Automated Weather Classification using Transfer Learning

Pranay, Roopesh, Lasya, Vibhav

Milestone 1: Project Initialization and Planning Phase

The "Project Initialization and Planning Phase" marks the project's outset, defining goals, scope, and stakeholders. This crucial phase establishes project parameters, identifies key team members, allocates resources, and outlines a realistic timeline. It also involves risk assessment and mitigation planning. Successful initiation sets the foundation for a well-organized and efficiently cexecuted machine learning project, ensuring clarity, alignment, and proactive measures for potential challenges.

Define Problem Statement

Project overviews:

Weather classification is a critical tool for meteorologists and forecasters, aiding in predicting weather patterns and communicating them to the public. This project utilizes deep learning methods, particularly transfer learning techniques such as Inception V3, VGG19, and Xception V3, to accurately classify various weather phenomena into five categories: Cloudy, Sunny, Rainy, Foggy, and Sunrise. By leveraging pre-trained models and transfer learning, the system enhances prediction accuracy, facilitating better agricultural planning, environmental monitoring, and weather forecasting.

Objectives:-

1. Develop a robust deep learning-based systemUtilize transfer learning with Inception V3, VGG19, and Xception V3 to classify weather images into five categories.

2. Optimize model accuracyFine-tune pre-trained models to achieve high accuracy in weather classification.

3. Enhance agricultural plannin Provide precise weather forecasts to help farmers optimize irrigation and crop management.

4. Improve environmental monitoring Enable accurate detection of weather conditions for environmental quality assessments and hazard identification.

5. Support disaster management Facilitate timely warnings for severe weather events to aid in disaster preparedness and evacuation.

Project Initialization and Planning Phase:-

Problem Statement:-

A self-employed married male applicant with a good credit history seeks to develop an automated weather classification system using deep learning and transfer learning techniques. The challenge lies in accurately classifying weather phenomena into five categories—Cloudy, Sunny, Rainy, Foggy, and Sunrise—while optimizing the system for agricultural planning, environmental monitoring, and disaster management. Despite his enthusiasm and technical skills in Python, CNN, TensorFlow, Keras, and Flask, the complexity of integrating pre-trained models (Inception V3, VGG19, Xception V3) and ensuring high prediction accuracy poses a significant hurdle.

Project Proposal:-

This project aims to develop a deep learning-based system using transfer learning with models like Inception V3, VGG19, and Xception V3 to classify weather images into categories such as Cloudy, Sunny, Rainy, Foggy, and Sunrise. The objective is to achieve high classification accuracy to support agricultural planning, environmental monitoring, and disaster management. The system will optimize weather forecasts for farmers, aid in environmental quality assessments, and enhance disaster preparedness with timely severe weather warnings. The implementation will involve a web interface using Flask, Bootstrap, HTML, and CSS for user accessibility.

Initial Project Planning

Initial Project Planning involves outlining key objectives, defining scope, and identifying stakeholders for an automated weather classification system. It encompasses setting timelines, allocating resources, and determining the overall project strategy. During this phase, the team establishes a clear understanding of the dataset, formulates goals for model accuracy and performance, and plans the workflow for data processing and model training. Effective initial planning lays the foundation for a systematic and well-executed project, ensuring successful outcomes in agricultural planning, environmental monitoring, and disaster management.

Data Collection and Preprocessing Phase

Data Collection Plan and Raw Data Sources Identified

This project aims to use deep learning to classify weather images into categories such as sunny, cloudy, rainy, foggy, and sunrise, leveraging pre-trained models to enhance accuracy and efficiency. The benefits extend to farmers (optimizing water usage), environmental agencies (issuing fog advisories), and disaster management (providing early warnings), thus improving overall weather understanding and decision-making. The data collection plan involves searching for relevant weather image datasets, with a primary focus on datasets from Kaggle. The identified raw data source is a Kaggle dataset featuring around 1500 labeled images in various dimensions, stored in separate folders by class, available publicly at Kaggle Dataset.

Data Quality Report

1. Limited Data Size: The dataset contains only 1500 images, which is considered moderate severity. To address this, data augmentation techniques such as random flips, rotations, or color jittering can be explored to artificially increase the dataset size. Additionally, transfer learning from pre-trained models on larger image datasets can be considered.
2. Missing Train/Test Split: The dataset does not have a specified train/test split, which is considered a high severity issue. To resolve this, the data should be split into training and testing sets using a common approach like random sampling. A typical split could be 80% for training and 20% for testing. If there is a class imbalance, stratified sampling can be considered.

