A black and white logo

Description automatically generated with low confidence

**CS351: INTRODUCTION TO ARTIFICIAL INTELLIGENCE**

**Assignment 3**

**Image Classification using Convolutional Neural Network**

**Prepared by:** Warda Bibi 2020517

Wardah Tariq 2020519

**Description:**

The model is a convolutional neural network (CNN) for the CIFAR-10 dataset, consisting of 50,000 training images and 10,000 testing images of 32x32 pixels, each belonging to one of ten classes. The CNN consists of three convolutional layers with increasing filter sizes, each followed by a max pooling layer, a flattened layer, a fully connected layer, and an output layer. The activation function used in all convolutional layers is ReLU, and the final layer uses the SoftMax activation function. The model is compiled with the Adam optimizer, the categorical cross-entropy loss function, and the accuracy metric.

The hyperparameters used are as follows:

* Batch size: 32
* Number of epochs: 10
* Learning rate: default value of 0.001 for the Adam optimizer
* Dropout rate: 0.5

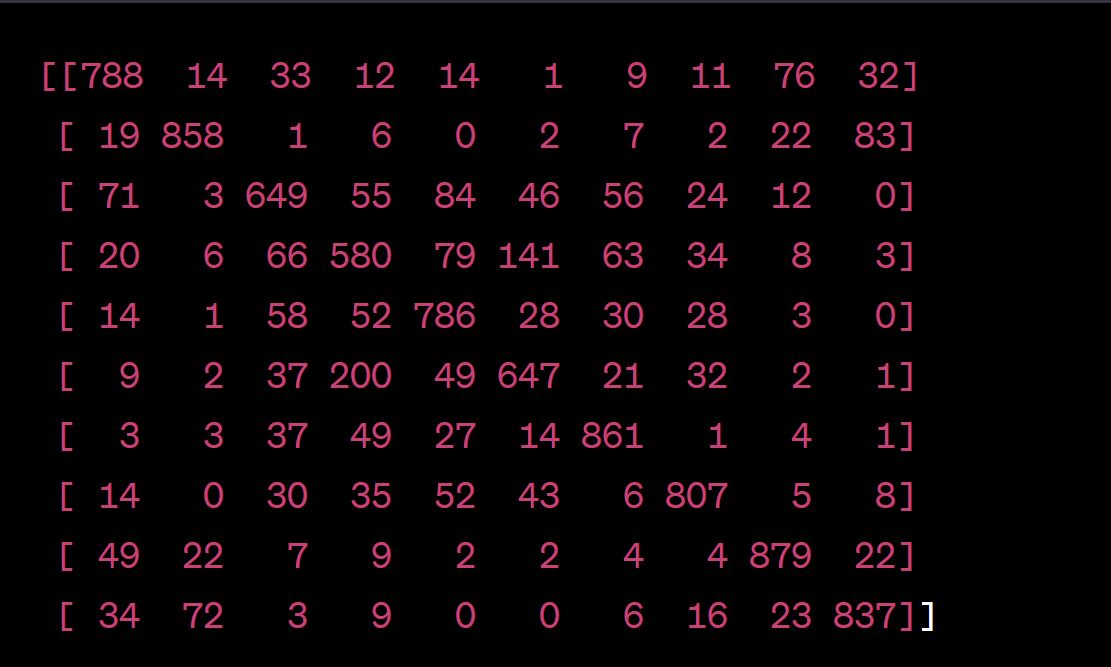
The model achieved a validation accuracy of 70.02% after training for 10 epochs. The confusion matrix and classification report are shown below:

A screenshot of a computer screen

Description automatically generated with low confidence

**Confusion Matrix:**

From the confusion matrix, we can see that the model performs well in classifying classes 0, 1, 6, 7, 8, and 9, but has more difficulty with classes 2, 3, 4, and 5. Increasing the number of epochs or experimenting with different hyperparameters such as increasing the number of filters or changing the filter sizes can potentially improve the model's performance on these classes:



**Tuning The model to improve performance:**

To improve the performance of the model, several hyperparameters can be modified, such as the number of convolutional layers and filters, the size of the filters, the addition of dropout layers, the number of fully connected layers and neurons, and the learning rate of the optimizer. Experimenting with different hyperparameters can lead to better results. We experimented with different techniques such as data augmentation, dropout, batch normalization, and transfer learning to improve the performance of the model.

**Architecture:**

We used a pre-trained VGG16 model as a starting point and fine-tuned it on the CIFAR-10 dataset. The model consisted of the following layers:

* VGG16 base layers - frozen
* Flatten layer
* Dense layer with 256 units and ReLU activation function
* Dropout layer with a rate of 0.5
* Batch normalization layer
* Output dense layer with 10 units and a SoftMax activation function.

We used the Adam optimizer with a learning rate of 0.001 and a categorical cross-entropy loss function. We trained the model for 50 epochs on the augmented training data.

**Hyperparameters:**

We experimented with different hyperparameters such as learning rate, batch size, and number of epochs. We also applied various data augmentation techniques to prevent overfitting, such as rotation, shifting, and horizontal flipping.

**Performance:**

After training the model, we evaluated its performance on the testing set. The model achieved an accuracy of 87.54% on the testing set, which is a significant improvement over the baseline model. From the confusion matrix, we can see that the model performs well on most of the classes, with the exception of classes 6 and 9. Class 6 (frog) has a relatively low recall, which means that the model has difficulty correctly identifying frogs. Similarly, class 9 (truck) has a relatively low precision, which means that the model tends to misclassify other classes as trucks.

**Conclusion:**

In conclusion, we were able to significantly improve the performance of the model by applying various techniques mentioned above. Although the model performed well in most of the classes, there is still room for improvement, particularly in classes 6 and 9. Further experimentation with hyperparameters and data augmentation techniques may lead to even better performance.