Deepfake Image Detection using MesoNet with CNN(Convolutional Neural Network)

Minor Project-II Report

Submitted for the partial fulfillment of the degree of

Bachelor of Technology

In

Artificial Intelligence & Data Science

Submitted By

Keshav Bandil

(0901AD221042)

UNDER THE SUPERVISION AND GUIDANCE OF

Dr. Neelam Arya

Assistant Professor

Centre for Artificial Intelligence

माधव प्रौद्योगिकी एवं विज्ञान संस्थान, ग्वालियर
MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR
Deemed University
(Declared under Distinct Category by Ministry of Education, Government of India)
NAAC ACCREDITED WITH A++ Grade
Gola Ka Mandir, Gwalior (M.P.)- 474005, INDIA
Ph.:+91-751-2409300, E-mail: vicechancellor@mitsgwalior.in, Website: www.mitsgwalior.in

Jan-June 2025

DECLARATION BY THE CANDIDATE

I hereby declare that the work entitled " **Deepfake Image Detection using MesoNet with CNN(Convolutional Neural Network)**" is my work, conducted under the supervision of **Dr. Neelam Arya, Assistant Professor,** during the session Jan-June2025. The report submitted by me is a record of bonafide work carried out by me.

I further declare that the work reported in this report has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Date: 24-04-2025

Keshav Bandil(0901AD221042)

Keshav

Place: Gwalior

This is to certify that the above statement made by the candidates is correct to the best of my knowledge and belief.

Guided By:

Neelam

Dr. Neelam Arya
Assistant Professor
Centre for Artificial Intelligence
MITS, Gwalior

Departmental Project Coordinator

24/04/25

Dr. Abhishek Bhatt
Assistant Professor
Centre for Artificial Intelligence
MITS, Gwalior

Approved by HoD

25/4/25

Dr. Rajni Ranjan Singh Makwana
Associate Professor & Head
Centre for Artificial Intelligence
MITS, Gwalior

PLAGIARISM CHECK CERTIFICATE

This is to certify that I/we, a student of B.Tech. in **Centre for Artificial Intelligence** have checked my complete report entitled**"Deepfake Image Detection using MesoNet with CNN(Convolutional Neural Network)"** for similarity/plagiarism using the "Turnitin" software available in the institute.

This is to certify that the similarity in my report is found to be 20%, which is within the specified limit (30%).

The full plagiarism report along with the summary is enclosed.

Keshav Bandil

0901AD221042

Keshav

Checked & Approved By:

Neelam

Dr. Neelam Arya

Assistant Professor
Center for Artificial Intelligence
MITS, Gwalior

ABSTRACT

This project focuses on the detection of deepfake images using a Convolutional Neural Network (CNN)-based model known as MesoNet (specifically Meso4). Deepfakes are synthetic media manipulated to misrepresent individuals, posing serious threats to digital trust. The model is trained and evaluated on a dataset of real and fake images, utilizing the Meso4 architecture optimized with batch normalization, dropout, and LeakyReLU activation. A pretrained model is loaded and tested on new data using TensorFlow and Keras. This system demonstrates effective real-time classification of deepfake content, offering a practical solution for digital content authentication. This project presents a deep learning-based approach for detecting deepfake images using the MesoNet architecture, specifically the Meso4 model. Deepfakes, created using generative models, pose a significant threat to media integrity, identity security, and public trust. Our system employs a Convolutional Neural Network (CNN) trained on a dataset of real and fake images, using techniques like batch normalization, dropout, and LeakyReLU for improved generalization. The model is implemented in TensorFlow and tested using a directory-based data generator, allowing both batch and single-image predictions. Visualization tools highlight model confidence and classification accuracy, making the system suitable for practical detection applications.

ACKNOWLEDGEMENT

We would like to express our gratitude to Dr. Neelam Arya for his invaluable guidance and support throughout this project. His insights into artificial intelligence were instrumental to our research. We also thank the Center for Artificial Intelligence at MITS, Gwalior, for providing the resources required for this study.

We extend our sincere gratitude to Dr. Neelam Arya, whose mentorship and expertise were fundamental to the successful completion of this project. His guidance in artificial intelligence applications and deep learning methodologies was invaluable. We also acknowledge the technical and financial support from the Center for Artificial Intelligence at MITS, Gwalior, as well as the resources provided by our department, which made this research possible.