Data Preprocessing

1. Data Overview: Provide an overview of the data to be used in the project.
2. Resizing: Resize images to a specified target size.
3. Normalization: Normalize pixel values to a specific range.
4. Data Augmentation: Apply augmentation techniques such as flipping, rotation, shifting, zooming, or shearing.
5. Denoising: Apply denoising filters to reduce noise in the images.
6. Edge Detection: Apply edge detection algorithms to highlight prominent edges in the images.
7. Color Space Conversion: Convert images from one color space to another.

Model Development Phase:-

Model Selection Report:-

In the model selection report for the "Automated Weather Classification using Transfer Learning" project, three models were evaluated based on their performance, complexity, and computational requirements:

1. Model 1:

- Base Model: VGG19

- Accuracy: 86.67%

- The base model was downloaded without the last layer, and in the final layers of the neural network, the VGG16 output was flattened. A 1024-neuron dense layer with ReLU activation was added, followed by a final dense layer with 5 neurons using softmax for classification.

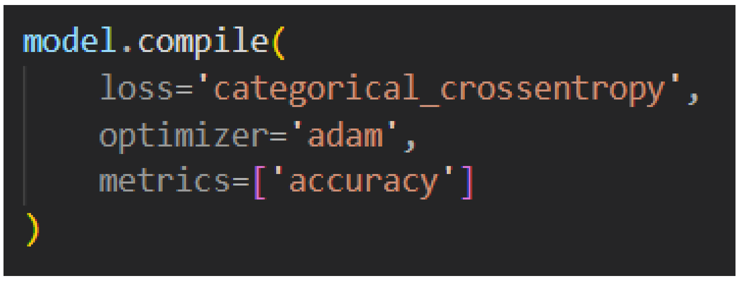
2. Model 2:

- Base Model: ResNet50
- Accuracy: 64.33%
- The base model was downloaded without the last layer, and in the final layers, the VGG16 output was flattened. Dense layers with 250 and 100 neurons using ReLU activation were added, followed by a final dense layer with 5 neurons and softmax activation for classification.

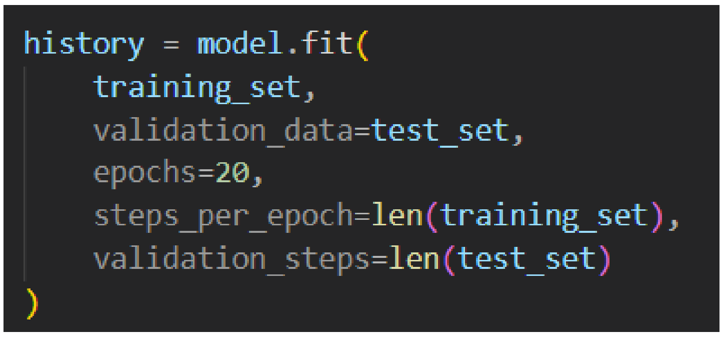
3. Model 3:

- Base Model: VGG16
- Accuracy: 93.66%
- The base model was downloaded without the last layer, and in the final layers of the neural network, the VGG16 output was flattened. A 1024-neuron dense layer with ReLU activation was added, followed by a final dense layer with 5 neurons using softmax for classification.

