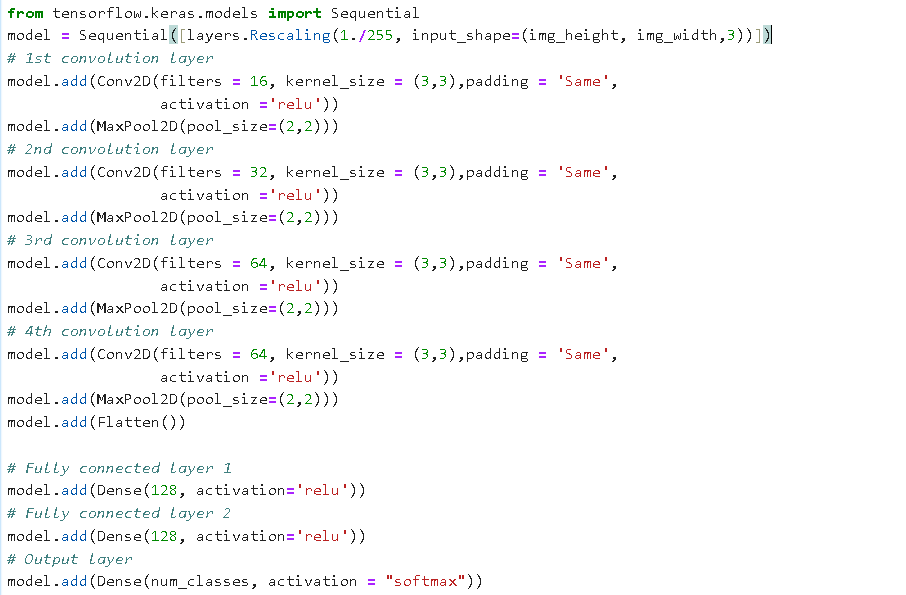
**Data Augmentation:**



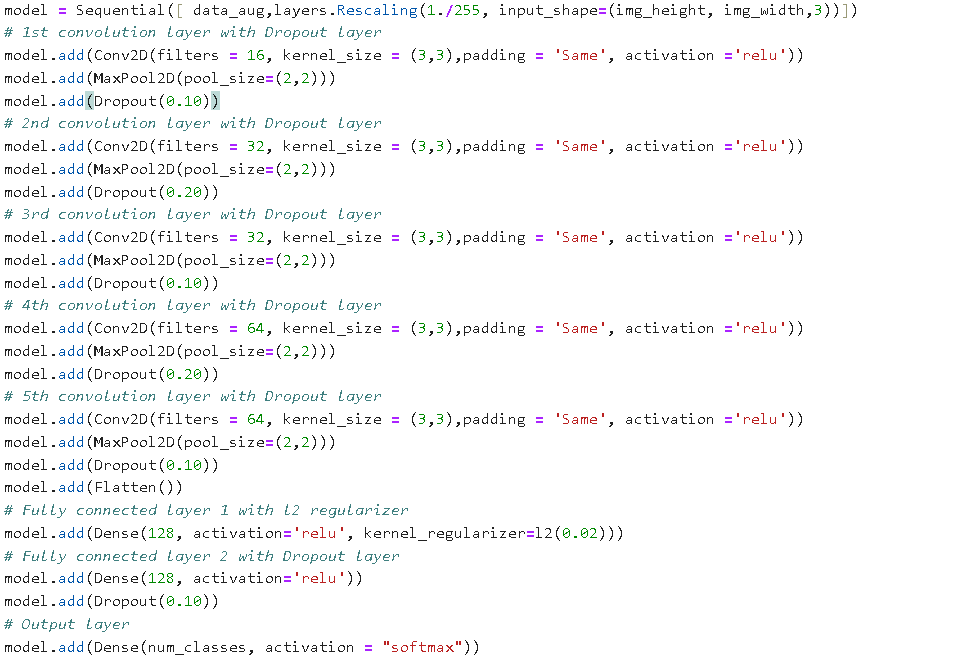
**4. CNN Model Architecture**

The CNN consists of 5 convolutional layers, each followed by max pooling and dropout layers to prevent overfitting. Two fully connected dense layers are used, with the first one regularized using L2. The output layer uses softmax activation for multi-class classification. A model without data augmentation was also created, which marked better accuracy than augmented model.