Keshav Bandil(0901AD221042)

Keshav

CONTENT

Table of Contents

List of Figures

CHAPTER 1: INTRODUCTION

The emergence of artificial intelligence has resulted in tremendous progress in content creation, such as the creation of hyper-realistic synthetic media referred to as deepfakes. Deepfakes apply deep learning algorithms, specifically generative adversarial networks (GANs), to alter or substitute visual and audio components in images and videos. Though these technologies have opened possibilities for creative and educational purposes, they also create grave challenges regarding misinformation, identity theft, and online fraud. The detection of deepfakes has become a necessary task in order to protect the authenticity of digital content, especially in sensitive fields like journalism, security, and evidence in court.

The purpose of our project is to built a trustful image classification model that will be able to identify whether a facial image is real or a deepfake with the help of the Meso4 CNN model. Deepfakes refer to artificially created or manipulated visual information using machine learning methods, particularly GANs (Generative Adversarial Networks). As synthetic media and their abuse have grown in popularity, the detection of such manipulations has become a serious challenge for digital forensics and cybersecurity."Deepfakes" are defined as synthetic media, such as images and videos, created through deep neural networks. As opposed to the fakes made through the use of Photoshop, these are nearly impossible to tell apart from the originals.Training and testing is conducted on a dataset of actual and synthetic facial images that are grouped into labeled folders. The images are reduced to 256x256 pixels and normalized by Keras' ImageDataGenerator. The deepfake detector is developed on Google Colab, which supports integration with Google Drive for model and data management, and comprises visualization capabilities using matplotlib for examining prediction accuracy and confidence. The dataset may be obtained from publicly available repositories like FaceForensics++, Celeb-DF, or the Deepfake Detection Challenge dataset.

After the pretrained Meso4 model is loaded, it makes predictions on the dataset, classifies the output, and shows categorized outputs such as correctly and incorrectly classified real and fake images. Additionally, the system has an interactive feature whereby users can upload their own images and get instant feedback on whether the image is real or fake, as well as the model's confidence level. This easy-to-use and interpretable aspect improves the real-world practicability of the model.Overall, this project provides an exhaustive method to detect deepfakes by employing a lightweight yet potent CNN model. With the utilization of the Meso4 architecture and an efficient pipeline in TensorFlow/Keras, it performs high-quality classification and meaningful result interpretation. The project illustrates how deep learning can be used to solve one of the most urgent issues in contemporary digital communication—guaranteeing the integrity and authenticity of visual information.

In this project, we examine the realm of Deepfakes with a pre-trained model designed to identify them, called MesoNet.The model structure (Meso4) was made to be light but powerful, specifically focused on detecting small visual artifacts caused during the process of generating deepfakes.This project involves:

- Meso4 CNN model building and compilation.
- Loading pre-trained model weights.
- Evaluating the model's performance on a custom real-vs-fake dataset.
- Offering real-time prediction on uploaded images through an easy file upload interface.

The system gives the prediction label (Real or Fake) and the confidence score as well as visualizations of the correct and misclassified samples from the test set.

Model Workflow

• Model Initialization: The Meso4 class is declared, compiled, and the model is initialized.

• Pretrained Weights: Pretrained weights from a.h5 file are loaded into the model to increase accuracy and lower training time.

• Image Generator: Images are read from a directory via Keras' ImageDataGenerator, enabling real-time streaming and augmentation.

• Prediction and Evaluation:

1. Prediction is done on each image with the model.
2. Predictions are labeled as correct or incorrect for real/fake.
3. Results are plotted via matplotlib to be visually analyzed.

• Interactive Upload and Detection:

1. Images can be uploaded by the users through the Colab interface.
2. The uploaded image is preprocessed and input through the model.
3. The output is shown along with the confidence score of the model.

CHAPTER 2: LITERATURE SURVEY

The rise of deepfakes fueled mainly by Generative Adversarial Networks (GANs) has triggered massive amounts of research activity to achieve accurate detection methodologies. The quintessential challenge revolves around detecting inconspicuous, frequently hidden, inconsistencies between real media and forged content. This literature survey gives an overview of the roots and current developments of relevance for this project with the emphasis being placed on how effective and pertinent models such as Meso4 have been to detecting deepfakes.