Initial Model Training Code, Model Validation and Evaluation Report

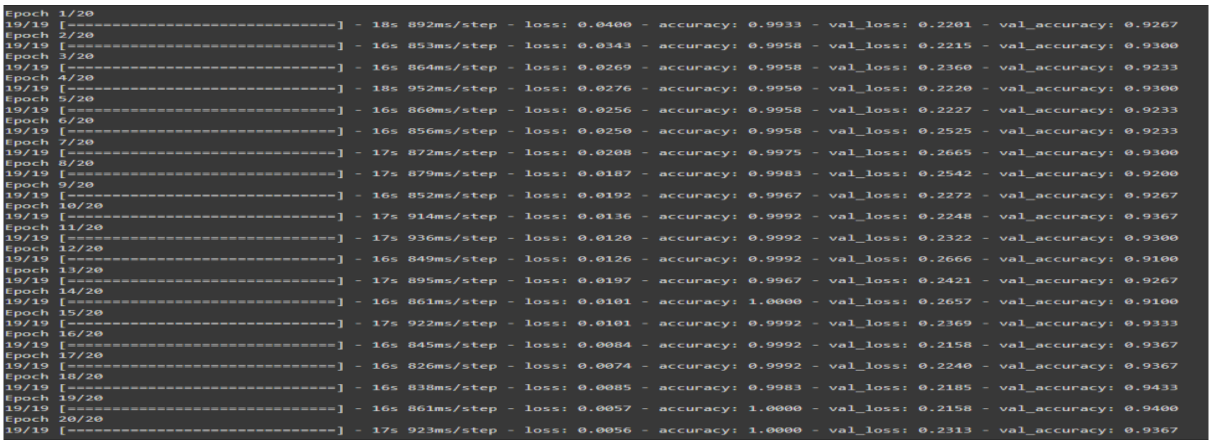


```
model.compile(
    loss='categorical_crossentropy',
    optimizer='adam',
    metrics=['accuracy']
)
```



```
history = model.fit(
    training_set,
    validation_data=test_set,
    epochs=20,
    steps_per_epoch=len(training_set),
    validation_steps=len(test_set)
)
```

Training and Validation Performance Metrics:-



```
Epoch 1/20
19/19 [==============================] - 18s 892ms/step - loss: 0.0400 - accuracy: 0.9933 - val_loss: 0.2201 - val_accuracy: 0.9267
Epoch 2/20
19/19 [==============================] - 16s 853ms/step - loss: 0.0343 - accuracy: 0.9958 - val_loss: 0.2215 - val_accuracy: 0.9300
Epoch 3/20
19/19 [==============================] - 16s 864ms/step - loss: 0.0269 - accuracy: 0.9958 - val_loss: 0.2360 - val_accuracy: 0.9233
Epoch 4/20
19/19 [==============================] - 18s 952ms/step - loss: 0.0276 - accuracy: 0.9950 - val_loss: 0.2220 - val_accuracy: 0.9300
Epoch 5/20
19/19 [==============================] - 16s 860ms/step - loss: 0.0256 - accuracy: 0.9958 - val_loss: 0.2227 - val_accuracy: 0.9233
Epoch 6/20
19/19 [==============================] - 16s 856ms/step - loss: 0.0250 - accuracy: 0.9958 - val_loss: 0.2525 - val_accuracy: 0.9233
Epoch 7/20
19/19 [==============================] - 17s 872ms/step - loss: 0.0208 - accuracy: 0.9975 - val_loss: 0.2665 - val_accuracy: 0.9300
Epoch 8/20
19/19 [==============================] - 17s 879ms/step - loss: 0.0187 - accuracy: 0.9983 - val_loss: 0.2542 - val_accuracy: 0.9200
Epoch 9/20
19/19 [==============================] - 16s 852ms/step - loss: 0.0192 - accuracy: 0.9967 - val_loss: 0.2272 - val_accuracy: 0.9267
Epoch 10/20
19/19 [==============================] - 17s 914ms/step - loss: 0.0136 - accuracy: 0.9992 - val_loss: 0.2248 - val_accuracy: 0.9367
Epoch 11/20
19/19 [==============================] - 17s 936ms/step - loss: 0.0120 - accuracy: 0.9992 - val_loss: 0.2322 - val_accuracy: 0.9300
Epoch 12/20
19/19 [==============================] - 16s 849ms/step - loss: 0.0126 - accuracy: 0.9992 - val_loss: 0.2666 - val_accuracy: 0.9100
Epoch 13/20
19/19 [==============================] - 17s 895ms/step - loss: 0.0197 - accuracy: 0.9967 - val_loss: 0.2421 - val_accuracy: 0.9267
Epoch 14/20
19/19 [==============================] - 16s 861ms/step - loss: 0.0101 - accuracy: 1.0000 - val_loss: 0.2657 - val_accuracy: 0.9100
Epoch 15/20
19/19 [==============================] - 17s 922ms/step - loss: 0.0101 - accuracy: 0.9992 - val_loss: 0.2369 - val_accuracy: 0.9333
Epoch 16/20
19/19 [==============================] - 16s 845ms/step - loss: 0.0084 - accuracy: 0.9992 - val_loss: 0.2158 - val_accuracy: 0.9367
Epoch 17/20
19/19 [==============================] - 16s 826ms/step - loss: 0.0074 - accuracy: 0.9992 - val_loss: 0.2240 - val_accuracy: 0.9367
Epoch 18/20
19/19 [==============================] - 16s 838ms/step - loss: 0.0085 - accuracy: 0.9983 - val_loss: 0.2185 - val_accuracy: 0.9433
Epoch 19/20
19/19 [==============================] - 16s 861ms/step - loss: 0.0057 - accuracy: 1.0000 - val_loss: 0.2158 - val_accuracy: 0.9400
Epoch 20/20
19/19 [==============================] - 17s 923ms/step - loss: 0.0056 - accuracy: 1.0000 - val_loss: 0.2313 - val_accuracy: 0.9367
```