Code: - without augmentation



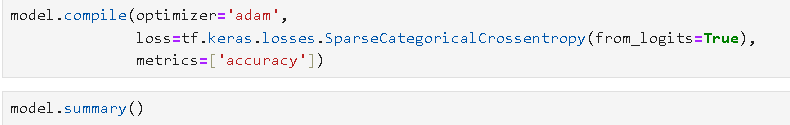
**With Augmentation:**

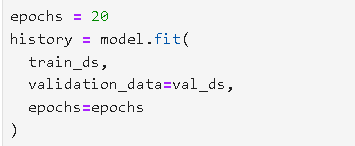


**5. Model Compilation & Training**

The model was compiled using the Adam optimizer and Sparse Categorical Crossentropy as the loss function. Early stopping and model checkpoint callbacks were used to monitor validation performance. The model was trained on the augmented training dataset and validated using a 15% split.

**Code:**

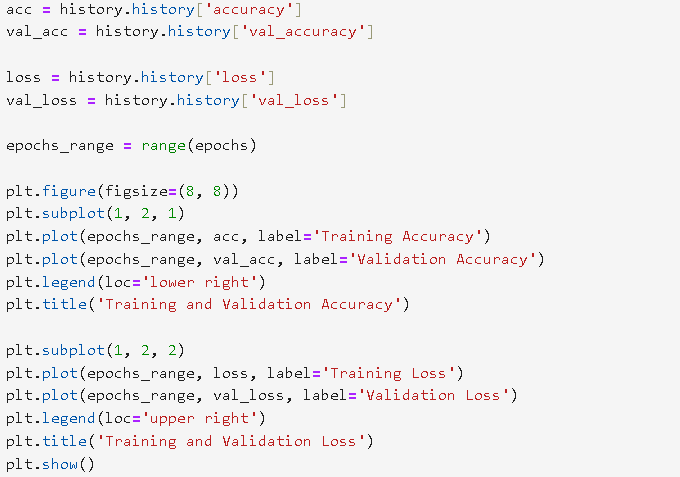




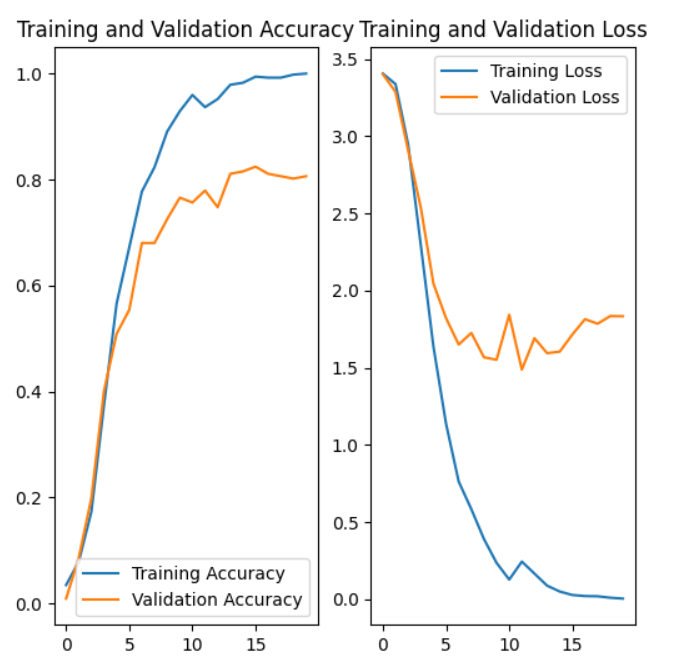
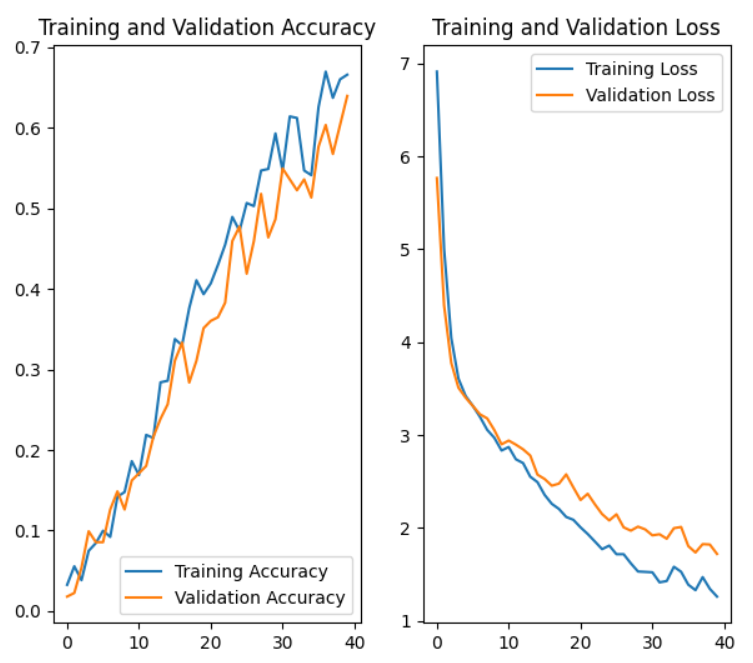
Epochs=20,40 were done.