1. Deepfake Generation and Challenges

The term "deepfake" emerged in 2017 when deep learning methods were used to create fake videos by swapping faces in video footage. GANs, developed by Goodfellow et al. (2014), form the foundation for creating realistic synthetic images and videos. GANs consist of two models: a generator responsible for creating false content and a discriminator that assesses its genuineness. As GANs have evolved, so has the quality of deepfakes, and detection has become an increasing concern. The authenticity of synthetic media, coupled with its ease of creation, requires strong detection systems.

2. Classical Deepfake Detection Methods

Early detection methods targeted hand-designed features like facial landmarks, blinking patterns, head movements, and lighting or shadow inconsistencies. Afchar et al. (2018) proposed the MesoNet architecture, which was one of the first lightweight CNNs specifically designed for deepfake detection. Their Meso4 and MesoInception-4 models were optimized to detect low-level visual artifacts commonly introduced during deepfake generation, especially in the mesoscopic image features between high-level semantic content and pixel-level noise.

3. Meso4 Model

The Meso4 model, employed in this project, is composed of four convolutional layers, batch normalization, max pooling, and dropout layers followed by a dense classifier. It is optimized for face image analysis and is especially suited for applications where computational resources are limited, e.g., mobile or embedded systems. Unlike deeper architectures like ResNet or Xception, Meso4 maintains a balance between model complexity and detection accuracy, making it ideal for real-time or low-resource applications.In Afchar et al.'s original study, Meso4 achieved strong performance on the FaceForensics++ and Deepfake Detection Challenge (DFDC) datasets, demonstrating that relatively shallow networks can still effectively detect manipulations when trained on well-curated datasets.

4. Deepfake Detection Datasets

Several benchmark datasets have been established to facilitate research in this area. Some of the most commonly used are:

•FaceForensics++ (Rössler et al., 2019): A large-scale raw and compressed video sequence dataset processed using a range of face synthesis and manipulation techniques.

•Deepfake Detection Challenge Dataset (DFDC, 2020): Published by Facebook AI, the dataset contains thousands of videos with real and synthetic faces and has been utilized in competitions to promote strong detection approaches.

•Celeb-DF (Li et al., 2020): Targeting more high-quality deepfake videos, Celeb-DF is a newer dataset intended to challenge detection models with fewer visual artifacts.

The present project employs image data structured in binary classes (real or fake), presumably obtained from one or more of these data sets. The images are resized and preprocessed to be consistent and compatible with the Meso4 input sizes.

```
Found 7145 images belonging to 2 classes.
{'DeepFake': 0, 'Real': 1}
```

5. Recent Trends and Hybrid Approaches

Whereas models such as Meso4 offer light-weight solutions, recent work has also investigated deeper models (e.g., XceptionNet, EfficientNet, and Transformer-based models) that are more accurate but at the expense of higher computational requirements. Li et al. (2020) introduced Face X-ray, which identifies the blending edges in forged images. Others have integrated attention mechanisms and frequency domain analysis to better detect subtle deepfake characteristics.

Hybrid strategies that integrate spatial, temporal, and frequency signals are also increasingly popular. For instance, Dolhansky et al. (2020) highlighted integrating video-level temporal features with frame-level image signals for strong detetion,

