**Experiment No:2**

**Title:** Develop a CNN to classify images from the CIFAR-10 dataset. Experiment with different architectures and hyperparameters to achieve the highest accuracy possible.

The CIFAR-10 small photo classification problem is a standard dataset used in computer vision and deep learning.

Although the dataset is effectively solved, it can be used as the basis for learning and practicing how to develop, evaluate, and use convolutional deep learning neural networks for image classification from scratch.

Building a Convolutional Neural Network (CNN) for image classification on the CIFAR-10 dataset involves experimenting with different architectures and hyperparameters. CIFAR-10 consists of 60,000 32x32 color images in 10 classes, with 6,000 images per class. Here's a step-by-step guide to create and optimize a CNN for this task using TensorFlow and Keras:

### **Step 1: Load and Preprocess Data**

First, import necessary libraries and load the CIFAR-10 dataset:

A screen shot of a computer

Description automatically generatedA computer screen shot of numbers and symbols

Description automatically generated

**Step 2: Define the CNN Architecture**

Next, define your CNN architecture. Here’s a basic architecture to start with:

A computer screen shot of white and green text

Description automatically generated

**Step 3: Compile the Model:**

Compile the model with an appropriate optimizer, loss function, and metrics:

A black screen with green and blue text

Description automatically generated

### **Step 4: Train the Model**

Train the model on the training data. Experiment with different batch sizes, number of

epochs, and early stopping to avoid overfitting:

A computer screen with white text

Description automatically generated

**Step 5: Evaluate the Model**

Evaluate the model on the test data and check the performance:

A black screen with white text

Description automatically generated

### **Step 6: Experiment with Architecture and Hyperparameters**

To achieve higher accuracy, consider these modifications and experiments:

* **Increase depth or width of the network**: Add more convolutional layers or increase the number of filters in each layer.
* **Adjust filter sizes**: Experiment with different filter sizes (e.g., 5x5, 7x7) in convolutional layers.
* **Use different activation functions**: Try alternatives like Leaky ReLU, ELU, or SELU.
* **Regularization**: Add more dropout layers or try other regularization techniques like L2 regularization.
* **Optimization parameters**: Adjust learning rate, optimizer (e.g., SGD, RMSprop), and batch size.
* **Data augmentation**: Apply data augmentation techniques to increase the diversity of training data (e.g., rotation, flips, shifts).

### **Example Advanced Architecture**

Here’s an example of a more advanced architecture you can try:

A screen shot of a computer program

Description automatically generatedA computer screen with white text

Description automatically generated

### **Conclusion:**

### Experimentation is key to finding the best CNN architecture and hyperparameters for your specific problem. Use the above steps as a starting point and adjust based on your observations from training and evaluation results.