**Results:**

Code for visualization:



**Visualization:**

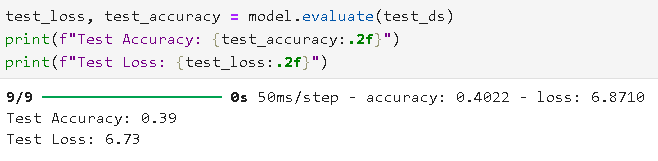


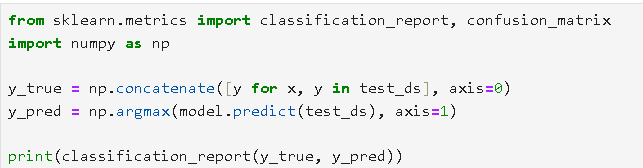
Before augmentation After augmentation

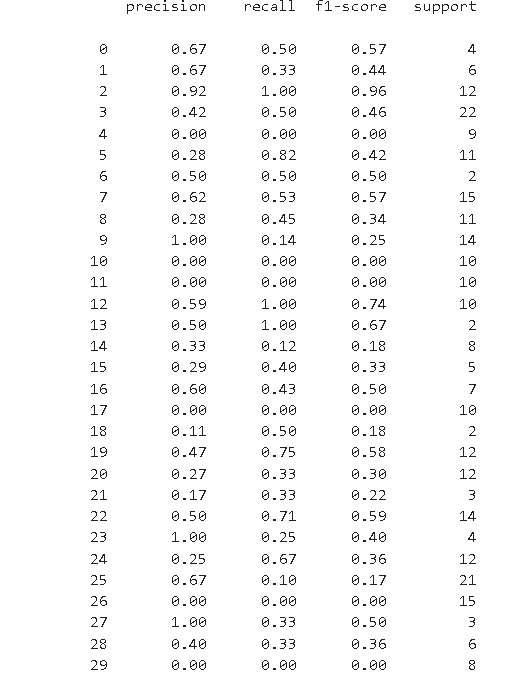
**Valuation Metrics:**

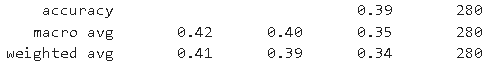
Training and validation accuracy is near 65-67% for augmented data and 100 and 82% for non-augmented one. Test accuracy is a bit very low for augmentation included model. But its better (but still low for non-augmented)