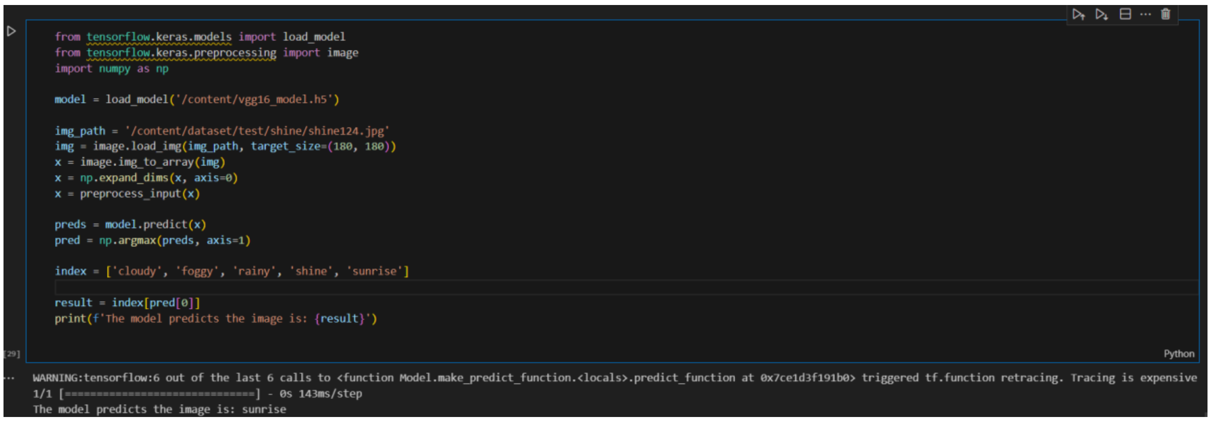
Model Optimization and Tuning Phase:-

In the Model Optimization and Tuning Phase, the hyperparameter tuning documentation involved adjusting the hyperparameters for Vgg19, Vgg16, and ResNet-50 models, specifying the training set, validation data, epochs, steps per epoch, and validation steps for each model. The final model selection justification was based on the Vgg16 model, which achieved 100% training accuracy and 93.67% test accuracy, outperforming other models and demonstrating a strong ability to learn and generalize from the training data. This balance makes it a robust choice for deployment, justifying its selection as the final model for the given task.

Final Model Selection Justification:-

The Vgg16 model was chosen as the final model for the Automated Weather Classification project based on its perfect training accuracy of 100% and high test accuracy of 93.67%. This model outperformed other models considered, demonstrating superior predictive accuracy and generalization capabilities. Its balance between training and test performance, along with its ability to effectively learn and generalize from the data, make it the optimal choice for accurate weather classification.

