Nilesh Dusane
2021621
Lakshit Bahl
2046611

# Deepfake Detection

## Vision And Perception


SAPIENZA
UNIVERSITÀ DI ROMA

Submitted to:
Prof. Irene Amerini
Prof. Paolo Russo

# DEEPFAKE DETECTION USING CNN

# OVERVIEW

- Problem Statement
- Approach towards the solution
- Dataset
- Model building and Training , Accuracy
- Conclusion
- Future work

# PROBLEM STATEMENT

- What is Deepfake ?

Deepfake is the artificially created media in the form of images, videos, audio recordings. It potentially alters the contents of the media in convincing manner that a human may not differentiate between deepfake and real.

- How is it a problem?

Deepfake technology poses significant dangers due to its potential for spreading misinformation, manipulating public opinion, compromising individuals' privacy, enabling fraud, and undermining trust in media and democratic processes.

# APPROACH TOWARDS SOLUTION

- The solutions can be many such as – Deepfake Detection tools, Media literacy, Legal impositions, Transparency by creators of such media.
- So, We are building a CNN model that can be used for such purpose to differentiate the real and fake images. This model can be used as baseline to the detection systems of a such kinds.

# DATASET

- The dataset that used is downloaded from Kaggle. It has two classes Real and Fake.
- There are over 190K samples in total and around 95K samples per each class.
- Each sample contains shot where a face of a person visible clearly and the is of the size 256 X 256.
- Real class contains original images and Fake contains morphed(Deepfake) images.

# ARCHITECTURE

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 148, 148, 32)      896

 max_pooling2d (MaxPooling2  (None, 74, 74, 32)        0
 D)

 conv2d_1 (Conv2D)           (None, 72, 72, 64)        18496

 max_pooling2d_1 (MaxPoolin  (None, 36, 36, 64)        0
 g2D)

 conv2d_2 (Conv2D)           (None, 34, 34, 128)       73856

 max_pooling2d_2 (MaxPoolin  (None, 17, 17, 128)       0
 g2D)

 flatten (Flatten)           (None, 36992)             0

 dense (Dense)               (None, 128)               4735104

 dense_1 (Dense)             (None, 1)                 129

=================================================================
```

# MODEL TRAINING & ACCURACY

- We cropped the images further to 150 X 150 to increase the efficiency and reduce training time.
- The data generator for Train, Valid, Test are as follows

```
Found 140002 images belonging to 2 classes.
Found 39428 images belonging to 2 classes.
Found 10905 images belonging to 2 classes.
```

- We train the model for 10 epochs and it took around 40 mins in total.
- As we have only two classes binary crossentropy is used as loss function.
- The validation accuracy is 0.93 and the test accuracy is 0.89

# MODEL ON CUSTOM IMAGES

```
# Load the image
img_path = '/content/Screenshot 2024-02-11 140333.png'
img = image.load_img(img_path, target_size=(img_width, img_height))

# Preprocess the image
img_array = image.img_to_array(img)
img_array = np.expand_dims(img_array, axis=0)
img_array /= 255.

# Make prediction
prediction = model.predict(img_array)
if prediction[0] < 0.5:
    print("Predicted class: Real")
else:
    print("Predicted class: Fake")
```

```
1/1 [==============================] - 0s 313ms/step
Predicted class: Fake
```

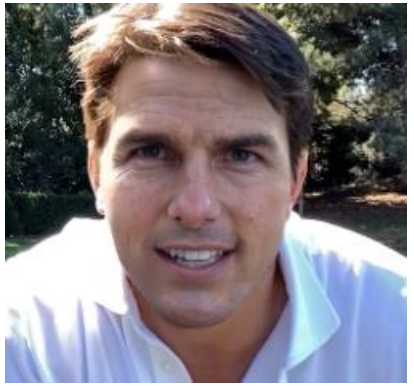
# MODEL ON CUSTOM IMAGES

```
# Load the image
img_path = '/content/Screenshot 2024-02-11 140426.png'
img = image.load_img(img_path, target_size=(img_width, img_height))

# Preprocess the image
img_array = image.img_to_array(img)
img_array = np.expand_dims(img_array, axis=0)
img_array /= 255.

