**Deliverable #2**

Karl Satchi E. Navida  
Deep Learning - MSCS23A  
April 10, 2025

1. **Dataset**

CIFAR-10 is a dataset composed of 60k images divided into five (5) batches and a test batch. It came from **Papers with Code** website and was downloaded directly from the CS Toronto Edu – CIFAR-10 website.

As previously stated, the CIFAR-10 dataset consists of 60k, 32x32 color images in 10 classes, with each class containing 6k images. The five batches contain a combined 50k images, with the 10k images under the test dataset. And under the test dataset, the 10k images is equally divided into 1k images per class.

1. **Classification Results**

The results of this deliverable will be separated into five (5) sections, documenting each configuration used until the highest accuracy was attained. These configurations are only ran once and thus, may as well just be an approximation of accuracy around the provided one in this results.

* 1. **Initial**

The **initial** configuration is the very first configuration used on its initial run and test. This configuration only has/uses three (3) Conv2D layers, with each Conv2D layer followed by a MaxPooling2D layer. In the hidden layer, Dropout layer was still not use in this stage.

The steps\_per\_epoch (SPE) was also set to 100 in this run while the validation\_steps (VS) was set to 50, leaving a lot of data unused during the fitting of the model.

Figure 1 shows a simple model diagram of the created neural network for this deliverable.

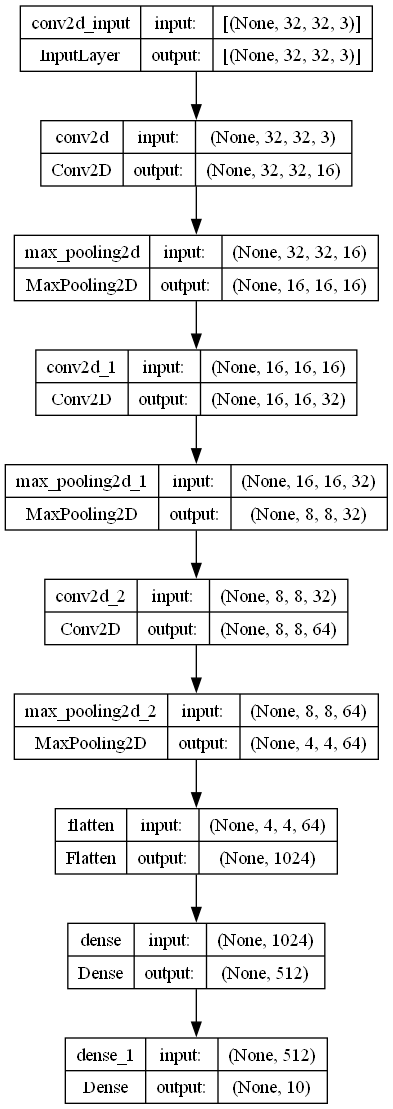
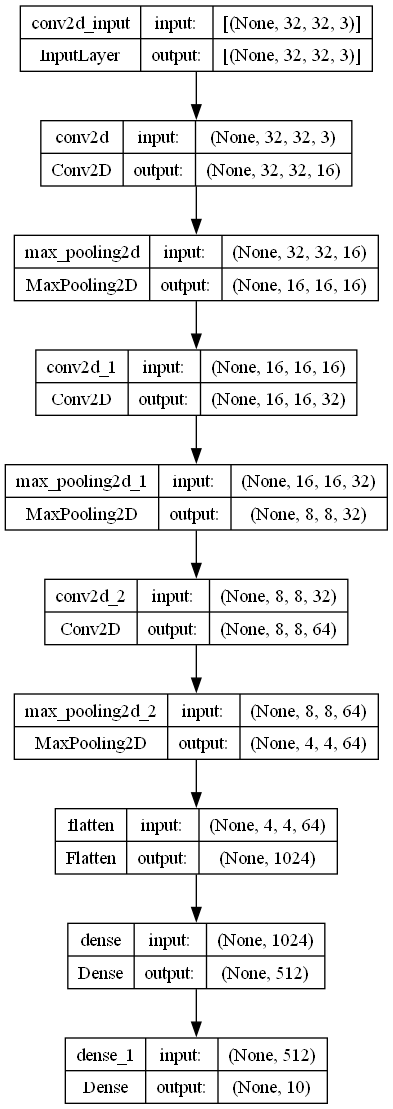


Figure : Model for "initial" configuration.