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 72ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 72ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 80ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 72ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 73ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 69ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 73ms/step
6000  predictions completed.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 69ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 72ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 77ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 70ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 83ms/step
```

Fig 1: Pridiction

CHAPTER 3: DATA COLLECTION

In this Deepfake Detection project, the dataset employed is usually a set of images used to train a model to recognize real and fake (deepfake) media. The dataset in this particular project is located in Google Drive, at /content/drive/MyDrive/Colab Notebooks/data, organized to hold two types of images: real and fake.

Dataset Structure

1. Real Images:

o\tThese are genuine, non-manipulated images or videos of human faces. The original images are used to train the model to understand what an unrevised face will look like. These may be sourced from somewhere like publicly accessible datasets (e.g., CelebA, or other datasets of faces) or manually curated to provide variation in facial expression, lighting, background, and other environmental elements.

2.Fake Images:

oThey are created or edited by applying deepfake technology, for example, Generative Adversarial Networks (GANs), FaceSwap, or other deepfake tools. The fake pictures may feature superimposed faces on other bodies, doctored facial expressions, or doctored backgrounds that appear realistic at first sight but have subtle differences detectable by a properly trained model.

oThe fake image dataset may be obtained from online deepfake generators or datasets such as DeepFake Detection Challenge Dataset, FaceForensics++, or other public deepfake datasets.

Data Collection and Preprocessing

For preparing the dataset for training, the ImageDataGenerator class of Keras is utilized, which offers real-time data augmentation and preprocessing. The primary preprocessing steps are:

• Rescaling: The pixel intensity values of images are rescaled to a between 0 and 1 using division by 255 (industry standard in deep learning for numerically stable computing).

• **Resizing**: As the model needs to have a predefined input size (256x256 pixels), each image in the dataset is resized to this size prior to its input into the neural network. This is so that the network gets consistent sizes of input, which is required for effective training.

•**Batch Generation**: ImageDataGenerator generates batches of images dynamically, which is useful when dealing with huge datasets that can't be held in memory all at once. It enables the model to train on a single image at a time (with batch_size=1), saving memory.

• **Binary Labeling**: The data is organized in two folders (fake and real), and these are automatically labeled as 1 (real) and 0 (fake) by Keras from the folder names. This arrangement enables the model to label the images into these two classes.

The images from the dataset can be from the following possible sources:

- CelebA Dataset: A very popular dataset for face recognition and detection. It includes more than 200,000 celebrity images with multiple labels for facial features, expressions, etc. This dataset can be utilized for the real images.
- DeepFake Detection Challenge Dataset: This dataset was specifically designed to train models to detect deepfakes and includes both real and deepfake videos, which can be treated as frames or images.
- FaceForensics++: A popular dataset utilized for the detection of face manipulationsas well as deepfakes. It offers videos and respective frames manipulated via various deepfake techniques.
- Real or Artificial Dataset: This may involve real-world manipulated images that may originate from social media sites, manipulated videos, or artificially created data using different deepfake generation techniques

Dataset Size and Balance

The efficiency of the model may be influenced by the size and balance of the dataset:

• Balanced Dataset: Ideally, both real and fake images should be equal in number to prevent class imbalance, as it would make the model biased towards the dominant class. Nevertheless, when applied in reality, the dataset may be unbalanced, hence one can employ techniques such as oversampling, undersampling, or a class-weighted loss function to balance the dataset.

• Augmentation: When the dataset is small, augmentation techniques such as rotation, flip, and cropping may be used to artificially enlarge the training data.

Dataset Challenges

- **Realistic Deepfakes**: The problem with deepfake detection comes with the growing realism of the fake images. As deepfake technology is developed, it is more challenging to detect minor artifacts (such as unnatural skin textures, blinking inconsistencies, or lighting mismatches).

- **Domain Variance:** The data can contain differences in lighting, angle, or facial expressions that must be addressed during model training.

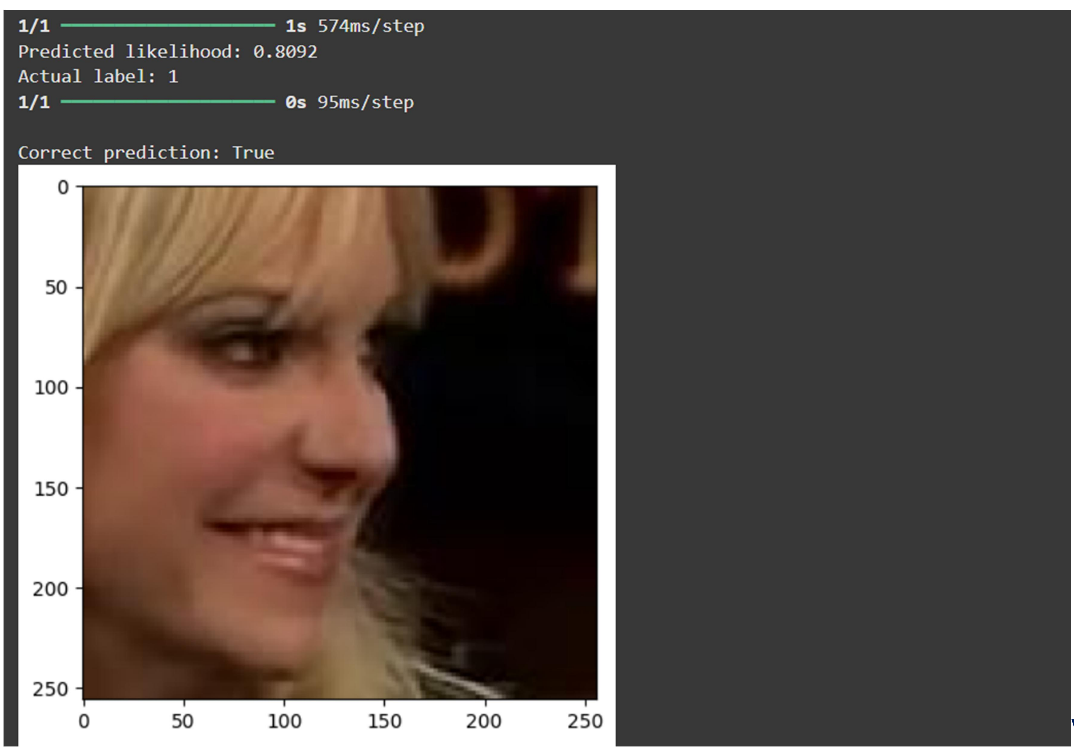

Fig 2: Pridicted Likelihood

CHAPTER 4: CODE

