**Assignment No: - 3**

**Name :- Isha Ghorpade**

**Enrollnment :- 22420020**

**Roll :- 381066**

**Problem Statement:**

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

Objectives

* To study the structure and functioning of Convolutional Neural Networks (CNNs).
* To preprocess image data effectively for deep learning models.
* To implement a CNN model using Keras and TensorFlow for multiclass classification.
* To analyze model accuracy and loss using validation data.
* To visualize performance trends such as accuracy and loss across training epochs.

Software and Hardware Requirements

* Operating System: Windows / Linux / macOS
* Kernel/Language: Python 3.x
* Tools/IDEs: Jupyter Notebook, Anaconda, or Google Colab
* Hardware: Minimum 4 GB RAM, GPU recommended for faster training
* Libraries/Packages: TensorFlow, Keras, NumPy, Matplotlib

Theory

A Convolutional Neural Network (CNN) is a specialized deep learning architecture designed for working with images. Unlike traditional neural networks, CNNs use convolutional filters to capture spatial patterns like edges, textures, and shapes directly from images.

Structure of CNN

* Input Layer: Accepts image data, usually normalized and resized.
* Convolutional Layers: Extract low-level to high-level features using filters/kernels.
* Pooling Layers: Reduce dimensions (down-sampling) to preserve important features while lowering computation.
* Fully Connected Layers: Flatten extracted features and connect them to dense layers for decision-making.
* Output Layer: Uses Softmax activation to predict class probabilities in multiclass tasks.

Activation Functions

* ReLU (Rectified Linear Unit): Introduces non-linearity, speeding up training.
* Softmax: Converts outputs into probability distributions across classes.

Backpropagation in CNNs

Training is achieved through backpropagation, where gradients of the loss function are calculated and weights are updated iteratively using optimizers such as Adam or SGD. This process minimizes classification error.

Methodology

1. Data Acquisition:
   * Use the CIFAR-10 dataset (60,000 32×32 color images across 10 classes like airplane, automobile, bird, etc.).
2. Data Preprocessing:
   * Normalize pixel values to a range [0,1].
   * Split dataset into training (50,000 images) and testing (10,000 images).
3. Model Architecture (Sequential CNN):
   * Conv Layer 1: 32 filters (3×3), ReLU activation, followed by MaxPooling (2×2).
   * Conv Layer 2: 64 filters (3×3), ReLU activation, followed by MaxPooling (2×2).
   * Conv Layer 3: 64 filters (3×3), ReLU activation.
   * Flatten Layer: Converts feature maps into 1D vector.
   * Dense Layer: Fully connected layer with 64 units, ReLU activation.
   * Output Layer: 10 neurons (for 10 classes), Softmax activation.
4. Model Compilation:
   * Optimizer: Adam
   * Loss Function: Sparse Categorical Crossentropy
   * Metric: Accuracy
5. Model Training:
   * Train for 10 epochs with batch size = 128.
   * Use training data for learning and test data for validation.
6. Model Evaluation:
   * Evaluate accuracy and loss on unseen test dataset.
7. Performance Visualization:
   * Plot training vs validation accuracy and loss curves over epochs using Matplotlib.

Advantages

* Automatic Feature Extraction: CNNs learn features directly from images, removing manual feature engineering.
* Translation Invariance: Small shifts or distortions in images do not affect performance significantly.
* Parameter Efficiency: Fewer parameters compared to fully connected networks.
* Hierarchical Feature Learning: From simple edges in early layers to complex objects in deeper layers.

Limitations

* Large Data Requirement: CNNs perform best with big datasets.
* Computationally Expensive: Training deep CNNs requires significant time and resources.
* Risk of Overfitting: Without regularization, models may memorize instead of generalizing.
* Hyperparameter Sensitivity: Performance depends heavily on kernel size, number of layers, learning rate, etc.

Applications

* Image Classification: Identifying objects in fields like healthcare (tumor detection), security, and retail.
* Facial Recognition: Verifying and identifying individuals in authentication systems.
* Medical Imaging: Detecting diseases in X-rays, MRIs, and CT scans.
* Autonomous Driving: Recognizing traffic signs, pedestrians, and obstacles.
* Video Analytics: Action recognition and activity detection in surveillance videos.

Working / Algorithm

1. Load CIFAR-10 dataset using TensorFlow/Keras.
2. Normalize images (divide pixel values by 255).
3. Visualize sample images with class labels.
4. Build CNN model using Sequential API in Keras.
5. Add convolutional, pooling, flatten, and dense layers.
6. Compile model with Adam optimizer and categorical crossentropy.
7. Train model for 10 epochs with validation data.
8. Plot accuracy and loss graphs.
9. Evaluate model on test dataset.
10. Print final test accuracy.

Conclusion

Convolutional Neural Networks (CNNs) provide an efficient and reliable method for multiclass image classification. By leveraging convolutional and pooling operations, CNNs automatically learn important features and reduce the need for manual feature extraction. In this experiment, CNNs trained on CIFAR-10 successfully classified images into 10 categories with good accuracy. Despite challenges like computational cost and the need for large datasets, CNNs remain one of the most powerful techniques in modern computer vision, with applications spanning healthcare, security, autonomous driving, and beyond.