This configuration managed to reach only 59.09% accuracy as seen in Figures 2 and 3. This shows the without any regularization such as Dropout or BatchNormalization, the training will be less accurate due to some of the neurons having to learn broad features or the values not being normalized.

A graph with red and blue lines

Description automatically generated

Figure : History Plot based on its fit() method.

A graph with numbers and a number in the center

Description automatically generated with medium confidence

Figure : The confusion matrix of the "initial" model confriguration, showing the amount of correct prediction along the diagonal line starting from the top left.

* 1. **Training Steps Modification (No Dropout & Batch Normalization)**

The second configuration tested; it was after the initial configuration was modified lightly so that the SPE and VS utilizes the training data to its maximum. By doing this, the following formula was used:

Whereas:

* is the amount of train data (in this case, 40k for the training data and 10k for the validation data)
* is the batch size (in this case, 32 for both the training and validation data)

Following the same model configuration, this configuration resulted to a much higher accuracy and output as seen in Figures 4 and 5.

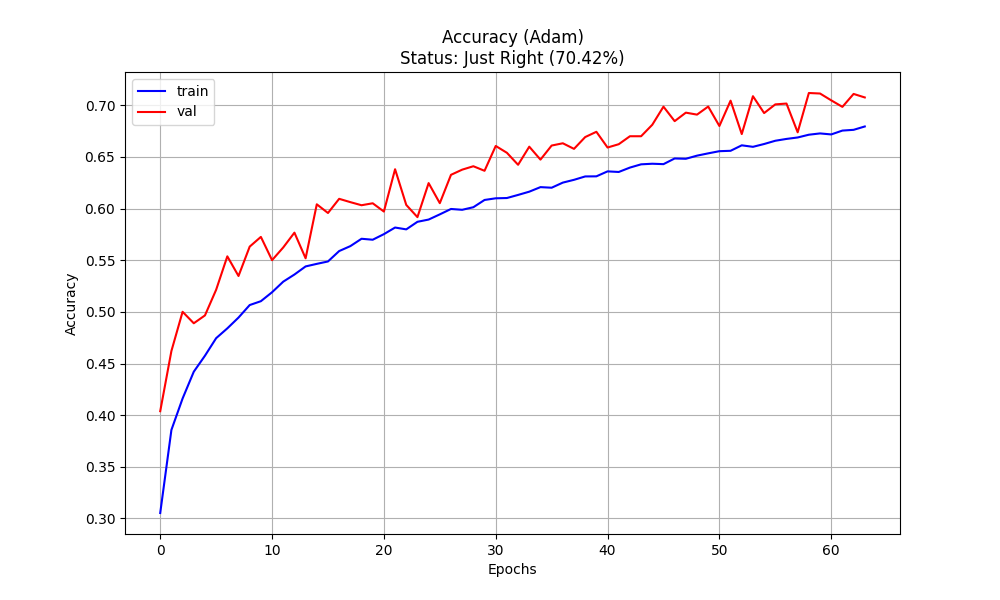


Figure : History Plot based on its fit() method.

