**Assignment No: 3**

**[Image Classification using CNNs]**

**Name: Snehal Yelwande**

**Batch: TY-A A3**

**PRN: 22311760**

**Problem Statement:**

Implement Image classification using Convolutional Neural Networks (CNNs) for multiclass classification.

**Objectives:**

* To understand the architecture and working of Convolutional Neural Networks.
* To learn how to preprocess image data for training CNNs.
* To implement a CNN model using Keras and TensorFlow for multiclass classification.
* To evaluate model performance using validation data.
* To visualize training accuracy and loss over epochs.

**S/W Packages and H/W Apparatus Used:**

* Operating System: Windows/Linux/MacOS
* Kernel: Python 3.x
* Tools: Jupyter Notebook, Anaconda, or Google Colab
* Hardware: CPU with minimum 4GB RAM; optional GPU for faster processing

**Libraries and Packages Used:**

* TensorFlow
* Keras
* NumPy
* Matplotlib

**Theory:**

A Convolutional Neural Network (CNN) is a deep learning algorithm primarily used for processing structured grid data, such as images. CNNs automatically detect features and patterns in images, making them highly effective for image classification tasks.

**Structure of CNN:**

1. Input Layer: Receives input images.
2. Convolutional Layers: Apply convolution operations to extract features from images. Filters learn to recognize edges, textures, and shapes.
3. Pooling Layers: Reduce the spatial dimensions of feature maps, retaining the most essential information while reducing computation.
4. Fully Connected Layers: Connect every neuron in one layer to every neuron in the next layer, leading to the output.
5. Output Layer: Produces class probabilities for the input images**.**

**Activation Functions:**

* ReLU: Introduces non-linearity and prevents vanishing gradients.
* Softmax: Converts outputs into class probabilities for multiclass classification.

**Training with Backpropagation:**

CNNs use backpropagation to update weights and minimize loss through optimizers like Adam.

**Methodology:**

1. Data Acquisition
   * Load the CIFAR-10 dataset containing 60,000 images across 10 classes.
2. Data Preparation
   * Normalize pixel values between 0 and 1 to speed up convergence.
3. Model Architecture
   * Create a Sequential model using Keras.
   * Add Convolutional + Pooling layers:
     + Conv2D (32 filters, 3×3 kernel, ReLU) + MaxPooling2D (2×2).
     + Conv2D (64 filters, 3×3 kernel, ReLU) + MaxPooling2D (2×2).
     + Conv2D (64 filters, 3×3 kernel, ReLU).
   * Flatten → Dense(64, ReLU) → Dense(10, Softmax).
4. **Model Compilation**
   * Optimizer: Adam
   * Loss Function: Sparse Categorical Crossentropy
   * Metric: Accuracy
5. **Model Training**
   * Train for 10 epochs with batch size = 128.
   * Validate using test dataset.
6. **Model Evaluation**
   * Evaluate performance on test set (accuracy & loss).
7. **Visualization**
   * Plot training & validation accuracy and loss.

**Advantages:**

* Feature Extraction: Learns features automatically, no manual engineering.
* Translation Invariance: Handles image shifts and distortions.
* Reduced Parameters: Fewer parameters than dense networks.
* Hierarchical Learning: Learns from edges → textures → shapes → objects.

**Limitations:**

* Requires large labeled datasets.
* Computationally expensive (needs GPU for large datasets).
* Risk of overfitting on small datasets.
* Sensitive to hyperparameters (filter size, layers, etc.).

**Applications:**

* Medical Imaging – Disease detection (X-rays, MRIs).
* Autonomous Vehicles – Object recognition.
* Security – Facial recognition.
* Retail – Product categorization.

**Working / Algorithm:**

1. Load CIFAR-10 dataset.
2. Normalize pixel values.
3. Visualize sample images.
4. Define CNN with Conv → Pool → Dense layers.
5. Compile model with Adam + Crossentropy.
6. Train model for 10 epochs.
7. Evaluate using test set.
8. Plot accuracy & loss curves.
9. Report final test accuracy.

**Diagram:**



**Conclusion:**

CNNs are powerful models for multiclass image classification. By leveraging feature extraction and hierarchical learning, they achieve high accuracy across applications. While computationally intensive, their ability to generalize and recognize complex image patterns makes them essential in fields like healthcare, autonomous systems, and security.