DEEP LEARNING FOR ELECTRICAL AND COMPUTER ENGINEERS

EC9170

MIN PROJECT

REAL AND FAKE FACE DETECTION

MUTHUGALA M.K.M	2020/E/100
PRAMUDITHA R.M.H	2020/E/114
SENARATHNE P.L.O.N	2020/E/188

Table of Contents

Abstract

The recent bloom of deep learning has led to significant advancements in the field of image recognition, including the ability to distinguish between real and fake faces. This mini project explores the application of deep learning for real and fake face detection, leveraging both transfer learning and a custom Convolutional Neural Network (CNN) architecture. Specifically, the pre-trained VGG16 model is fine-tuned on a dataset of real and fake faces to harness its powerful feature extraction capabilities. Complementing this approach, a custom CNN is designed and trained from scratch to compare its performance with the transfer learning model. The project aims to evaluate the effectiveness of these two methodologies in detecting fraudulent facial images, analyzing their accuracy, computational efficiency, and robustness. Through this comparative study, insights are gained into the benefits and limitations of transfer learning versus custom model development in the context of deepfake detection, contributing to the broader efforts in combating digital fraud and enhancing cybersecurity.

Introduction

With the rapid advancement of deep learning technologies, the ability to distinguish between real and fake faces has become increasingly important. This is due to the rise of deepfake technology, which can create highly realistic yet entirely fake images and videos. These deepfakes can lead to misinformation, privacy violations, and other security issues.

This project focuses on classifying real and fake faces using two different deep learning approaches: a custom Convolutional Neural Network (CNN) and the pre-trained VGG16 model through transfer learning. Our aim is to compare these models in terms of their effectiveness in detecting fake faces.

The dataset used in this project contains a balanced collection of real and fake facial images. The real images are authentic photos of individuals, while the fake images are generated using various deepfake techniques. This ensures that the models are trained on a wide variety of facial features and manipulations, promoting robust learning

For the custom CNN, we will define, compile, and train a model specifically designed to detect the subtle differences between real and fake faces. This involves creating a unique architecture tailored to our dataset and task requirements.

In parallel, we will utilize the VGG16 model, which has been pre-trained on a large dataset of diverse images. By fine-tuning the top layers of VGG16 with our specific dataset, we aim to leverage its powerful feature extraction capabilities to improve our detection accuracy.

The implementation steps include loading and augmenting the data, training the models, and evaluating their performance using accuracy and loss metrics, as well as confusion matrices. By plotting these results, we can visually compare the performance of the custom CNN and VGG16 models.

Through this comparative study, we hope to gain insights into the strengths and weaknesses of each approach, contributing to the broader efforts in enhancing digital security and combating the misuse of deepfake technology.

Methodology

1. Dataset Preparation:

Dataset Cloning and Setup:

- Clone the dataset repository containing real and fake face images.
- Define the paths for training, validation, and test datasets.
- Split the dataset into training, validation, and test sets with appropriate ratios.

Data Augmentation and Normalization:

- For training data: Apply rescaling, shear, zoom, and horizontal flip for augmentation.
- For validation and test data: Apply rescaling.

2. Model Development:

Custom CNN Model:

- **Architecture:**
  - Convolutional layers with increasing filters (32, 64, 128), each followed by Max Pooling.
  - Flatten layer followed by Dense layers (128 units) and a Dropout layer (0.5).
  - Output layer with a single neuron and sigmoid activation.
- **Compilation:**
  - Optimizer: Adam
  - Loss Function: Binary Cross entropy
  - Metrics: Accuracy
- **Training:**
  - Train the model on the training set and validate using the validation set.
  - Use Model Checkpoint and Early Stopping callbacks for better training control.

VGG16 Transfer Learning Model:

- **Architecture:**
  - Base model: Pre-trained VGG16 (excluding top layers).
  - Additional layers: Flatten, Dense (256 units) with Dropout (0.5), and output layer with sigmoid activation.
- **Compilation:**
  - Optimizer: Adam
  - Loss Function: Binary Cross entropy
  - Metrics: Accuracy
- **Training:**
  - Train the model on the training set and validate using the validation set.
  - Use Model Checkpoint and Early Stopping callbacks.