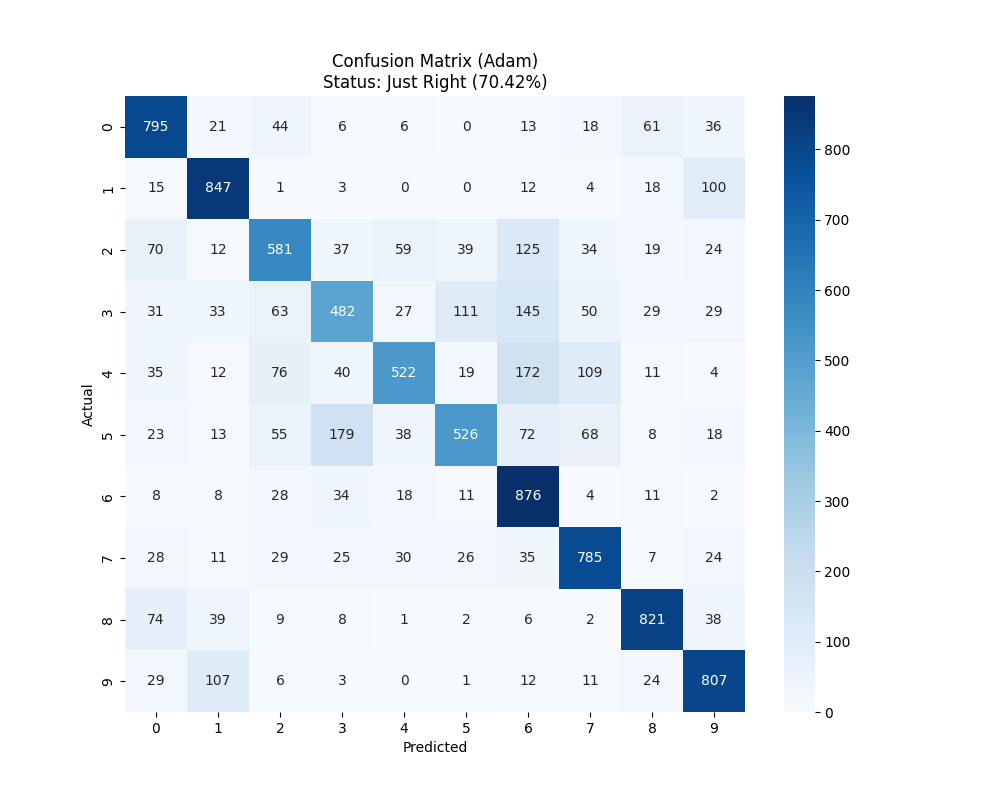


Figure : The confusion matrix of the second model configuration, showing the amount of correct prediction along the diagonal line starting from the top left.

This alone made a significance leap in improvement, showing a 28.87% increase in accuracy, making it jump up to 70.42% from the previous 50.09% accuracy.

* 1. **Introduction of Dropout and Batch Normalization (No Reduced Learning Rate)**

This third configuration introduces the Dropout and BatchNormalization layers, resulting to a further small increase in accuracy. This new architecture introduces a BatchNormalization layer after every Conv2D layer, normalizing its values before feeding it to the MaxPooling2D layer. Finally, the Dropout layer is placed just before the output layer, which has a 40% rate. Figure 6 shows the updated model diagram while Figures 7 and 8 shows the training history and accuracy.

