Deepfake Project

Problem Statement

The objective of this project is to develop a system capable of distinguishing between real and manipulated (deep fake) facial images. Given the increasing prevalence of deep fake technologies, this project is critical for applications in cybersecurity, media authenticity verification, and public safety. The detection task is treated as a binary classification problem where the two classes are "real" and "fake."

Dataset Description

The dataset used in this project was sourced from KAGGLE and further processed for maximum effectiveness. The dataset contains:

- **10,000 images in total**:
  - 5,000 real images
  - 5,000 manipulated images
- Each image is of resolution **256x256 pixels** in JPG format.

For training purposes, the dataset was processed and split using **3-fold cross-validation**, ensuring a balanced distribution of real and fake images in each fold.

Cross-Validation Strategy

- **3-Fold Cross-Validation**:
  - The dataset was divided into three subsets.
  - In each fold, two subsets were used for training, and the remaining subset was used for validation.
  - This ensures robustness and minimizes overfitting.

Preprocessing

The preprocessing steps included the following transformations:

1. **Resizing**:
   - Images were resized to **224x224 pixels** for all models except Inception, which required **299x299 pixels** due to its architecture.
2. **Normalization**:
   - Applied pixel value normalization using transforms. Normalize([0.5, 0.5, 0.5], [0.5, 0.5, 0.5]).
3. **Conversion to Tensor**:
   - Images were converted into tensors to be compatible with PyTorch models.

Model Selection and Fine-Tuning

The following pretrained models were fine-tuned for the classification task:

1. **ResNet-50**:
   - Fine-tuned the **last two layers**.
2. **DenseNet-121**:
   - Fine-tuned the **last five blocks**.
3. **VGG-16**:
   - Fine-tuned the **last two layers**.
4. **Inception**:
   - Fine-tuned the **last four layers**.
5. **MobileNet**:
   - Fine-tuned the **last five layers**.
6. **Vision Transformer**:
   - Fine-tuned the **last transformer block.**

Learning Rate Optimization with Optuna

- **Optuna** was used to optimize the learning rate (LR) for each model.
- The optimization process identified the LR that yielded the highest validation accuracy for each fold.

Real

Fake

Vgg-16 Performance without Fine-Tune:

Best learning rate 0.0001754805798745925

Fold 1 - Accuracy: 0.5771, Precision: 0.6157, Recall: 0.5771, F1-Score: 0.5547

Confusion Matrix:

[[265 58]

[230 128]]

Fold 2 - Accuracy: 0.6000, Precision: 0.6330, Recall: 0.6000, F1-Score: 0.5812
Confusion Matrix:

[[271 58]

[214 137]]

Fold 3 - Accuracy: 0.5735, Precision: 0.6655, Recall: 0.5735, F1-Score: 0.5361
Confusion Matrix:

[[278   30]

[260 112]]

Average Performance of VGG16 Without Fine-Tune

Average Accuracy: 0.5943

Average Precision: 0.6244

Average Recall: 0.5943 Average

F1-Score: 0.5810 Average

Confusion Matrix:

[[240.66666667 79.33333333]

[196.66666667 163.66666667]]

Vgg-16 Performance with Fine-Tune:

Best learning rate 2.5126450824282805e-05

Fold 1 - Accuracy: 0.9235, Precision: 0.9247, Recall: 0.9235, F1-Score: 0.9234

Confusion Matrix:

```
[[1774   90]
 [ 188 1583]]
```

Fold 2 - Accuracy: 0.9221, Precision: 0.9245, Recall: 0.9221, F1-Score: 0.9220

Confusion Matrix:

```
[[1761   73]
 [ 210 1591]]
```

Fold 3 - Accuracy: 0.9202, Precision: 0.9205, Recall: 0.9202, F1-Score: 0.9202

Confusion Matrix:

```
[[1673  121]
 [ 169  1672]]
```

Average Performance of VGG16 With Fine-Tune

Average Accuracy: 0.9220

Average Precision: 0.9232

Average Recall: 0.9220

Average F1-Score: 0.9219

Average Confusion Matrix:

```
[[1736.          94.66666667]
 [ 189.        1615.33333333]]
```