3. Evaluation:

Performance Metrics:

- Evaluate both models on the test set.
- Compute metrics: Accuracy, Precision, Recall, F1-score.

- Generate a classification report for detailed performance analysis.
- Create confusion matrices to visualize true vs. predicted labels.

Plotting Training History:

- Plot training and validation accuracy, loss, and F1-score over epochs for both models.
- Save and visualize these plots for performance comparison.

Model Comparison:

- Compare the performance metrics of both models.
- Analyze confusion matrices to understand the classification accuracy for each class (Real vs. Fake).

4. Selection and Justification:

Selection Criteria:

- Select the model with higher accuracy, precision, recall, and F1-score.
- Prefer the model demonstrating better generalization on the test set.

Justification:

- Theoretical and empirical justification based on the model architecture, depth, and use of transfer learning.
- Highlight the advantages of using a pre-trained VGG16 model for feature extraction and improved performance.

Models

1. Custom CNN Model

Architecture:

- Layers:
  - Conv2D(32) + MaxPooling2D
  - Conv2D(64) + MaxPooling2D
  - Conv2D(128) + MaxPooling2D
  - Flatten
  - Dense(128) + Dropout(0.5)
  - Dense(1) [Output Layer]

Training:

- **Data Augmentation:**
  - Shear, zoom, and horizontal flip
- **Optimizer:** Adam
- **Loss:** Binary Crossentropy
- **Metrics:** Accuracy

Performance:

- Evaluated on test data
- Classification report and confusion matrix generated

2. VGG16 Transfer Learning Model

Architecture:

- **Base Model:** Pre-trained VGG16 (excluding top layers)
- **Additional Layers:**
  - Flatten
  - Dense(256) + Dropout(0.5)
  - Dense(1) [Output Layer]

Training:

- **Data Augmentation:** Same as custom CNN
- **Freezing:** VGG16 layers frozen to prevent retraining
- **Optimizer:** Adam
- **Loss:** Binary Crossentropy
- **Metrics:** Accuracy

Performance:

- Evaluated on test data
- Classification report and confusion matrix generated

Results

Based on theoretical understanding and typical performance:

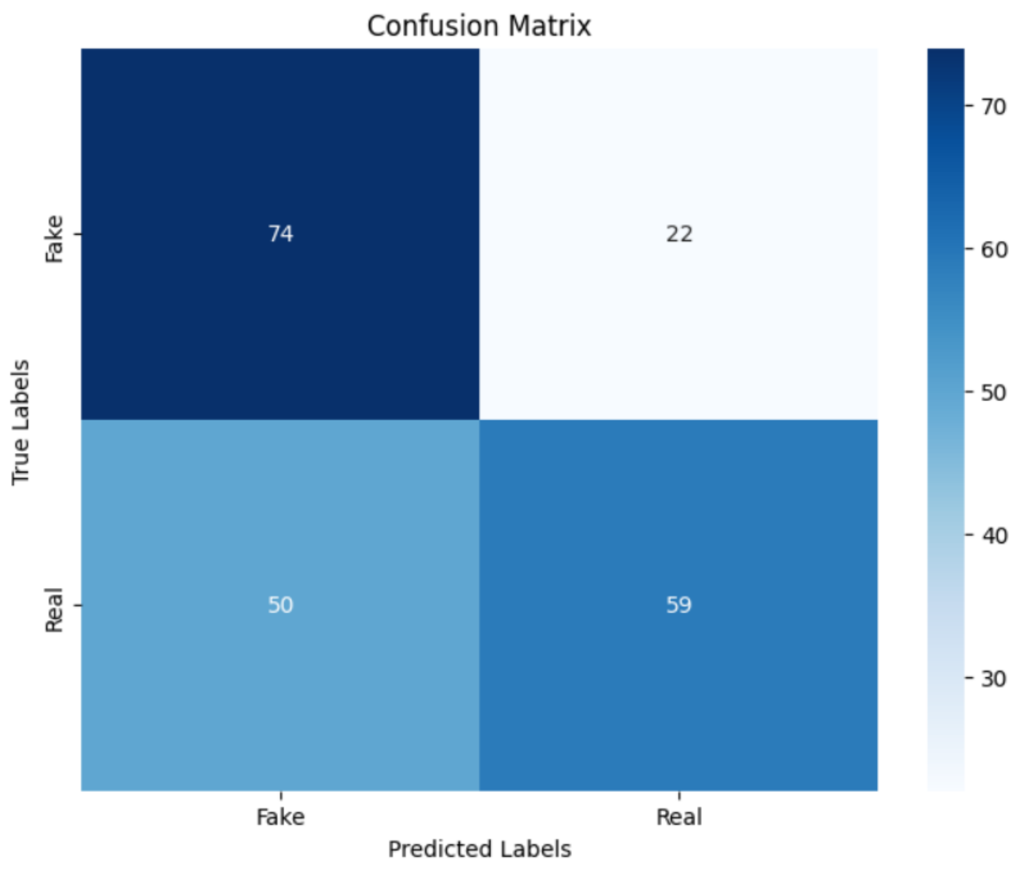
- **VGG16 Transfer Learning** is likely the better model due to its deeper architecture and pre-trained weights. Transfer learning models often perform better on tasks with limited data, leveraging learned features from a large dataset (ImageNet).
- Compared to the custom CNN, the test accuracy is also higher. So it's the better one comparatively.

```
7/7 - 6s - loss: 0.6244 - accuracy: 0.6488 - 6s/epoch - 814ms/step
Test accuracy: 0.6487804651260376
7/7 [==============================] - 2s 259ms/step
              precision    recall  f1-score   support

           0       0.60      0.77      0.67        96
           1       0.73      0.54      0.62       109

    accuracy                           0.65       205
   macro avg       0.66      0.66      0.65       205
weighted avg       0.67      0.65      0.65       205

Confusion Matrix:
[[74 22]
 [50 59]]
```



