**Name:** Suyog Bodke  
**Roll No:** 381006  
**PRN:** 22310072

**Assignment No: -** 3  
**Title: -** Image Classification using Convolutional Neural Networks (CNNs)

**Problem Statement:**

Implement image classification using convolutional neural networks (CNNs) for **multiclass classification**.

**Objective:**

* To learn the implementation of CNNs for image recognition.
* To apply data preprocessing and augmentation for improving performance.
* To train a deep CNN on CIFAR-10 dataset (10 classes).
* To evaluate model performance using accuracy, precision, and recall.
* To test the trained model on new user-uploaded images.

**S/W Packages and H/W apparatus used:**

* **Operating System:** Windows/Linux/MacOS
* **Kernel:** Python 3.x
* **Tools:** Jupyter Notebook / Google Colab
* **Hardware:** GPU (optional, for faster training)
* **Libraries:** TensorFlow, Keras, NumPy, Matplotlib, scikit-learn, OpenCV

**Theory:**

A **Convolutional Neural Network (CNN)** is a deep learning architecture widely used for image recognition tasks.

Key components:

* **Convolutional Layers:** Extract spatial features using kernels.
* **Batch Normalization:** Normalizes activations, speeds up convergence.
* **Pooling Layers:** Down-sample feature maps, reduce computation.
* **Dropout:** Prevents overfitting by randomly disabling neurons.
* **Fully Connected Layers:** Perform classification based on extracted features.
* **Softmax Activation:** Outputs class probabilities for multiclass classification.

In this assignment, CIFAR-10 dataset (60,000 images of 10 classes such as airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck) is used.

**Methodology:**

1. **Dataset Loading:** CIFAR-10 with 50,000 training and 10,000 test images.
2. **Preprocessing:** Normalize pixel values (0–1) and one-hot encode labels.
3. **Data Augmentation:** Apply rotation, shift, and horizontal flip.
4. **Model Design:**
   * Conv Block 1: Conv2D(32), BatchNorm, MaxPooling, Dropout
   * Conv Block 2: Conv2D(64), BatchNorm, MaxPooling, Dropout
   * Conv Block 3: Conv2D(128), BatchNorm, MaxPooling, Dropout
   * Dense(128), Dropout, Dense(10, softmax)
5. **Compilation:** Adam optimizer, categorical crossentropy loss, accuracy metric.
6. **Training:** 10 epochs with augmented data.
7. **Evaluation:** Accuracy, Precision, Recall on test data.
8. **Prediction:** Tested with custom uploaded images.

**Results:**

* **Training Accuracy:** Improved steadily across epochs.
* **Final Test Accuracy:** **76%**
* **Precision (Macro):** **0.79**
* **Recall (Macro):** **0.76**

**Advantages:**

* CNN automatically learns relevant features from images.
* Augmentation improves generalization.
* Works well for multiclass image recognition tasks.

**Limitations:**

* Training is computationally expensive.
* Accuracy depends heavily on hyperparameters and architecture depth.
* CNNs require large datasets for best performance.

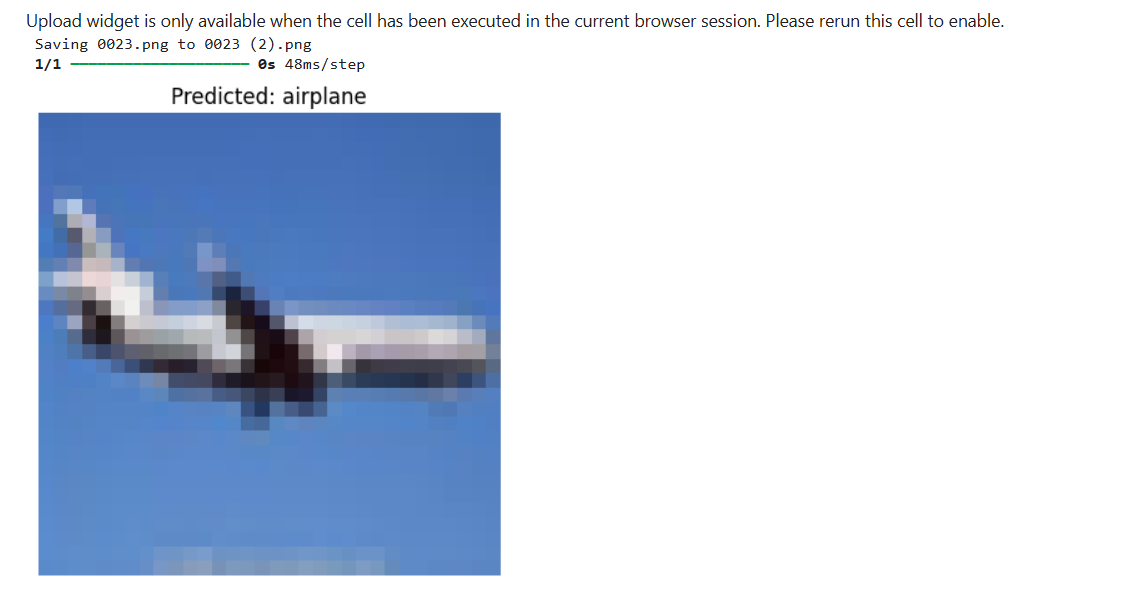
**Applications:**

* Object detection and recognition.
* Autonomous driving (traffic sign and vehicle recognition).
* Medical imaging (disease detection).
* Security systems and surveillance.

**Working / Algorithm:**

1. Load CIFAR-10 dataset.
2. Normalize and preprocess labels.
3. Apply augmentation to training set.
4. Build CNN with convolution, pooling, dropout, and dense layers.
5. Compile and train using Adam optimizer.
6. Evaluate on test dataset.
7. Test with custom image uploads.

**Diagram:**

****

**Conclusion:**

The CNN model successfully classified images from the CIFAR-10 dataset into 10 categories with a test accuracy of **76%**. Precision and recall values indicate good overall performance. Data augmentation played a major role in improving generalization. The experiment highlights the power of CNNs for multiclass image classification tasks.