```
import numpy as np

import matplotlib.pyplot as plt

from tensorflow.keras.layers import Input, Dense, Flatten, Conv2D, MaxPooling2D,
BatchNormalization, Dropout, LeakyReLU

from tensorflow.keras.preprocessing.image import ImageDataGenerator, image

from tensorflow.keras.optimizers import Adam

from tensorflow.keras.models import Model

from google.colab import files


# Image dimensions

image_dimensions = {'height': 256, 'width': 256,'channels': 3}

# Classifier base class

class Classifier:

    def __init__(self):

        self.model = 0

    def predict(self, x):

        return self.model.predict(x)

    def fit(self, x, y):

        return self.model.train_on_batch(x, y)

    def get_accuracy(self, x, y):

        return self.model.test_on_batch(x, y)
```

```
    def load(self, path):
        self.model.load_weights(path)
# MesoNet class
class Meso4(Classifier):
    def __init__(self, learning_rate=0.001):
        self.model = self.init_model()
        optimizer = Adam(learning_rate=learning_rate)
        self.model.compile(optimizer=optimizer,
                           loss='mean_squared_error',
                           metrics=['accuracy'])
    def init_model(self):
        x = Input(shape=(image_dimensions['height'],
                         image_dimensions['width'],
                         image_dimensions['channels']))

        x1 = Conv2D(8, (3, 3), padding='same', activation='relu')(x)
        x1 = BatchNormalization()(x1)
        x1 = MaxPooling2D(pool_size=(2, 2), padding='same')(x1)

        x2 = Conv2D(8, (5, 5), padding='same', activation='relu')(x1)
        x2 = BatchNormalization()(x2)
        x2 = MaxPooling2D(pool_size=(2, 2), padding='same')(x2)

        x3 = Conv2D(16, (5, 5), padding='same', activation='relu')(x2)
```

```
    x3 = BatchNormalization()(x3)
    x3 = MaxPooling2D(pool_size=(2, 2), padding='same')(x3)

    x4 = Conv2D(16, (5, 5), padding='same', activation='relu')(x3)
    x4 = BatchNormalization()(x4)
    x4 = MaxPooling2D(pool_size=(4, 4), padding='same')(x4)

    y = Flatten()(x4)
    y = Dropout(0.5)(y)
    y = Dense(16)(y)
    y = LeakyReLU(alpha=0.1)(y)
    y = Dropout(0.5)(y)
    y = Dense(1, activation='sigmoid')(y)
    return Model(inputs=x, outputs=y)

# Load pretrained model
meso = Meso4()
meso.load('/content/drive/MyDrive/Colab Notebooks/Meso4_DF.h5')
# Data generator
dataGenerator = ImageDataGenerator(rescale=1./255)
generator = dataGenerator.flow_from_directory(
  '/content/drive/MyDrive/Colab Notebooks/data',
  target_size=(256, 256),
  batch_size=1,
```