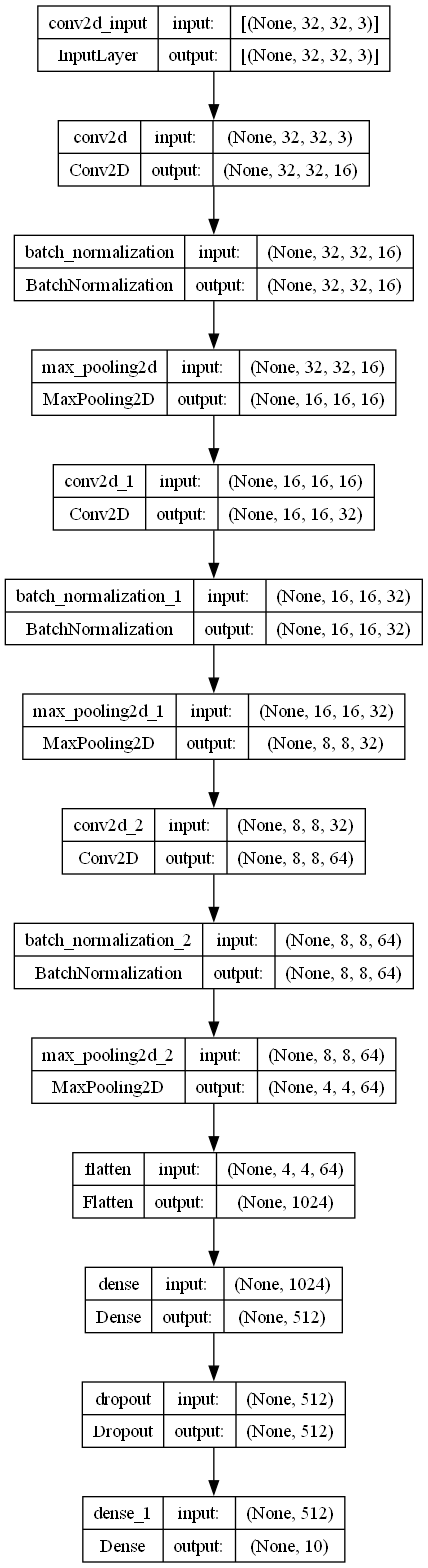
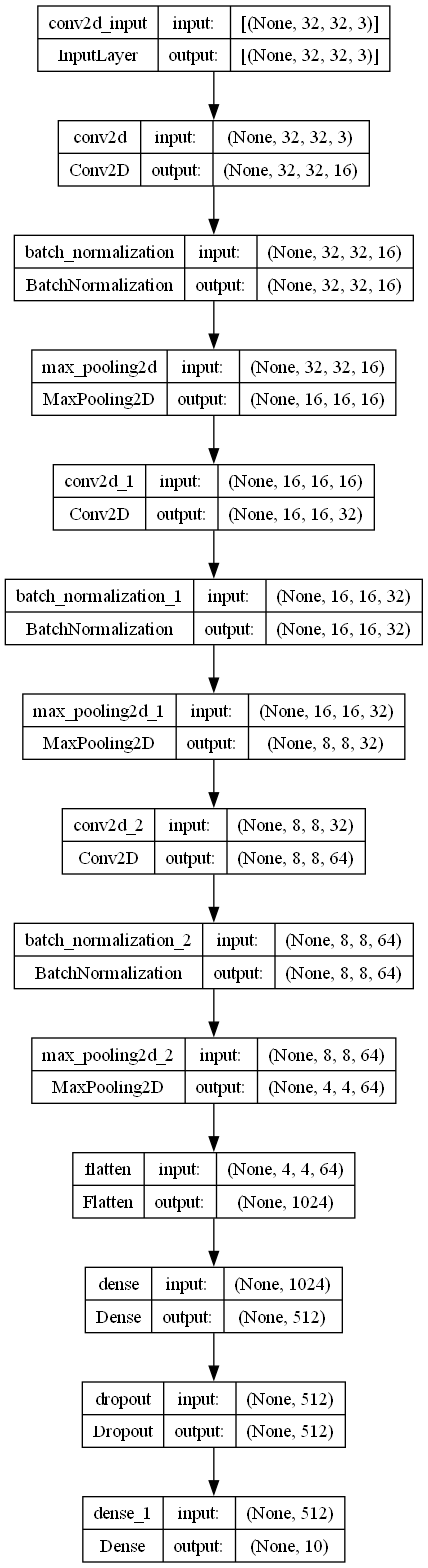


Figure : Model diagram after adding the two new regularization layers.

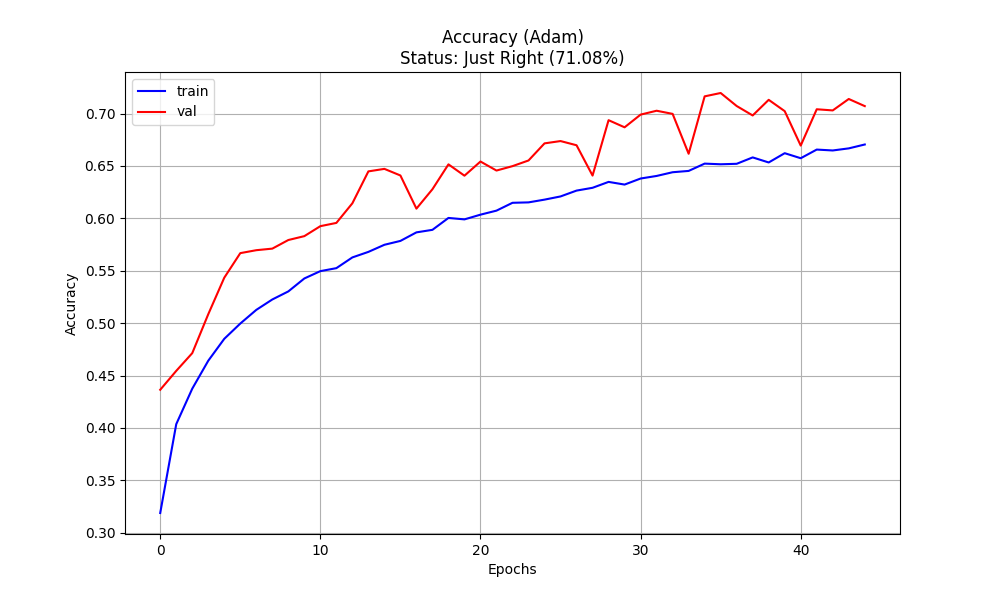


Figure : History Plot based on its fit() method.