ResNet-50 Performance without Fine-Tune:

Best learning rate = 0.0006172423196498955

Fold 1 - Accuracy: 0.7243, Precision: 0.7252, Recall: 0.7243, F1-Score: 0.7236

Confusion Matrix:

```
[[1436  428]
 [ 574   1197]]
```

Fold 2 - Accuracy: 0.7175, Precision: 0.7313, Recall: 0.7175, F1-Score: 0.7127

Confusion Matrix:

```
[[1547  287]
 [ 740   1061]]
```

Fold 3 - Accuracy: 0.7260, Precision: 0.7381, Recall: 0.7260, F1-Score: 0.7231

Confusion Matrix:

```
[[1495  299]
 [ 697   1144]]
```

Average Performance of RESNET-50 Without Fine-Tune

Average Accuracy: 0.7226

Average Precision: 0.7315

Average Recall: 0.7226

Average F1-Score: 0.7198

Average Confusion Matrix:

```
[[1492.66666667      338]
 [ 670.33333333    1134]]
```

ResNet-50 Performance with Fine-Tune :

best learning rate 0.00084522309417377776

Fold 1 - Accuracy: 0.9307, Precision: 0.9309, Recall: 0.9307, F1-Score: 0.9306

Confusion Matrix:

```
[[1760  104]
 [ 148 1623]]
```

Fold 2 - Accuracy: 0.9287, Precision: 0.9302, Recall: 0.9287, F1-Score: 0.9287

Confusion Matrix:

```
[[1758   76]
 [ 183 1618]]
```

Fold 3 - Accuracy: 0.9301, Precision: 0.9325, Recall: 0.9301, F1-Score: 0.9301

Confusion Matrix:

```
[[1733   61]
 [ 193 1648]]
```

Average Performance of RESNET-50 With Fine-Tune

Average Accuracy: 0.9298

Average Precision: 0.9312

Average Recall: 0.9298

Average F1-Score: 0.9298

Average Confusion Matrix:

```
[[1750.33333333   80.33333333]
 [ 174.66666667 1629.66666667]]
```

Mobilenet Performance without Fine-Tune:

Best learning rate 0.007232401612829719

Fold 1 - Accuracy: 0.7073, Precision: 0.7079, Recall: 0.7073, F1-Score: 0.7066

Confusion Matrix:

```
[[1399  465]
 [ 599 1172]]
```

Fold 2 - Accuracy: 0.7254, Precision: 0.7264, Recall: 0.7254, F1-Score: 0.7253

Confusion Matrix:

```
[[1281  553]
 [ 445  1356]]
```

Fold 3 - Accuracy: 0.6790, Precision: 0.7387, Recall: 0.6790, F1-Score: 0.6557

Confusion Matrix:

```
[[ 751  1043]
 [ 124  1717]]
```

Average Performance of Mobilenet Without Fine-Tune

Average Accuracy: 0.7039

Average Precision: 0.7243

Average Recall: 0.7039

Average F1-Score: 0.6959

Average Confusion Matrix:

```
[[1143.66666667    687.   ]
 [ 389.33333333     1415.]]
```

Mobilenet Performance with Fine-Tune:

Best learning rate 0.0009289966381913214

Fold 1 - Accuracy: 0.8825, Precision: 0.8863, Recall: 0.8825, F1-Score: 0.8821

Confusion Matrix:

```
[[1744   120]
 [ 307  1464]]
```

Fold 2 - Accuracy: 0.8751, Precision: 0.8751, Recall: 0.8751, F1-Score: 0.8751

Confusion Matrix:

```
[[1618   216]
 [ 238  1563]]
```

Fold 3 - Accuracy: 0.8957, Precision: 0.9006, Recall: 0.8957, F1-Score: 0.8955

Confusion Matrix:

```
[[1703   91]
 [ 288  1553]]
```

Average Performance of Mobilenet With Fine-Tune

Average Accuracy: 0.8845

Average Precision: 0.8874

Average Recall: 0.8845

Average F1-Score: 0.8842

Average Confusion Matrix:

```
[[1688.33333333   142.33333333]
 [ 277.66666667    1526.66666667]]
```