**For non-augmented:**

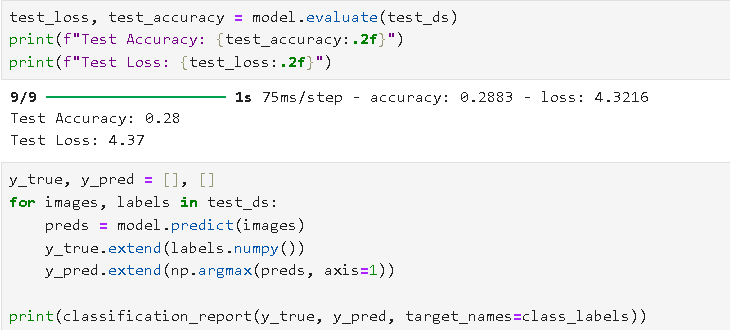


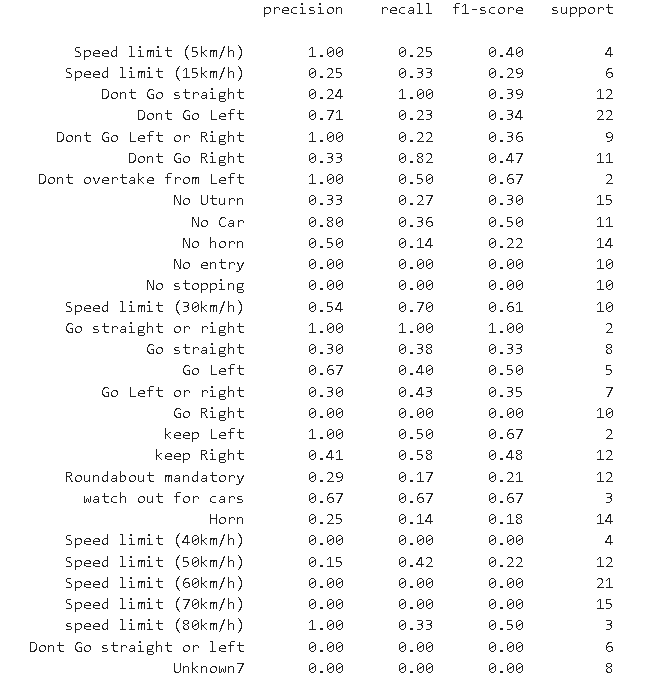


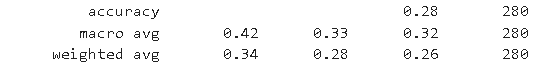




**For Augmented data:**







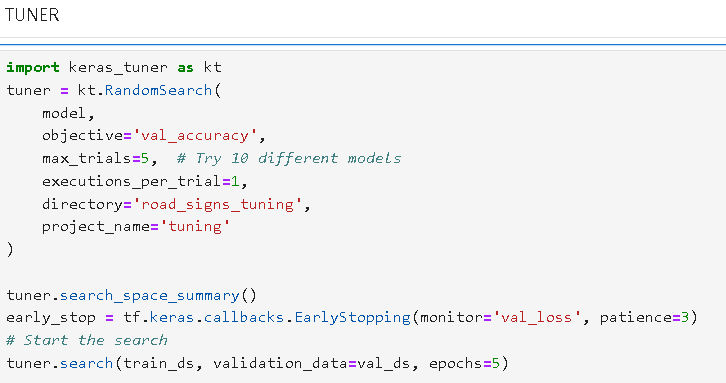
**6.Hyperparameter Tuning:**

Keras Tuner’s RandomSearch was used to explore different hyperparameter combinations including filters, kernel size, dense layer units, dropout rates, and learning rate. The best performing model from tuning was retrained for additional epochs.

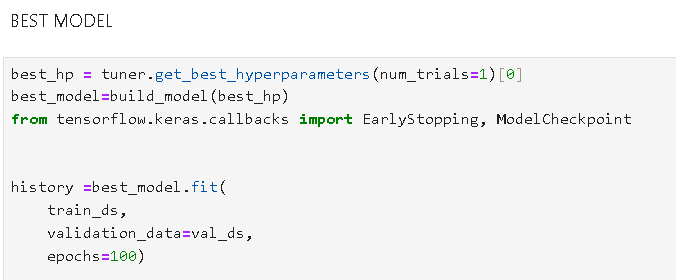
Code:



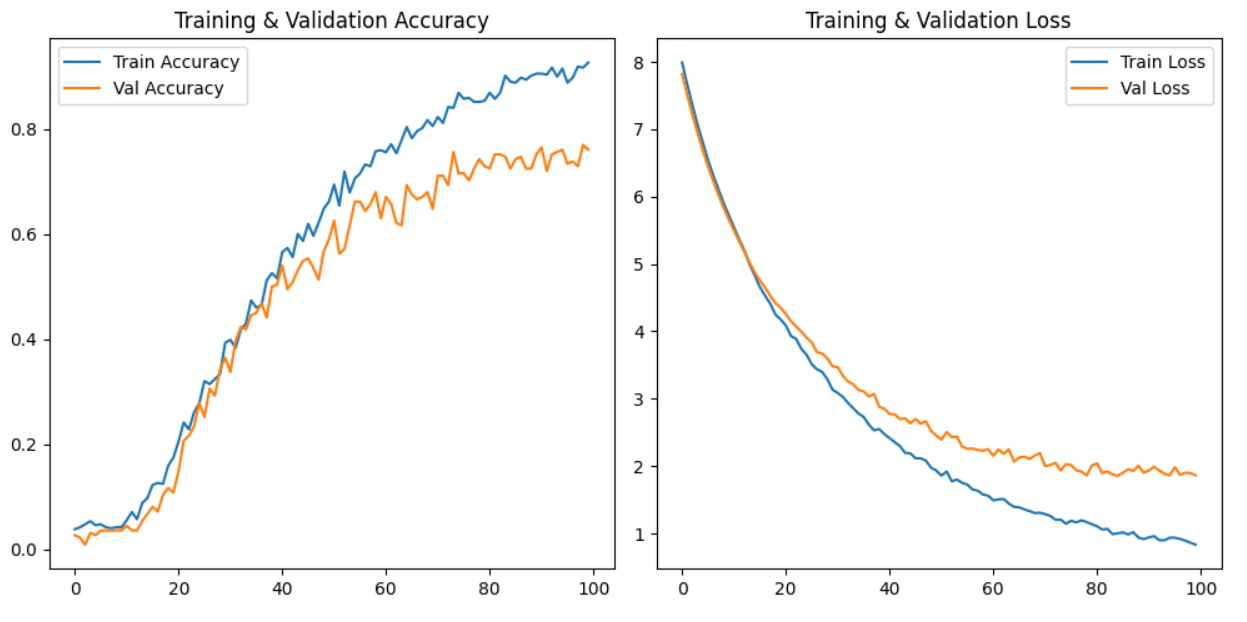


**Tuner**:

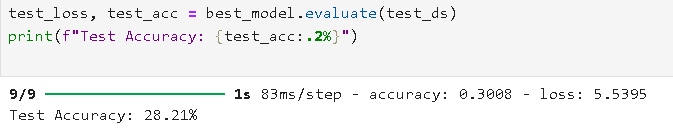
**Model:**

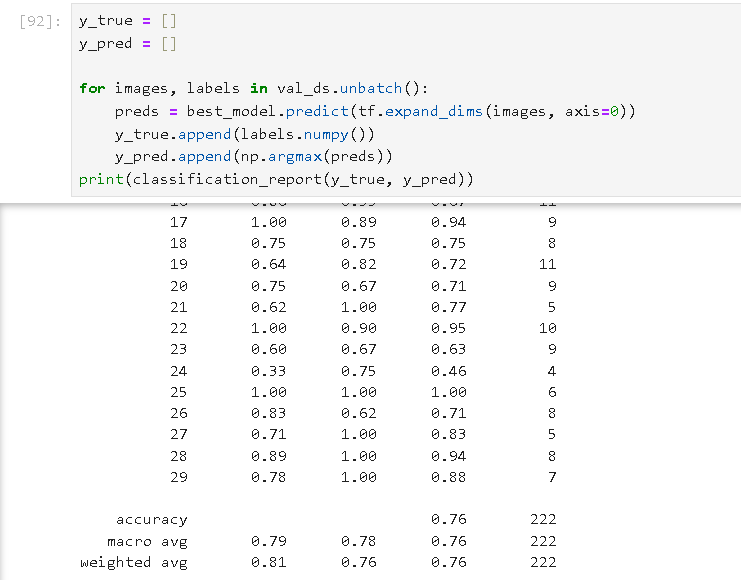


**Results:**



**Test Accuracy**:

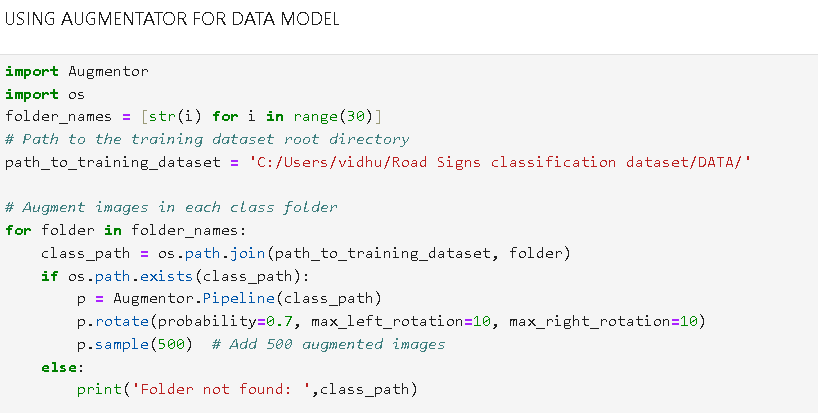