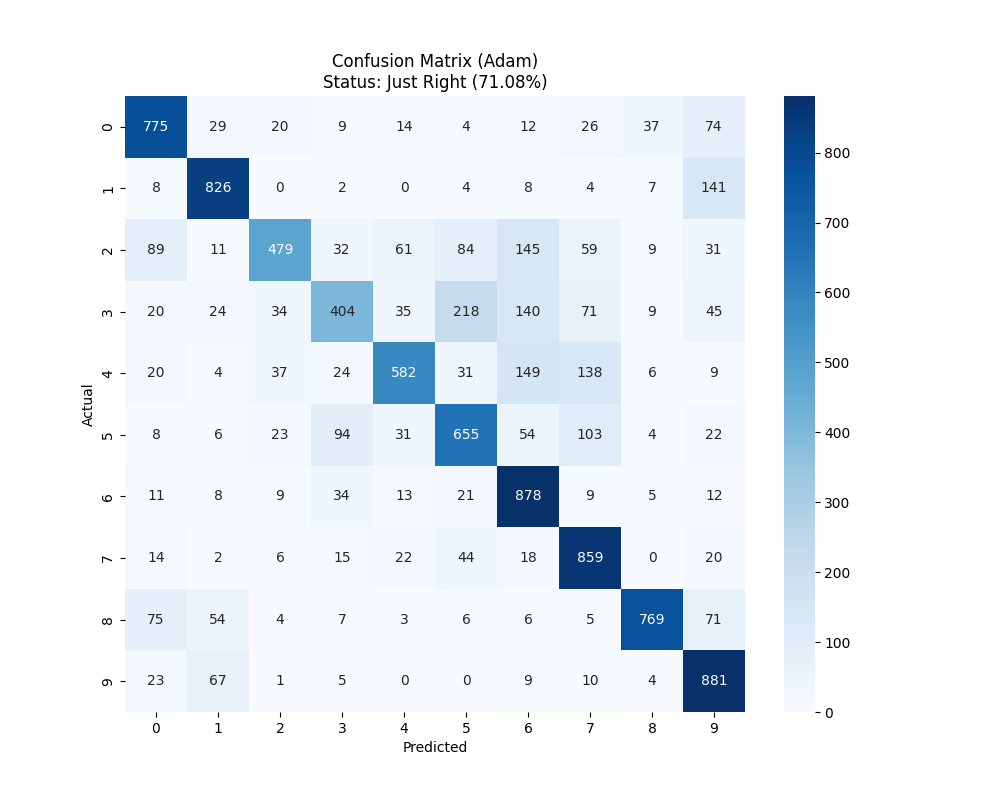


Figure : The confusion matrix of the third model configuration, showing the amount of correct prediction along the diagonal line starting from the top left.

* 1. **Addition of Callbacks (Three Convolutional 2D Layer)**

This configuration introduces callbacks to the fit() method; adding both the EarlyStopping and ReduceLROnPlateau callbacks. Both callbacks monitor the val\_loss metrics with a 10 and 5 patience, respectively.

The EarlyStopping callback was set higher than the ReduceLROnPlateau to allow the model to have a wiggle room before the former callback kicks in. Furthermore, the ReduceLROnPlateau has its factor set to 50% (0.5) and its minimum learning rate to 1e-6 (0.0000001).

This application provided minimal increase in accuracy, but the result may just be its minimum in its range.

