Ankita Vashisht

Building the first Convolutional Neural Network (CNN) model.

A screenshot of a computer

Description automatically generated

Import the necessary libraries-

1. os: Operating system operations.
2. numpy: Numerical operations and array handling.
3. shuffle from sklearn.utils: Used to shuffle data.
4. ImageDataGenerator from tensorflow.keras.preprocessing.image: Helps with image data augmentation.
5. layers, models, optimizers from keras: Components for defining and configuring neural networks.
6. to\_categorical from tensorflow.keras.utils: Converts class vectors (integers) to binary class matrices.
7. callbacks from keras: Provides callbacks for training.

A screenshot of a computer

Description automatically generated

Defines class names and creates a dictionary class\_labels where each class name is associated with a numerical label.

A screenshot of a computer

Description automatically generated

Both training and testing datasets (train\_images, test\_images, train\_labels, test\_labels) are shuffled using shuffle() to randomize the order of data.

Images are normalized by dividing pixel values by 255 to scale them between 0 and 1 (trn\_images\_norm, tst\_images\_norm).

A screenshot of a computer

Description automatically generated

Convolutional layers (layers.Conv2D) with ReLU activation, followed by max pooling (layers.MaxPooling2D).

Flattening layer (layers.Flatten) to convert 2D feature maps into a 1D vector.

Dense layers (layers.Dense) for classification, with ReLU activation in the hidden layer and softmax activation in the output layer.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

convert integer class labels to categorical matrices using to\_categorical.

A screenshot of a computer

Description automatically generated

EarlyStopping: Stops training when a monitored metric (val\_loss) has stopped improving.

ModelCheckpoint: Saves the model ('bestvalue.h5') after every epoch if it's the best according to val\_loss.

A screenshot of a computer

Description automatically generated

trn\_images\_norm and train\_labels are used as training data.

epochs=5 specifies the number of training epochs.

batch\_size=32 sets the number of samples per gradient update.

validation\_data=(tst\_images\_norm, test\_labels) provides validation data for evaluation.

callbacks=callbacks\_list applies the defined callbacks during training.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Save the trained model to 'image\_classifier\_cnn1.h5' in HDF5 format for later use.