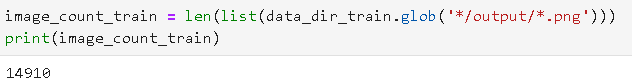


**With Augmenter:**

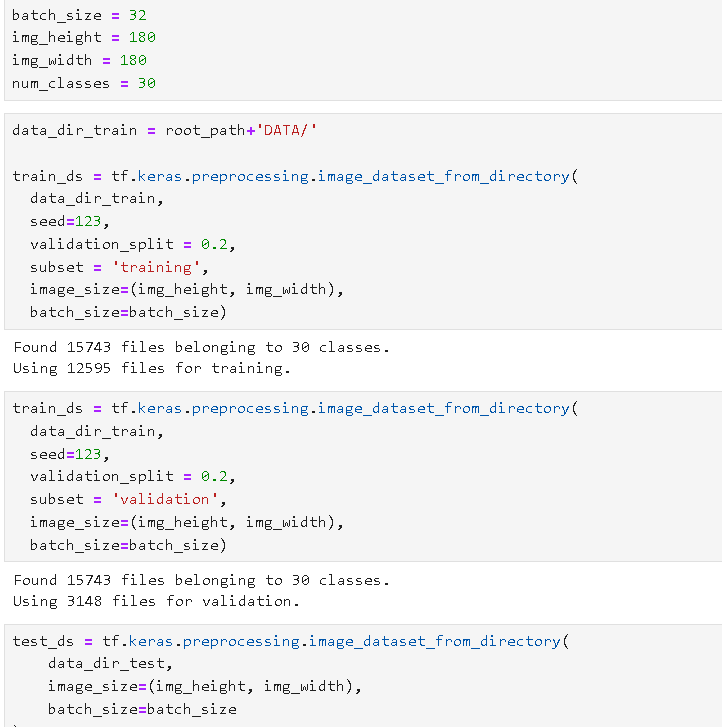
With data augmenter training data is balanced by adding images and model is made and compiled again.

