**12. Image Classifier: An Application of Deep Learning**

**AIM:** To implement image classification technique using Deep Learning.

**ALGORITHM:**

1. **Import Required Libraries**: use TensorFlow and Keras for building and training the CNN model.
2. **Load the MNIST Dataset**: TensorFlow Keras provides easy access to the MNIST dataset.
3. **Preprocess the Data**: This involves normalizing the pixel values and reshaping the data to fit the CNN input requirements.
4. **Define the CNN Model**: A basic CNN model includes convolutional layers, activation functions, pooling layers, and fully connected layers.
5. **Compile and Train the Model**: Use the 'adam' optimizer and 'sparse\_categorical\_crossentropy' as the loss function.
6. **Evaluate the Model**: Assess the model's performance on the test set.

**PROGRAM**

import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, Dense, Flatten, MaxPooling2D

# Load and preprocess the MNIST dataset

mnist = tf.keras.datasets.mnist

(X\_train, y\_train), (X\_test, y\_test) = mnist.load\_data()

X\_train, X\_test = X\_train / 255.0, X\_test / 255.0  # Normalize pixel values

# Define the CNN model

model = Sequential([

    Conv2D(32, kernel\_size=(3, 3), activation='relu', input\_shape=(28, 28, 1)),

    MaxPooling2D(pool\_size=(2, 2)),

    Conv2D(64, (3, 3), activation='relu'),

    MaxPooling2D(pool\_size=(2, 2)),

    Flatten(),

    Dense(64, activation='relu'),

    Dense(10, activation='softmax')

])

# Compile the model

model.compile(optimizer='adam',

              loss='sparse\_categorical\_crossentropy',

              metrics=['accuracy'])

# Train the model

model.fit(X\_train, y\_train, epochs=5, validation\_split=0.1)

# Evaluate the model

test\_loss, test\_acc = model.evaluate(X\_test, y\_test)

print(f'\nTest accuracy: {test\_acc}')