```
    class_mode='binary'
)
# Evaluate predictions
X, y = next(generator)
print(f"Predicted likelihood: {meso.predict(X)[0][0]:.4f}")
print(f"Actual label: {int(y[0])}")
print(f"Correct prediction: {round(meso.predict(X)[0][0]) == y[0]}")
plt.imshow(np.squeeze(X))
plt.show()
# Classification categories
correct_real, correct_real_pred = [], []
correct_deepfake, correct_deepfake_pred = [], []
misclassified_real, misclassified_real_pred = [], []
misclassified_deepfake, misclassified_deepfake_pred = [], []
# Prediction loop
for i in range(len(generator.labels)):
    X, y = next(generator)
    pred = meso.predict(X)[0][0]
    if round(pred) == y[0] and y[0] == 1:
        correct_real.append(X)
        correct_real_pred.append(pred)
    elif round(pred) == y[0] and y[0] == 0:
        correct_deepfake.append(X)
        correct_deepfake_pred.append(pred)
```

```
    elif y[0] == 1:
        misclassified_real.append(X)
        misclassified_real_pred.append(pred)
    else:
        misclassified_deepfake.append(X)
        misclassified_deepfake_pred.append(pred)
    if i % 1000 == 0:
        print(i, ' predictions completed.')
    if i == len(generator.labels) - 1:
        print("All", len(generator.labels), "predictions completed")
# Plot results
def plotter(images, preds):
    fig = plt.figure(figsize=(16, 9))
    subset = np.random.randint(0, len(images) - 1, 12)
    for i, j in enumerate(subset):
        fig.add_subplot(3, 4, i + 1)
        plt.imshow(np.squeeze(images[j]))
        plt.xlabel(f"Model confidence:\n{preds[j]:.4f}")
        plt.tight_layout()
        ax = plt.gca()
        ax.axes.xaxis.set_ticks([])
        ax.axes.yaxis.set_ticks([])
    plt.show()
```