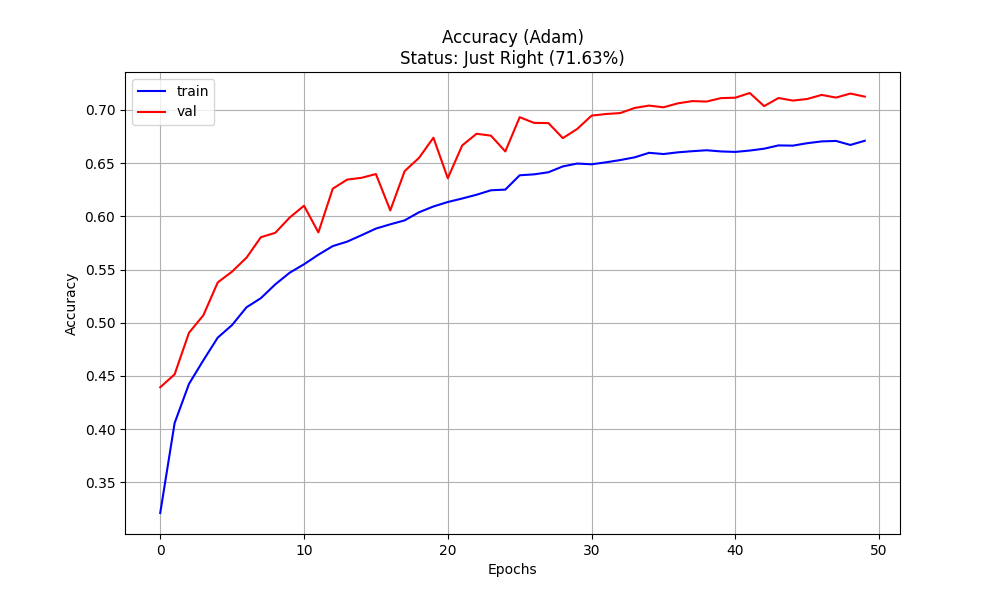


Figure : History Plot based on its fit() method.

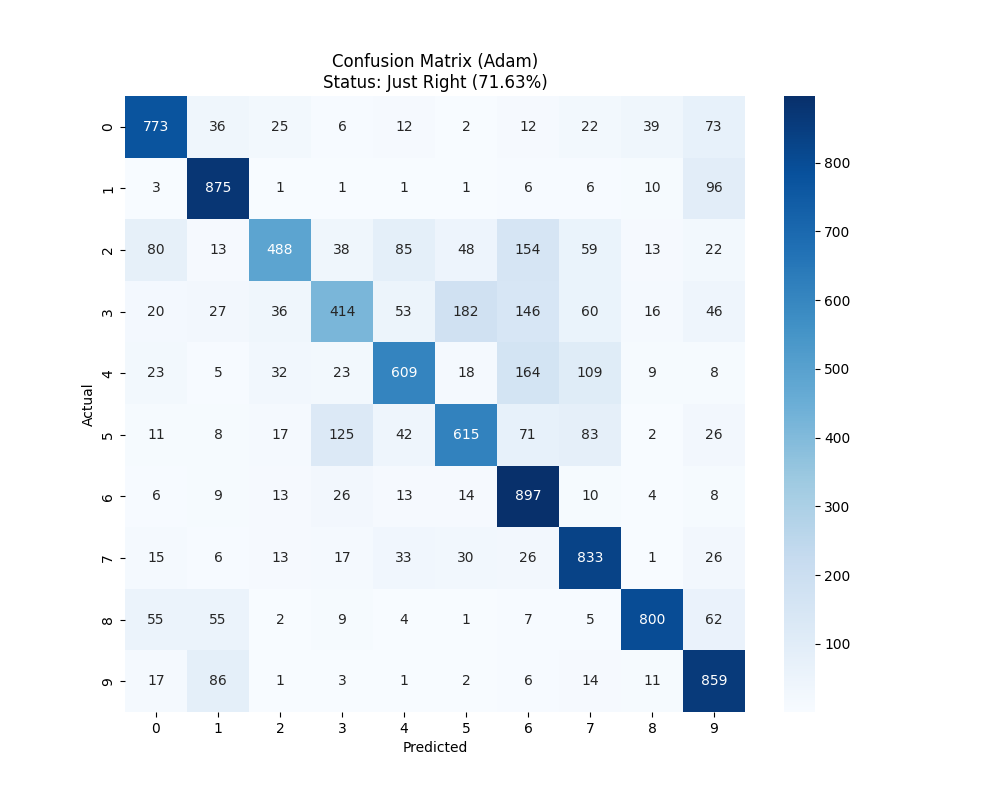


Figure : The confusion matrix of the fourth model configuration, showing the amount of correct prediction along the diagonal line starting from the top left.

* 1. **Final (Original)**

The final configuration has the highest accuracy of them all. Simply due to the addition of a fourth Conv2D layer with its BatchNormalization and MaxPooling2D layers.

The updated model diagram in Figure 11 shows the changes made to the model’s architecture.

