**Experiment No. 07**

**Aim**: Design and implement a CNN model for image classification.

**Objectives**:

1. Understand the fundamentals of Convolutional Neural Networks (CNNs).
2. Learn to preprocess image data for CNN-based classification tasks.
3. Design and build a CNN architecture suitable for image classification.
4. Train the CNN model using appropriate training techniques.
5. Evaluate the performance of the CNN model on a test dataset.
6. Gain practical experience in image classification using CNNs.

**Theory**:

Convolutional Neural Networks (CNNs):

* CNNs are a class of deep neural networks widely used for image-related tasks such as image classification, object detection, and segmentation.
* They consist of multiple layers, including convolutional layers, pooling layers, and fully connected layers.
* CNNs leverage the concept of local connectivity and weight sharing to automatically learn hierarchical representations of visual data.

Preprocessing Image Data:

* Image preprocessing involves steps such as resizing, normalization, and data augmentation to prepare the image data for training.
* Resizing ensures that all input images have the same dimensions, which is necessary for feeding them into the CNN model.
* Normalization standardizes the pixel values of images to a common scale, typically between 0 and 1, to facilitate better convergence during training.
* Data augmentation techniques such as rotation, flipping, and zooming are applied to increase the diversity of training data and improve the model's generalization.

Designing a CNN Model:

* The CNN architecture comprises a stack of convolutional layers followed by pooling layers for feature extraction.
* The convolutional layers learn to detect patterns and features from input images through the application of convolutional filters.
* Pooling layers downsample the feature maps, reducing spatial dimensions and extracting the most important features.
* Fully connected layers at the end of the network combine the extracted features for classification.

Training the CNN Model:

* The CNN model is trained using a labeled dataset through an optimization algorithm such as stochastic gradient descent (SGD) or Adam.
* During training, the model learns to minimize a loss function that quantifies the difference between predicted and actual labels.
* Backpropagation is employed to update the model parameters (weights and biases) based on the computed gradients of the loss function with respect to the parameters.

Evaluating Model Performance:

* The performance of the trained CNN model is evaluated on a separate test dataset to assess its accuracy and generalization ability.
* Metrics such as accuracy, precision, recall, and F1-score are commonly used to measure the classification performance.
* Confusion matrices and ROC curves provide insights into the model's ability to correctly classify different classes.

**Conclusion**:

Designing and implementing a CNN model for image classification involves understanding the architecture of CNNs, preprocessing image data, designing an appropriate model architecture, training the model, and evaluating its performance. By completing this experiment, you will gain practical experience in building CNN-based classifiers for various image classification tasks.