**EXPERIMENT 5**

**Aim: Design and implement to classify 32x32 images using MLP using tensorflow/keras and check the accuracy**

**Description:**

**Dataset used:**

The **CIFAR-10** dataset consists of 60,000 32x32 color images in 10 classes, with 6,000 images per class. The 10 different classes represent various objects such as airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks.

**Key Characteristics:**

* **Image Size:** 32x32 pixels
* **Color Channels:** 3 (RGB)
* **Number of Classes:** 10
* **Total Images:** 60,000

The program implements an **MLP (Multi-Layer Perceptron)** model to classify CIFAR-10 images. Unlike convolutional neural networks (CNNs), which are designed for spatial data, the MLP treats the image as a flat vector. The steps include:

1. **Flattening the Image**: Reshaping the 32x32x3 images into 1D arrays of size 3072 (32×32×3).
2. **Building the Model**: An MLP with multiple dense layers and activation functions.
3. **Training and Evaluation**: Using the CIFAR-10 dataset to train the MLP and evaluate its accuracy on the test set.

**Functions used:**

Sequential: Used to build the model layer by layer in a linear stack.

Flatten: Reshapes the 3D image array (32x32x3) into a 1D vector (3072), enabling the fully connected layers to process the data.

Dense: Fully connected layers with specified neurons and activation functions.

Activation functions used:

ReLU (Rectified Linear Unit): Helps to learn complex patterns by introducing non-linearity.

Softmax: Converts raw scores into probabilities for multiclass classification.

compile: Configures the model for training.

Parameters:

Optimizer: Adam is chosen for its adaptive learning rate.

* Loss Function: Sparse categorical cross-entropy, suitable for integer-labeled multiclass classification.

Metrics: Accuracy to monitor performance.

fit: Trains the model on the training data for a given number of epochs and batch size.

evaluate: Tests the model on unseen data and calculates the loss and accuracy.

plot: Visualizes training and validation accuracy over epochs to assess the model's performance.

#### ****Model Key Layers****:

**Flatten**: Converts the 3D input (32x32x3) into a 1D vector (3072 elements) to be used with fully connected layers.

**Dense**: Fully connected layers with varying numbers of neurons (512, 256, 128).

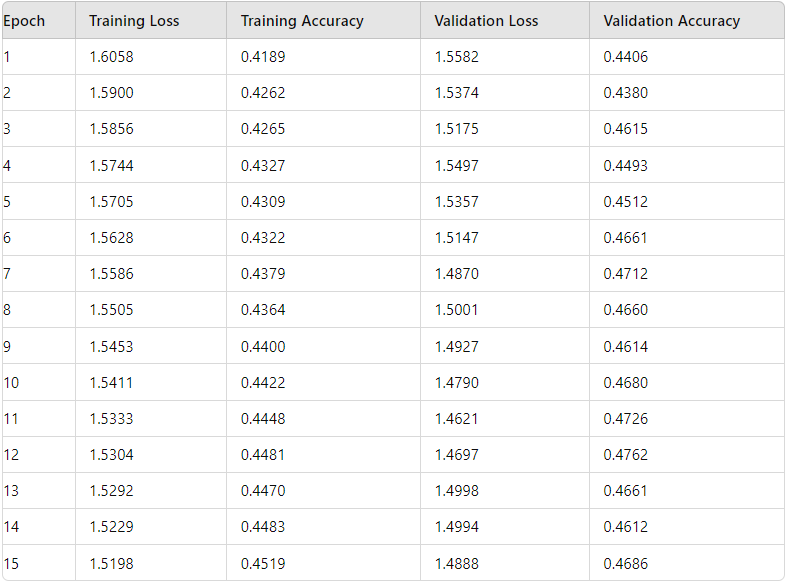
**Activation:** **ReLU (Rectified Linear Unit)** for non-linear transformations.

**Dropout**: A regularization technique to reduce overfitting by randomly disabling 20% of neurons during training.

**Output Layer**:

Dense(10, activation='softmax'): 10 neurons for the 10 classes, with probabilities for each class.

**Result Analysis:**



**Key Observation:**

**Accuracy**:

Training accuracy starts around \_\_\_\_\_\_\_ and improves slightly, reaching \_\_\_\_% by the 15th epoch.

Validation accuracy also improves moderately to **~\_\_\_\_\_%**.

**Loss**:

Training and validation loss show \_\_\_\_\_\_\_\_\_\_\_\_, indicating convergence.

Validation loss remains higher than training loss, suggesting slight \_\_\_\_\_\_\_\_\_.

**Limitations**:

MLP lacks \_\_\_\_\_\_\_\_\_\_\_\_, making it less effective for image data like CIFAR-10 compared to CNNs.

Dropout helps reduce overfitting, but performance is still constrained by the architecture.

**Improvements**:

Replacing MLP with CNNs could significantly improve performance by capturing spatial hierarchies.

Data augmentation (e.g., flipping, rotation) can improve generalization.