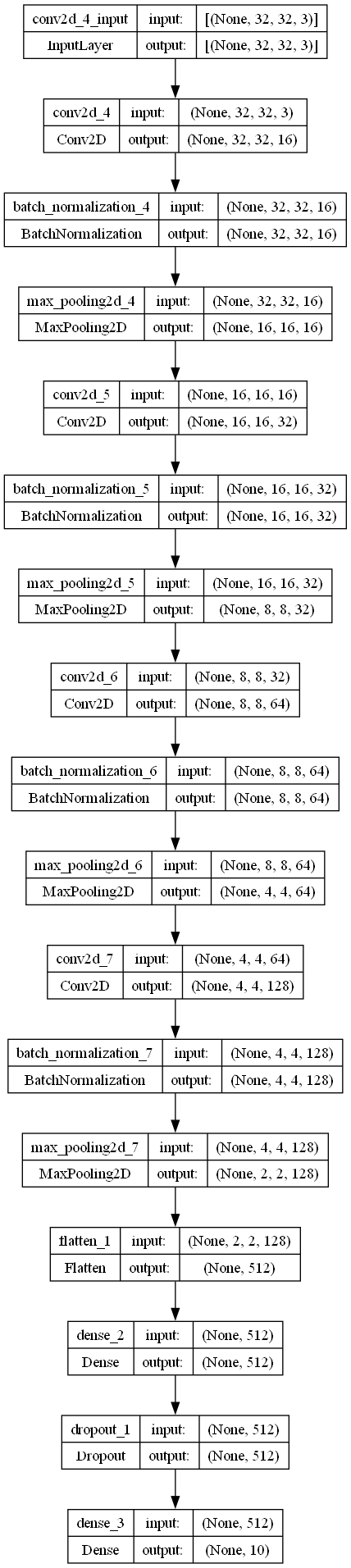
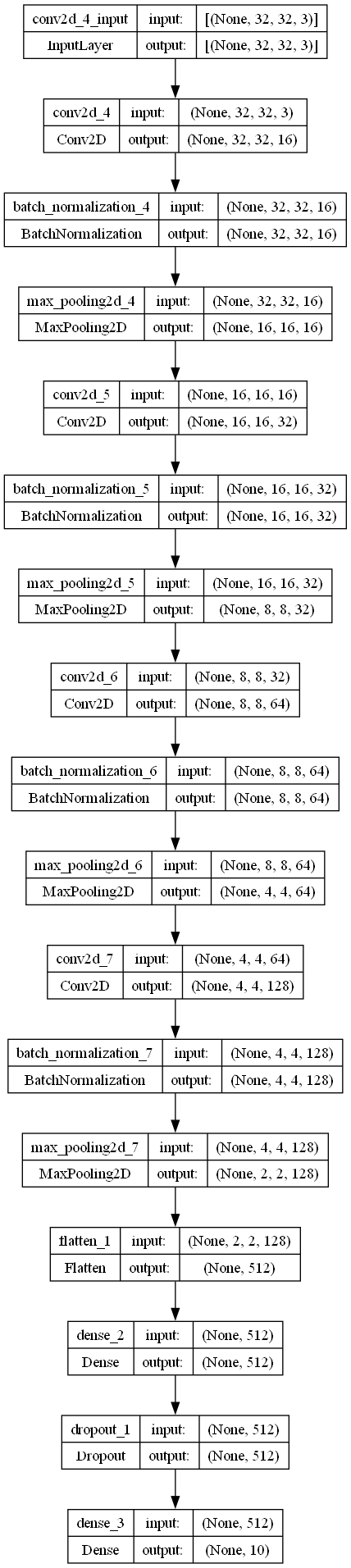


Figure : Model diagram after adding the new layers.

With this new configuration, the model’s accuracy jumped a measly 6.62% from its previous 71.63%, returning a 76.71% accuracy with this modification.