# Make prediction
prediction = model.predict(img_array)
if prediction[0] < 0.5:
    print("Predicted class: Real")
else:
    print("Predicted class: Fake")
```

```
1/1 [==============================] - 0s 26ms/step
Predicted class: Real
```



# REDUCING DATASET

- We tried to add data augmentation as well but this time we reduced each class to 5000 images.
- We train the model for 10 epochs and it took around 10 mins in total.
- As we have only two classes binary crossentropy is used as loss function.
- The validation accuracy is 0.78 and the test accuracy is 0.66
- As we reduced no, of images we also reduced the training time but in turn we lost accuracy significantly.

# CUSTOM MODEL WITH DATA AUGUMENTATION

- With reduced dataset, We did some data augmentation

```
rotation_range=20,
width_shift_range=0.2,
height_shift_range=0.2,
shear_range=0.2,
zoom_range=0.2,
horizontal_flip=True,
fill_mode='nearest')
```

- We trained the model for 10 epochs, and it took around 29 mins in total..
- The validation accuracy is 0.78 and the test accuracy is 0.66

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Fake | 0.70 | 0.57 | 0.63 | 5001 |
| Real | 0.64 | 0.75 | 0.69 | 5001 |
| accuracy | | | 0.66 | 10002 |
| macro avg | 0.67 | 0.66 | 0.66 | 10002 |
| weighted avg | 0.67 | 0.66 | 0.66 | 10002 |

Confusion Matrix

```
[[2859 2142]
 [1240 3761]]
```

# VGG TRANSFER LEARNING WITH DATA AUGUMENTATION

- With reduced dataset, We did some data augmentation

```
rotation_range=20,
width_shift_range=0.2,
height_shift_range=0.2,
shear_range=0.2,
zoom_range=0.2,
horizontal_flip=True,
fill_mode='nearest')
```

- We trained the model for 10 epochs, and it took around 28 mins in total..
- The validation accuracy is 0.87 and the test accuracy is 0.71

Classification Report on Test Data:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Fake | 0.79 | 0.64 | 0.71 | 5001 |
| Real | 0.70 | 0.83 | 0.76 | 5001 |
| accuracy | | | 0.74 | 10002 |
| macro avg | 0.74 | 0.74 | 0.73 | 10002 |
| weighted avg | 0.74 | 0.74 | 0.73 | 10002 |

Confusion Matrix

```
[[3223 1778]
 [ 856 4145]]
```

# EFFICIENTNET TRANSFER LEARNING WITH DATA AUGUMENTATION

- With reduced dataset, We did some data augmentation

```
rotation_range=20,
width_shift_range=0.2,
height_shift_range=0.2,
shear_range=0.2,
zoom_range=0.2,
horizontal_flip=True,
fill_mode='nearest')
```

- We trained the model for 10 epochs, and it took around 24 mins in total..
- The validation accuracy is 0.64 and the test accuracy is 0.61

Classification Report on Test Data:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Fake | 0.69 | 0.44 | 0.54 | 5001 |
| Real | 0.59 | 0.80 | 0.68 | 5001 |
| accuracy | | | 0.62 | 10002 |
| macro avg | 0.64 | 0.62 | 0.61 | 10002 |
| weighted avg | 0.64 | 0.62 | 0.61 | 10002 |

Confusion Matrix

```
[[2202 2799]
 [ 999 4002]]
```

# RESNET TRANSFER LEARNING WITH DATA AUGUMENTATION

- With reduced dataset, We did some data augmentation

```
rotation_range=20,
width_shift_range=0.2,
height_shift_range=0.2,
shear_range=0.2,
zoom_range=0.2,
horizontal_flip=True,
fill_mode='nearest')
```

- We trained the model for 10 epochs, and it took around 28 mins in total..
- The validation accuracy is 0.49 and the test accuracy is 0.50

Classification Report on Test Data:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Fake | 0.50 | 1.00 | 0.67 | 5001 |
| Real | 0.64 | 0.00 | 0.00 | 5001 |
| accuracy | | | 0.50 | 10002 |
| macro avg | 0.57 | 0.50 | 0.33 | 10002 |
| weighted avg | 0.57 | 0.50 | 0.33 | 10002 |

Confusion Matrix

```
[[4997    4]
 [4994    7]]
```

# CONCLUSION

- The model accurately differentiates the fake from real. The accuracy can be increased by more training epochs.
- After data augmentation and using transfer learning with VGG, EfficientNet, ResNet, we found out that VGG had the best results among all in terms of accuracy.

| VGG | Efficient | ResNet | Custom |
|---|---|---|---|
| 0.74 | 0.62 | 0.50 | 0.66 |

# THANK YOU!