**Code:**

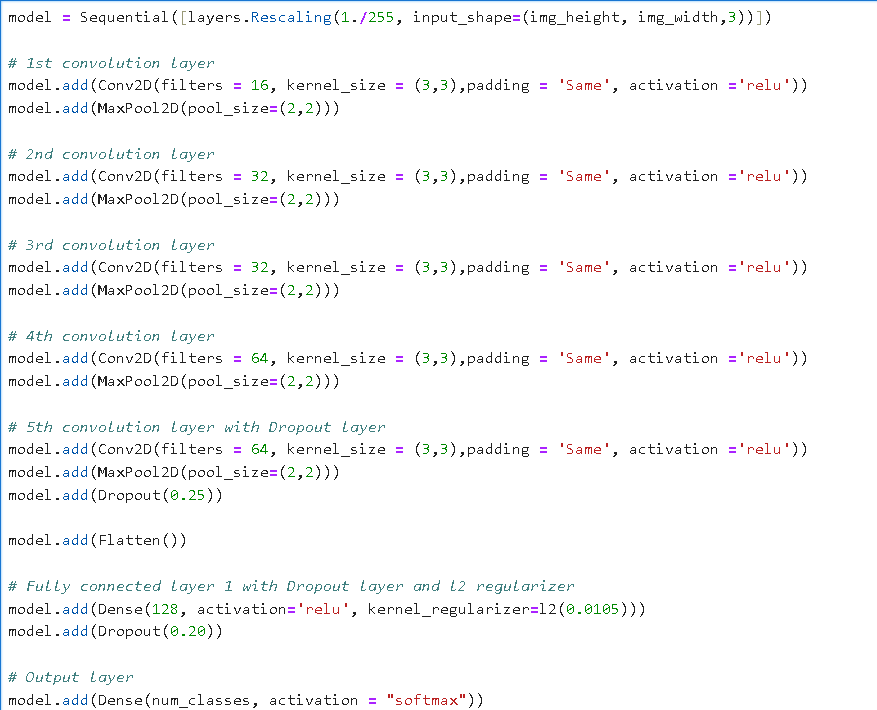




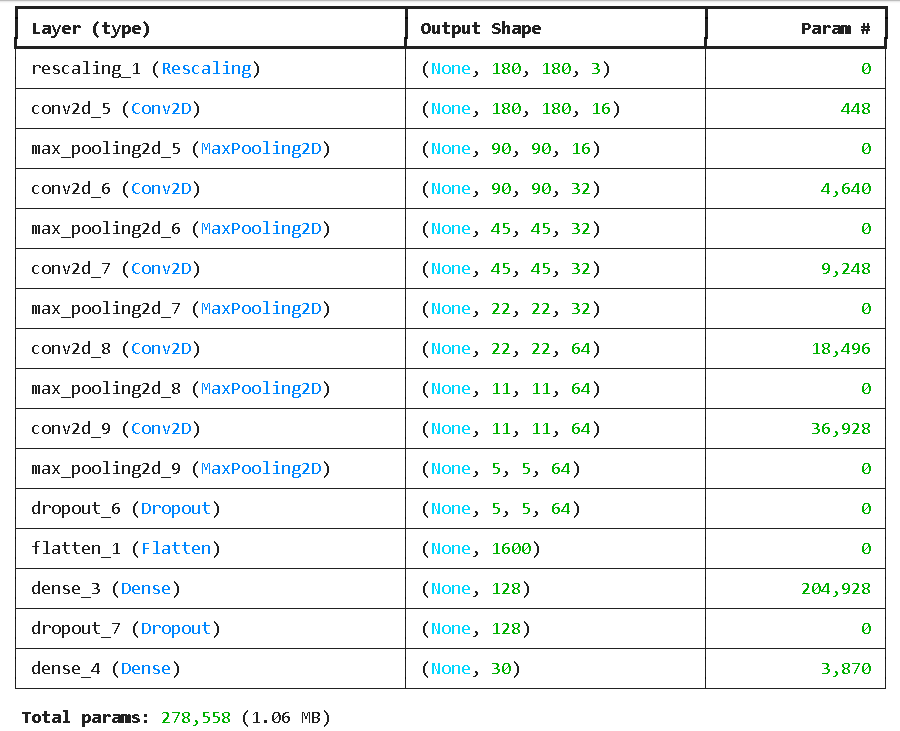




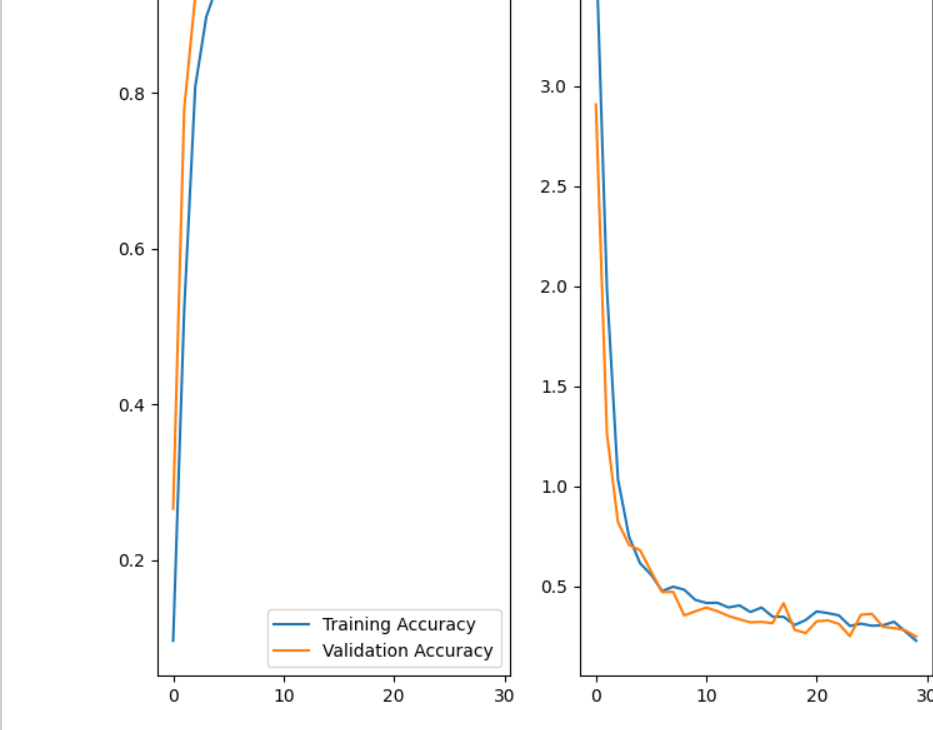
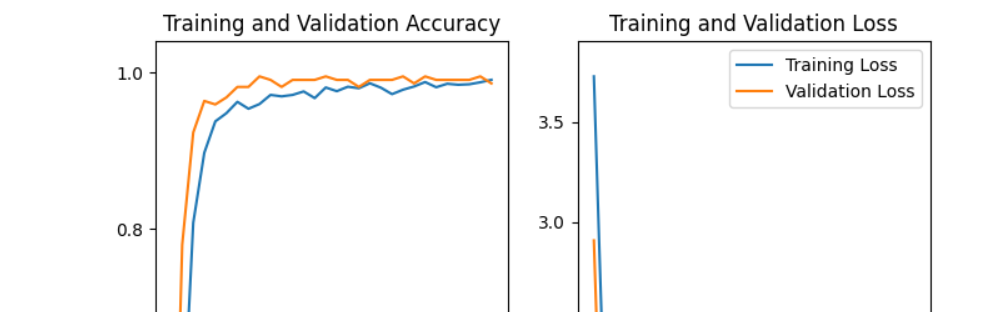
**Model:**

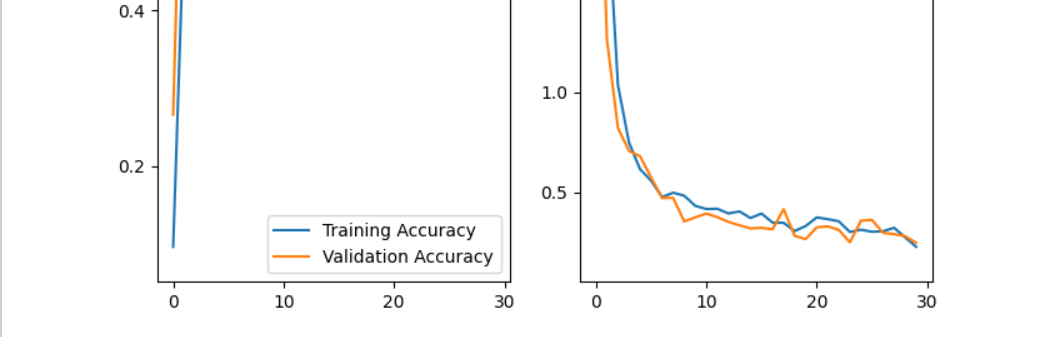


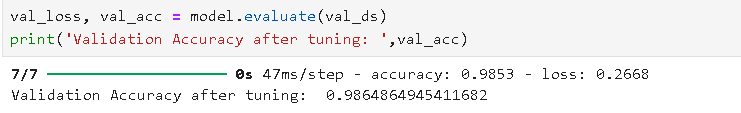
**Model Summary:**



**Results:**







**7. Conclusion & Insights**

This project successfully implemented a CNN for road sign classification using data augmentation and tuning techniques. Although tuning improved model performance, misclassification still occurred in complex cases.