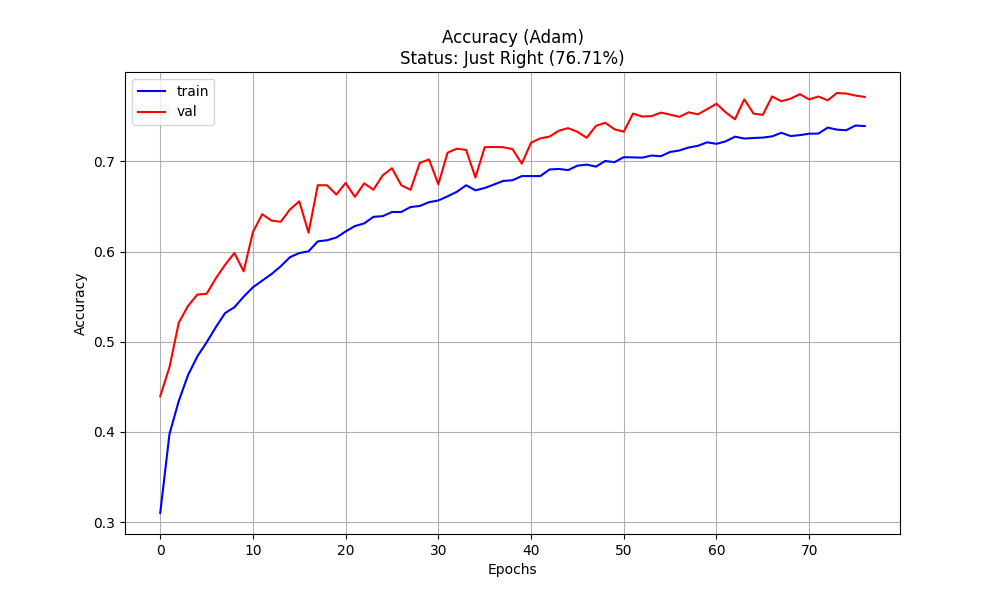


Figure : History Plot based on its fit() method.

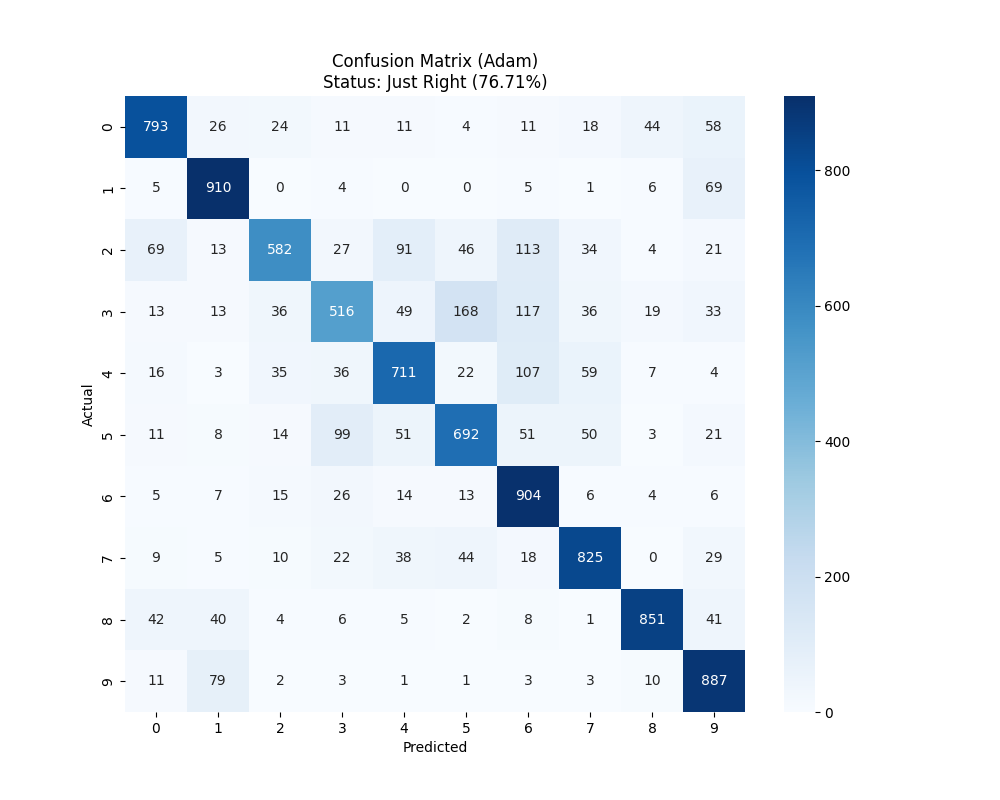


Figure : The confusion matrix of the fifth and last model configuration, showing the amount of correct prediction along the diagonal line starting from the top left.