|  |
| --- |
| Title: Skin Disease Predictor using Transfer Learning in CNN |
| Done by : Pritesh Ram Keshri Project Number : 06, Submission Date: 19th June 2025 |
| **Overview:**  This project demonstrates an image classification system designed to predict various skin diseases using Convolutional Neural Networks (CNN) and the Transfer Learning approach. The model leverages pre-trained weights and has been trained on a labeled image dataset sourced from the DermNet dataset. |
| **Libraries used:**   * **TensorFlow/Keras –** Model creation, training, and prediction * **NumPy –** Efficient array operations * **Glob –** Accessing image file paths for training * **Matplotlib –** Visualization and analysis of model performance |
| **Dataset Details:**   * **Source:** [**DermNet Skin Disease Dataset**](https://www.kaggle.com/datasets/shubhamgoel27/dermnet) * **Format:** Image classification dataset * **Classes:** Multiple skin disease categories such as eczema, psoriasis, acne, etc. * **Structure:** Labeled image folders compatible with Keras’ flow\_from\_directory function |
| **APIs Integrated:**  No external APIs integrated. |
| **Source code 1: File Name : Data Preprocessing (preprocess.py)**  import os  import glob  import numpy as np  from tensorflow.keras.preprocessing.image import ImageDataGenerator  # Set image size and path  IMG\_SIZE = (224, 224)  BATCH\_SIZE = 32  # Data generators for training and validation  train\_datagen = ImageDataGenerator(rescale=1./255, validation\_split=0.2)  train\_data = train\_datagen.flow\_from\_directory(  'dataset/',  target\_size=IMG\_SIZE,  batch\_size=BATCH\_SIZE,  class\_mode='categorical',  subset='training'  )  val\_data = train\_datagen.flow\_from\_directory(  'dataset/',  target\_size=IMG\_SIZE,  batch\_size=BATCH\_SIZE,  class\_mode='categorical',  subset='validation'  ) |
| **Source code 2: File name: Model Architecture & Training (train\_model.py)**  from tensorflow.keras.applications import MobileNetV2  from tensorflow.keras import layers, models  # Load base model with pre-trained weights  base\_model = MobileNetV2(weights='imagenet', include\_top=False, input\_shape=(224, 224, 3))  base\_model.trainable = False # Freeze layers  # Build the complete model  model = models.Sequential([  base\_model,  layers.GlobalAveragePooling2D(),  layers.Dense(128, activation='relu'),  layers.Dropout(0.3),  layers.Dense(train\_data.num\_classes, activation='softmax')  ])  # Compile the model  model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])  # Train the model  model.fit(train\_data, validation\_data=val\_data, epochs=10) |
| **Source code 3: Evaluation & Visualization (evaluate.py)**  import matplotlib.pyplot as plt  # Evaluate the model  loss, accuracy = model.evaluate(val\_data)  print(f"Validation Accuracy: {accuracy \* 100:.2f}%")  # Plot training history  def plot\_history(history):  plt.figure(figsize=(12, 4))    # Accuracy  plt.subplot(1, 2, 1)  plt.plot(history.history['accuracy'], label='Train Accuracy')  plt.plot(history.history['val\_accuracy'], label='Val Accuracy')  plt.title('Model Accuracy')  plt.legend()    # Loss  plt.subplot(1, 2, 2)  plt.plot(history.history['loss'], label='Train Loss')  plt.plot(history.history['val\_loss'], label='Val Loss')  plt.title('Model Loss')  plt.legend()    plt.show() |
| **Source Code 4: Prediction Interface (predict.py)**    import numpy as np  from tensorflow.keras.preprocessing import image  # Load model  from tensorflow.keras.models import load\_model  model = load\_model('skin\_disease\_model.h5')  # Predict single image  def predict\_image(img\_path):  img = image.load\_img(img\_path, target\_size=(224, 224))  img\_array = image.img\_to\_array(img) / 255.0  img\_array = np.expand\_dims(img\_array, axis=0)  prediction = model.predict(img\_array)  predicted\_class = np.argmax(prediction)    return predicted\_class |
|  |
| **Output screenshots:**  **1. Model Architecture Summary:** Shows the complete structure of the transfer learning model and custom layers.    **2. Training Curves**:Displays model accuracy and loss plots across epochs for both training and validation datasets.    **3. Prediction Samples**: Example images with predicted disease class labels. |
| **What you learned:**  Through this project, I gained practical knowledge of:   * Building custom image classifiers with CNN * Applying transfer learning to medical datasets * Handling large image datasets and preprocessing * Evaluating classification performance |
| **What the Skills you gained:**   * Proficient in TensorFlow and Keras API * Applied deep learning in healthcare domain * Built and tested transfer learning models * Performed model evaluation and visualization |
| **Real Time Applications:**   * **Telemedicine:** Quick preliminary diagnostics * **Dermatology Assistants:** AI-driven decision support * **Awareness Campaigns:** Mobile tools for disease recognition * **Rural Healthcare:** Low-cost diagnostic solutions |
| **Further Enhancement Suggestions:**   * Create a Streamlit-based web interface for prediction * Expand dataset for better generalization * Deploy on mobile apps for field usage * Integrate with electronic health record (EHR) systems |