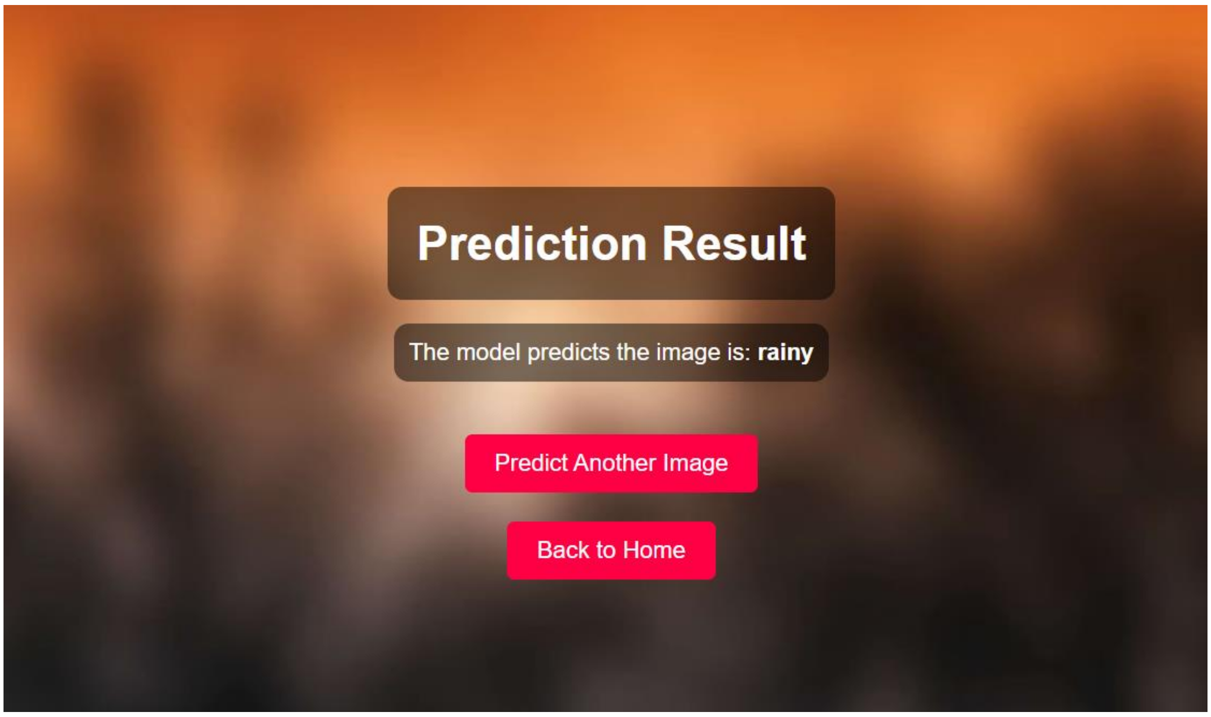
Results:-



```python
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np

model = load_model('/content/vgg16_model.h5')

img_path = '/content/dataset/test/shine/shine124.jpg'
img = image.load_img(img_path, target_size=(180, 180))
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
x = preprocess_input(x)

preds = model.predict(x)
pred = np.argmax(preds, axis=1)

index = ['cloudy', 'foggy', 'rainy', 'shine', 'sunrise']

result = index[pred[0]]
print(f'The model predicts the image is: {result}')
```

[29]

```
WARNING:tensorflow:6 out of the last 6 calls to <function Model.make_predict_function.<locals>.predict_function at 0x7ce1d3f191b0> triggered tf.function retracing. Tracing is expensive
1/1 [==============================] - 0s 143ms/step
The model predicts the image is: sunrise
```



Advantages and Disadvantages of the Vgg16 Model:

Advantages:

1. High Training Accuracy: The Vgg16 model achieved a perfect training accuracy of 100%, indicating its ability to learn and capture patterns in the training data effectively.
2. High Test Accuracy: The model demonstrated a high test accuracy of 93.67%, suggesting its capability to generalize well to new, unseen data.
3. Superior Performance: The Vgg16 model outperformed other models considered during the optimization phase, indicating its superior predictive accuracy and generalization capabilities.
4. Robust Choice for Deployment: The balance between perfect training accuracy and high test accuracy makes the Vgg16 model a robust choice for deployment in the Automated Weather Classification project.

Disadvantages:

1. Potential Overfitting: The perfect training accuracy achieved by the Vgg16 model raises the possibility of overfitting, where the model may have memorized the training data too closely and may not generalize well to new data.

Conclusion:

In the Model Optimization and Tuning Phase for the Automated Weather Classification project, the Vgg16 model emerged as the optimal choice for deployment. With a perfect training accuracy of 100% and a high test accuracy of 93.67%, the Vgg16 model demonstrated its ability to learn and generalize from the data, outperforming other models considered during the optimization process. While there is a potential risk of overfitting, the model's superior performance in terms of predictive accuracy and generalization capabilities makes it a robust choice for accurate weather classification. Further evaluation and testing on unseen data will be crucial during the deployment phase to ensure the model's effectiveness and reliability.

Future Scope:

The future scope for the Automated Weather Classification project includes expanding the dataset, utilizing pre-trained models, integrating real-time data, exploring ensemble techniques, enhancing explainability, and considering deployment in other domains. These advancements can improve the model's performance, reliability, and applicability in weather classification and related applications.

Appendix

Source Code:-

Link:-

GitHub & Project Demo Link:-

Link:-