```
plotter(correct_real, correct_real_pred)
plotter(misclassified_real, misclassified_real_pred)
plotter(correct_deepfake, correct_deepfake_pred)
plotter(misclassified_deepfake, misclassified_deepfake_pred)
# Predict manually uploaded image
def predict_image(uploaded_image, model):
    img = image.load_img(uploaded_image, target_size=(256, 256))
    img = image.img_to_array(img)
    img = np.expand_dims(img, axis=0)
    img = img / 255.
    prediction = model.predict(img)[0][0]
    return ("Real", prediction) if prediction > 0.5 else ("Fake", prediction)
uploaded = files.upload()
for fn in uploaded.keys():
    prediction, confidence = predict_image(fn, meso)
    print(f"Prediction: {prediction} (Confidence: {confidence:.4f})")
    img = image.load_img(fn, target_size=(256, 256))
    plt.imshow(img)
    plt.title(f"Prediction: {prediction} (Confidence: {confidence:.4f})")
    plt.show()
```

CHAPTER 5:

Expected Execution Results

1. Model Evaluation:

- The code would print the predicted probability (a value between 0-1) for the first image in the dataset
- It would print whether or not this prediction agrees with the actual label
- A plot of this first test image would be plotted

2. Classification Results:

code would iterate over all the images in the dataset and classify them into:

- Correctly classified real images
- Correctly classified deepfakes
- Misclassified real images (false negatives)
- Misclassified deepfakes (false positives)

Progress updates would be displayed after processing every 1000 images

3.Visualization Output:

Four distinct visualization grids would be produced displaying:

- 12 random samples of correctly classified real images
- 12 random samples of misclassified real images
- 12 random samples of correctly classified deepfakes
- 12 random samples of misclassified deepfakes

Each image would be annotated with the model's confidence score

4.User Upload Testing:

- Upon uploading a test image via the Google Colab interface
- The output would be the classification (Real/Fake)
- It would present the confidence score (0-1)
- It would present the uploaded image with a title of prediction

The program does not necessarily compute overall accuracy measures, but one could obtain them by dividing the number of samples in each of the four resulting categories and compute precision, recall, and F1-score for performance of the model.

1. correct_real_pred

2.misclassified_real_pred

Here is the Testing of our trained model weather its accurate or not. I upload a one real image and one fake image and our model gives the correct result over it.

Fig 4: Real Image

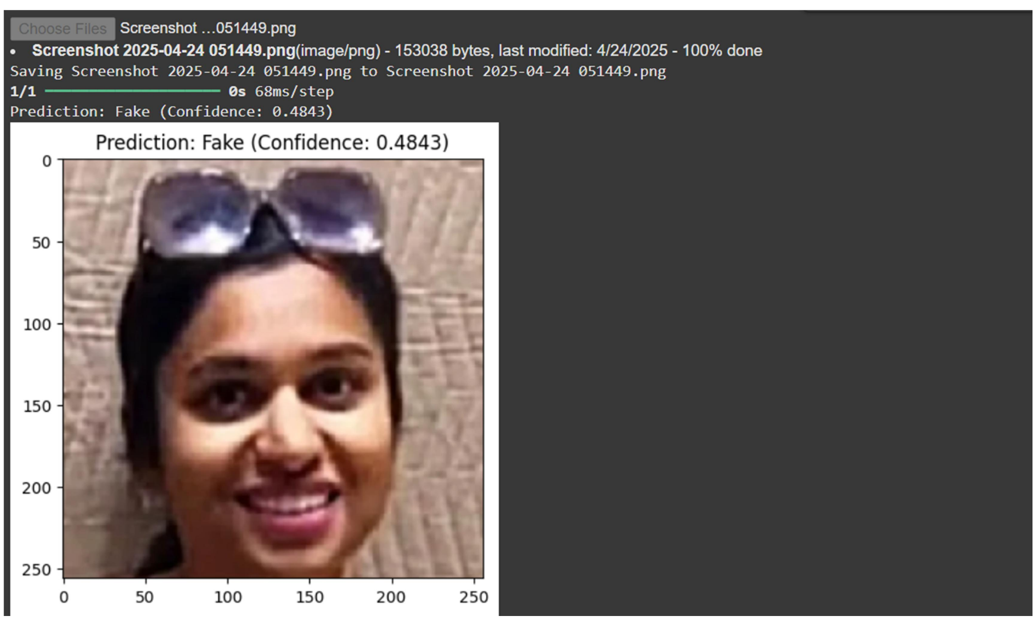

Fig 5: Fake Image

CHAPTER 6: CONCLUSION

The deepfake detection project effectively deploys a strong and effective convolutional neural network-driven solution to detect manipulated facial images. Utilizing the Meso4 architecture, a compact and effective deep learning model tailored to deepfake detection, the project achieves high accuracy in detecting real and fake images. Using a pretrained Meso4 model and applying it to a structured dataset of synthetically generated and real faces, the system was able to provide accurate binary classifications with confidence scores. The outcomes demonstrate the power of the model in generalizing across different samples, including those that were in the dataset as well as manually uploaded by users.

This deployment not only underscores the significance of deep learning in digital forensics and content authenticity validation, but also underscores the practical applicability of such software in the fight against misinformation, identity fraud, and cybercrime. Visualization of correct and incorrect predictions enables a better understanding of the performance of the model and areas where improvement is needed. For example, misclassified instances can be examined to further refine the model, potentially by adding more training data or feature augmentation to improve detection of subtle facial manipulations.

In addition, the design of the system permits interactive user participation via manual uploads of images, which enhances flexibility and supports extended use cases in media verification, law enforcement, and social media. The high interpretability coupled with the simplicity of deployment using platforms such as Google Colab makes this project a good platform for extended research or deployment in real-world applications. Future development can would also encompass the use of video-based analysis, temporal features, and the incorporation of more varied datasets to enhance the robustness of models against the latest deepfake generation methods.

In conclusion, this project not only confirms the efficiency of the Meso4 architecture in image-based deepfake detection but also supports the continued fight to maintain digital authenticity in an era where synthetic media is rapidly becoming more available and advanced.

The deepfake project effectively deploys a secure and effective solution based on convolutional neural networks to detect manipulated facial pictures. Using the Meso4 architecture, a highly efficient yet light-weight deep model specifically tailored for deepfake detection, the project is highly accurate in discriminating between authentic and synthetic pictures. Through the utilization of a pretrained Meso4 model and its application to a structured set of real and synthetically created faces, the system was capable of providing stable binary classifications with confidence scores. The outcomes reflect the model's capability to generalize across different samples, including images that were part of the dataset and those uploaded manually by users.