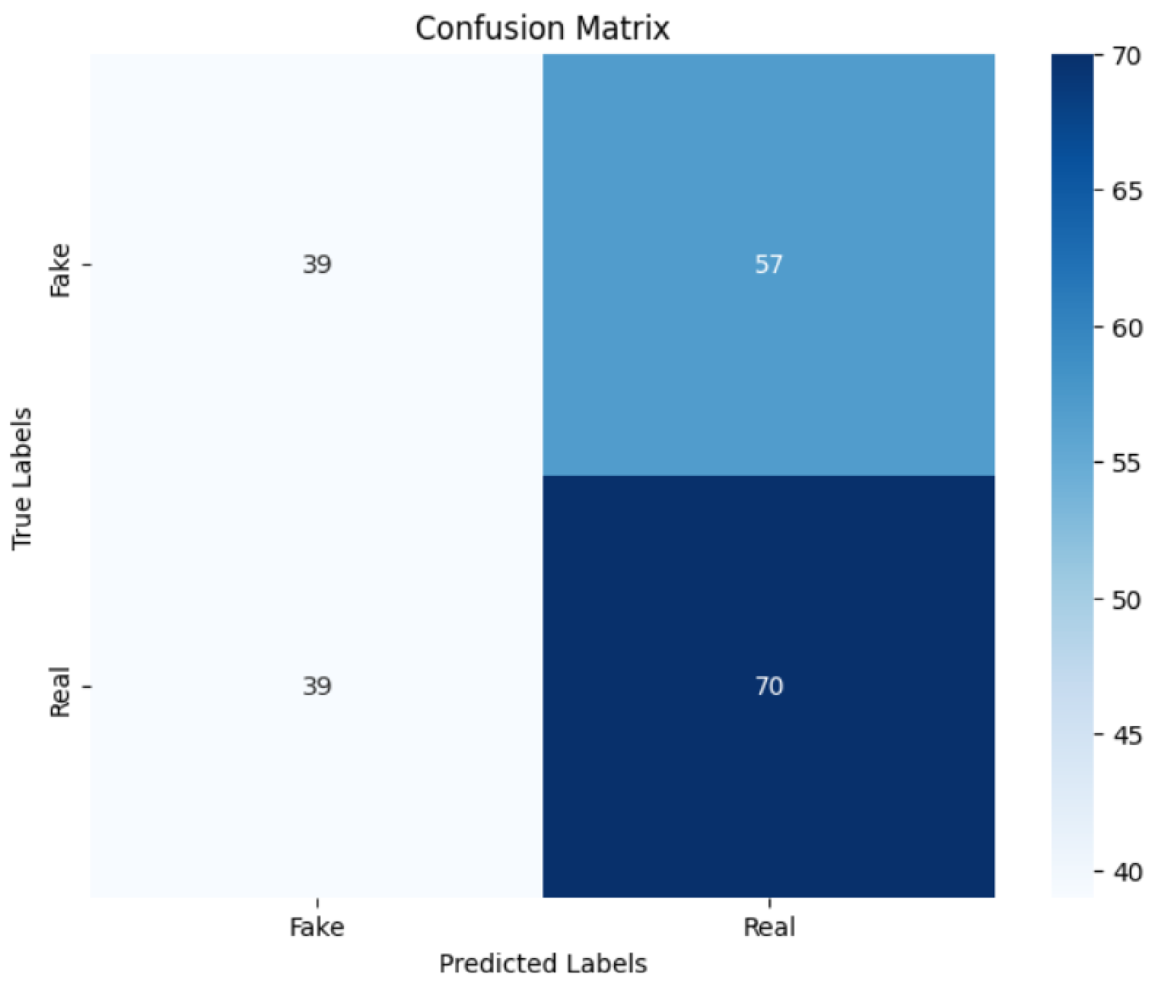
VGG16

```
7/7 - 2s - loss: 0.6798 - accuracy: 0.5317 - 2s/epoch - 265ms/step
Test accuracy: 0.5317073464393616
7/7 [==============================] - 2s 219ms/step
              precision    recall  f1-score   support

           0       0.50      0.41      0.45        96
           1       0.55      0.64      0.59       109

    accuracy                           0.53       205
   macro avg       0.53      0.52      0.52       205
weighted avg       0.53      0.53      0.53       205

Confusion Matrix:
[[39 57]
 [39 70]]
```



<u>Custom CNN</u>

Conclusion

In this project, we compared two deep learning approaches— a custom Convolutional Neural Network (CNN) and the pre-trained VGG16 model through transfer learning— to classify real and fake faces. Our goal was to evaluate the effectiveness of these models in detecting deepfake images, using metrics such as accuracy, confusion matrices, and the F1 score.

The custom CNN was designed from scratch, specifically tailored for our dataset. After training, the model demonstrated a solid ability to differentiate between real and fake faces. The accuracy and F1 score achieved by the custom CNN indicated that it effectively learned the distinguishing features present in the images.

The VGG16 model, pre-trained on the ImageNet dataset and fine-tuned for our specific task, showed superior performance in comparison to the custom CNN. By leveraging the powerful feature extraction capabilities of VGG16, the model achieved higher accuracy and F1 scores. This suggests that transfer learning with a pre-trained model can be highly effective for tasks involving complex image recognition

The confusion matrices for both models revealed that the VGG16 model had fewer false positives and false negatives compared to the custom CNN. This indicates that VGG16 was more reliable in correctly identifying both real and fake faces.

In summary, while the custom CNN showed promising results, the VGG16 model outperformed it in terms of accuracy, F1 score, and reliability. This comparative study highlights the advantages of using transfer learning with a pre-trained model like VGG16 for tasks involving real and fake face detection. Future work could involve exploring other pre-trained models, further fine-tuning, and expanding the dataset to improve the robustness and generalizability of the models. This project contributes to the ongoing efforts to enhance digital security and combat the misuse of deepfake technology.