Densenet121 Performance without Fine-Tune:

Best learning rate 0.0005446615699330324

Fold 1 - Accuracy: 0.7004, Precision: 0.7223, Recall: 0.7004, F1-Score: 0.6907

Confusion Matrix:

```
[[1618  246]
 [ 843  928]]
```

Fold 2 - Accuracy: 0.6894, Precision: 0.7208, Recall: 0.6894, F1-Score: 0.6770

Confusion Matrix:

```
[[1617  217]
 [ 912  889]]
```

Fold 3 - Accuracy: 0.6897, Precision: 0.7215, Recall: 0.6897, F1-Score: 0.6794

Confusion Matrix:

```
[[1568  226]
 [ 902  939]]
```

Average Performance of Densenet121 Without Fine-Tune

Average Accuracy: 0.6932

Average Precision: 0.7215

Average Recall: 0.6932

Average F1-Score: 0.6824

Average Confusion Matrix:

```
[[1601.         229.66666667]
 [ 885.66666667  918.66666667]]
```

Densenet121 Performance with Fine-Tune:

Best learning rate 0.0008263872006727257

Fold 1 - Accuracy: 0.9208, Precision: 0.9254, Recall: 0.9208, F1-Score: 0.9204

Confusion Matrix:

```
[[1817   47]
 [ 241 1530]]
```

Fold 2 - Accuracy: 0.9232, Precision: 0.9280, Recall: 0.9232, F1-Score: 0.9230

Confusion Matrix:

```
[[1791   43]
 [ 236 1565]]
```

Fold 3 - Accuracy: 0.9387, Precision: 0.9391, Recall: 0.9387, F1-Score: 0.9386

Confusion Matrix:

```
[[1710   84]
 [ 139 1702]]
```

Average Performance of Densenet121 With Fine-Tune

Average Accuracy: 0.9276

Average Precision: 0.9308

Average Recall: 0.9276

Average F1-Score: 0.9274

Average Confusion Matrix:

```
[[1772.66666667    58.    ]
 [ 205.33333333   1599.    ]]
```

Inception performance without Fine-Tune:

Best learning rate = 0.00015312307074091057

Fold 1

Accuracy: 0.6322, Precision: 0.6895, Recall: 0.6322, F1-Score: 0.5965

Confusion Matrix:

```
[[1711  153]
 [1184   587]]
```

Fold 2

Accuracy: 0.6757, Precision: 0.7100, Recall: 0.6757, F1-Score: 0.6607

Confusion Matrix:

```
[[1617  217]
 [ 962  839]]
```

Fold 3

Accuracy: 0.6283, Precision: 0.6885, Recall: 0.6283, F1-Score: 0.5988

Confusion Matrix:

```
[[1624  170]
 [1181  660]]
```

Average Performance of Inception Without Fine-Tune

Average Accuracy: 0.6454

Average Precision: 0.6960

Average Recall: 0.6454

Average F1-Score: 0.6187

Average Confusion Matrix:

```
[[1650.66666667  180.        ]
 [1109.         695.33333333]]
```

Inception performance with Fine-Tune:

Best learning rate = 0.0003267594210449535

Fold 1

Accuracy: 0.8919, Precision: 0.8983, Recall: 0.8919, F1-Score: 0.8912

Confusion Matrix:

```
[[1786   78]
 [ 315 1456]]
```

Fold 2

Accuracy: 0.9029, Precision: 0.9056, Recall: 0.9029, F1-Score: 0.9027

Confusion Matrix:

```
[[1733  101]
 [ 252 1549]]
```

Fold 3

Accuracy: 0.9106, Precision: 0.9152, Recall: 0.9106, F1-Score: 0.9104

Confusion Matrix:

```
[[1726   68]
 [ 257 1584]]
```

Average Performance of Inception with Fine-Tune

Average Accuracy: 0.9018

Average Precision: 0.9064

Average Recall: 0.9018

Average F1-Score: 0.9014

Average Confusion Matrix:

```
[[1748.33333333   82.33333333]
 [ 274.66666667 1529.66666667]]
```

ResNet-50 with adding one layer:

Best learning rate = 0.00021707829167244333

Fold 1 - Accuracy: 0.9307, Precision: 0.9313, Recall: 0.9307, F1-Score: 0.9306

Confusion Matrix:

```
[[1775   89]
 [ 163 1608]]
```

Fold 2 - Accuracy: 0.9268, Precision: 0.9283, Recall: 0.9268, F1-Score: 0.9267

Confusion Matrix:

```
[[1755   79]
 [ 187 1614]]
```

Fold 3 - Accuracy: 0.9260, Precision: 0.9303, Recall: 0.9260, F1-Score: 0.9259

Confusion Matrix:

```
[[1749   45]
 [ 224 1617]]
```

Average Performance of ResNet-50 With Fine-Tune

Average Accuracy: 0.9278

Average Precision: 0.9300

Average Recall: 0.9278

Average F1-Score: 0.9277

Average Confusion Matrix:

```
[[1759.66666667   71.    ]
 [ 191.33333333 1613.    ]]
```

Vision Transformer (VIT) with Fine-Tune:

Best learning rate = 0.0005134706023131347

Fold 1 - Accuracy: 0.8080, Precision: 0.8308, Recall: 0.8080, F1-Score: 0.8037

Confusion Matrix:

```
[[1759  105]
 [ 593   1178]]
```

Fold 2 - Accuracy: 0.8072, Precision: 0.8235, Recall: 0.8072, F1-Score: 0.8044

Confusion Matrix:

```
[[1690  144]
 [ 557   1244]]
```

Fold 3 - Accuracy: 0.8085, Precision: 0.8199, Recall: 0.8085, F1-Score: 0.8071

Confusion Matrix:

```
[[1614  180]
 [ 516   1325]]
```

Average Performance of VIT With Fine-Tune

Average Accuracy: 0.8079

Average Precision: 0.8247

Average Recall: 0.8079

Average F1-Score: 0.8051

Average Confusion Matrix:

```
[[1687.66666667  143.       ]
 [ 555.33333333 1249.       ]]
```

Performance For All Models

Model	Fine-Tuning	Best Learning Rate	Accuracy	Precision	Recall	F1-Score
VGG16	Without	0.000175481	59.43%	62.44%	59.43%	58.10%
	With	2.51E-05	92.20%	92.32%	92.20%	92.19%
ResNet-50	Without	0.000617242	72.26%	73.15%	72.26%	71.98%
	With	0.000845223	92.98%	93.12%	92.98%	92.98%
MobileNet	Without	0.007232402	70.39%	72.43%	70.39%	69.59%
	With	0.000928997	88.45%	88.74%	88.45%	88.42%
DenseNet-121	Without	0.000544662	69.32%	72.15%	69.32%	68.24%
	With	0.000826387	92.76%	93.08%	92.76%	92.74%
Inception	Without	0.000153123	64.54%	69.60%	64.54%	61.87%
	With	0.000326759	90.18%	90.64%	90.18%	90.14%
ResNet-50 (1 Layer)	With	0.000217078	92.78%	93.10%	92.78%	92.77%
VIT (Vision Transformer)	With	0.000513470	80.79%	82.47%	80.79%	80.51%

Conclusion

Among the tested models, **ResNet-50** performed the best, achieving the highest F1-Score of 93%. This demonstrates its capability to accurately classify real and manipulated images.