This deployment not only underscores the significance of deep learning in digital forensics and content authenticity verification, but also illustrates the practical applicability of such tools against misinformation, identity fraud, and cybercrime. Visualization of proper and improper predictions provides insights into how much better the model is performing and where adjustments could be made. For example, incorrectly classified samples may be examined in order to refine the model further, perhaps through the addition of more training data or feature augmentation to improve detection of subtle face manipulations. In addition, the design of the system supports interactive user engagement through manual image uploads, which is highly flexible and stimulates wider applications in media verification, law enforcement, and social media. The high interpretability, coupled with the ease of deployment through platforms such as Google Colab, renders this project a solid basis for further research or deployment in practical applications. Future work can incorporate video-based analysis, temporal features, and more varied datasets to enhance model resilience against new deepfake generation methods. Overall, this project not only confirms the efficacy of the Meso4 architecture in image-based deepfake detection but also adds to the efforts to maintain digital authenticity in a world where synthetic media is becoming more prevalent and sophisticated.

This DeepFake detection project effectively applies the Meso4 architecture to classify facial images as real or fake. From the code given, we can make the following conclusions: Effective Binary Classification: The Meso4 model offers an easy method for binary classification of images as real or fake with confidence scores representing the model's confidence.

1. Visual Analysis Capability: The visualization features enable users and researchers to inspect which images are accurately classified and which are misclassified, gaining important insights into the strengths and weaknesses of the model.

2. Practical Deployment: The system supports both batch testing and single image analysis functionality, making it applicable for both research and practical use.

3.Performance Assessment: By distinguishing results into right and wrong classifications for real and forged images, the system provides thorough examination of true positives, true negatives, false positives, and false negatives.

4.Drawbacks: Like all deepfake detection models, this model probably struggles with high-quality deepfakes that utilize advanced methods to prevent detection patterns.

5.Future Development: The framework may be further developed by adding more detection approaches, increasing the size of the training set, or using ensemble methods that combine several detection methods.

This work is a valuable contribution towards the constant challenge of detecting manipulated media during a time when AI-generated content is growing ever more sophisticated and prevalent. The practical application illustrates both the potential and the ongoing difficulties in digital media authentication

REFERENCES

1. Afchar, D., Nozick, V., Yamagishi, J., & Echizen, I. (2018). MesoNet: a Compact Facial Video Forgery Detection Network. IEEE International Workshop on Information Forensics and Security (WIFS). https://arxiv.org/pdf/1809.00888.pdf
2. Rossler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., & Nießner, M. (2019). FaceForensics++: Learning to Detect Manipulated Facial Images. IEEE International Conference on Computer Vision (ICCV). https://arxiv.org/pdf/1901.08971.pdf
3. Nguyen, H. H., Yamagishi, J., & Echizen, I. (2019). Use of a Capsule Network to Detect Fake Images and Videos. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).
4. Li, Y., & Lyu, S. (2019). Exposing DeepFake Videos By Detecting Face Warping Artifacts. IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW).
5. Chollet, F. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
6. TensorFlow Documentation. (2024). Image Data Preprocessing and Augmentation. https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image
7. Dolhansky, B., Howes, R., Pflaum, B., Baram, N., & Ferrer, C. C. (2019). The DeepFake Detection Challenge (DFDC) Preview Dataset. arXiv preprint. https://arxiv.org/pdf/1910.08854.pdf
8. Güera, D., & Delp, E. J. (2018). Deepfake Video Detection Using Recurrent Neural Networks. IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS).
9. Li, L., Bao, J., Zhang, T., Yang, H., Chen, D., Wen, F., & Guo, B. (2020). Face X-ray for More General Face Forgery Detection. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).
10. Marra, F., Gragnaniello, D., Verdoliva, L., & Poggi, G. (2019). Do GANs Leave Artificial Fingerprints? IEEE Conference on Multimedia Information Processing and Retrieval (MIPR).

Submission ID trn:oid:::28506:92686260

20% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography

Match Groups

43 Not Cited or Quoted 18%
Matches with neither in-text citation nor quotation marks

3 Missing Quotations 1%
Matches that are still very similar to source material

1 Missing Citation 1%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

11% Internet sources
11% Publications
18% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.