TOWARDS ENHANCING THE PERFORMANCE OF DEEP FAKE DETECTION SYSTEM

A PROJECT REPORT
PHASE 2

Submitted by

THAHIR IRFAN S (21BEC165)
SANJAYKUMAR S (21BEC214)
SHAGAR SHREE RAJA S L (21BEC133)

in partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING
in
ELECTRONICS AND COMMUNICATION ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY
(An Autonomous Institution affiliated to Anna University, Chennai)
Post Box No: 2034, Coimbatore - 641049

APRIL 2025

KUMARAGURU COLLEGE OF TECHNOLOGY

Coimbatore - 641049

ELECTRONICS AND COMMUNICATION ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report **"TOWARDS ENHANCING THE PERFORMANCE OF DEEP FAKE DETECTION SYSTEM"** is the Bonafide work of **"THAHIR IRFAN S (21BEC165), SANJAYKUMAR S (2BEC214), SHAGAR SHREE RAJA S L (21BEC133) "** who carried out the project work under my supervision.

SIGNATURE
Dr. S. ARUN KUMAR
SUPERVISOR
Assistant Professor
Electronics and Communication Engineering.
Kumaraguru College of Technology

SIGNATURE
Dr. M. BHARATHI
HEAD OF THE DEPARTMENT
Professor
Electronics and Communication Engineering
Kumaraguru College of Technology

The candidates with college roll/ register number 21BEC165, 21BEC214, 21BEC133 were examined in the Project Viva-Voce examination held on 22/04/2025.

Internal Examiner

External Examiner

ACKNOWLEDGEMENTS

First, we would like to express our praise and gratitude to the Lord, who has showered his grace and blessings enabling us to complete this project in an excellent manner.

We express our sincere thanks to the management of Kumaraguru College of Technology and President **Shri. Shankar Vanavarayar,** for his kind support and for providing necessary facilities to carry out the work.

We would like to express our sincere thanks to our beloved Principal **Dr. Ezhilarasi M,** Kumaraguru College of Technology, who encouraged us with his valuable thoughts.

We would like to thank **Dr. Bharathi M,** Head of the Department, Electronics and Communication Engineering, for her kind support and for providing the necessary facilities to carry out the project work. We wish to thank everlasting gratitude to the project coordinator **Ms. Pavithra P,** Department of Electronics and Communication Engineering, for her consistent support throughout the course of this project work.

We are privileged to express our heartfelt thanks to our project guide, **Dr. Arun Kumar S**, Assistant Professor, Department of Electronics and Communication Engineering, for his expert counseling and guidance to make this project a great deal of success.

Finally, we thank our parents, family members and friends for giving us moral support and abundant blessings in all our activities and helped us to endure our challenging times with their support and warm wishes.

TABLE OF CONTENTS

I. ABSTRACT

With the fast growth of artificial intelligence, it has become a significant challenge to differentiate between authentic and manipulated digital content. Deep fakes refer to complex algorithms that create realistic fake images, videos, or audio. They pose threats to information integrity, cybersecurity, and public trust. This work addresses the growing need for robust detection mechanisms to combat deep fake technologies. A system for deep fake detection for image and video is proposed in this work. For deep fake image detection, Vision Transformer (ViT) model is used for global feature extraction and EfficientNet model is used for local feature extraction and for deep fake video detection, Convolutional Neural Networks (CNN) model is used for local feature extraction. The results are then combined and used for classification process. Further, a comparative study is presented for different types of classifiers. Among the various classifiers implemented, Sigmoid SVM showed the best accuracy (85.97%) for deep fake image detection. For the deep fake detection video the accuracy (98.74%). The models have been designed using public datasets. Metrics such as Accuracy, Loss, Precision, Sensitivity (recall), Specificity, F1-score, MCC, BCR and Cohen-Kappa are evaluated for the proposed system.

Keywords – Deep Fake Detection, Convolutional Neural Networks (CNN), Vision Transformer (ViT), EfficientNet, Data Preprocessing, Performance Metrics.

II. LIST OF ABBREVIATIONS

S. NO.	ABBREVIATION	EXPANSION
1	CNN	Convolutional Neural Network
2	DL	Deep Learning
3	ML	Machine Learning
4	ViT	Vision Transformer
5	AI	Artificial Intelligence
6	MCC	
7	BCR	
8	FP	False Positive
9	FN	False Negative
10	TP	True Positive
11	TN	True Negative
12	SVM	Support Vector Machine
13	RBF	

III. LIST OF FIGURES

IV. LIST OF TABLES

CHAPTER 1
INTRODUCTION

The project aims to develop an extremely efficient and robust deep fake detection system based on the advanced deep learning techniques to be implemented for both image as well as video content. This project intends to implement CNN, Vision Transformers and EfficientNet to achieve the goal of an effective detection system by detecting manipulated images and video. For image, the performances of ViT and EfficientNet models have been evaluated to reveal effective identification of deep fakes.

For video deep fake detection, it converts videos into frames and then applies CNN models, which is then evaluated to reveal effective identification of deep fakes. With it, the idea is to achieve high accuracy in both robustness and the ability to tell apart the real from the fake by completely pre-processing the data, training the models, and fine-tuning them. The efficiency of the models is assessed by evaluating performance metrics in terms of accuracy, loss, precision, sensitivity (recall), specificity F1-score, MCC, BCR and Cohen-Kappa. Comparative studies will be conducted between ViT and EfficientNet (CNN) on the image while applying CNN to the video model for deep fake detection potential.

1.1 RESEARCH OBJECTIVE

The first objective is to improve the detection accuracy and robustness of the model. By developing an advanced deep learning model, higher accuracy in detecting steps can be achieved, reducing both false positive and false negative rates. The second objective is to use a wide range of datasets. A diverse dataset will be used for training, validating and testing the model, ensuring improved performance. Lack of dataset's quality and quantity is one of the common issues in the poor performance of the model. For the project both image and video dataset are required. Hence, by increasing quantity and quality of the dataset, high performance can be achieved.

1.2 SOCIAL RELEVANCE

Deep fake technology poses deep challenges to the integrity of digital media, cybersecurity, and public trust. Therefore, with this kind of content, risks such as spreading misinformation, acts of political manipulation, and defamation are going to be very serious. Truth and accountability are starting to become very crucial in a world where any communication now depends on multimedia.

This project is highly relevant to societies; it directly meets the growing need to have highly effective tools intended for the detection of deep fakes. Models for image and video deep fake detection will improve digital forensics, media verification, and online content moderation. It is going to empower the individual, organization, and government to fight disinformation, protect privacy, and keep authenticity in digital media intact.

1.3 RESEARCH FLOW

The research starts with collecting the datasets. The datasets are pre-processed by rescaling, augmenting and normalizing. Then the preprocessed datasets are fed to deep learning models such as ViT and EfficinetNet for further processing. In the case of video datasets, the videos are converted to frames and then preprocessed and fed to the deep learning model. Optimization and Hyperparameter tuning are performed, and the model is trained, validated and tested for obtaining the performance metrics.

1.4 ORGANIZATION OF THESIS

The thesis started with building a Vision Transformer (ViT) model for deep fake image detection. Then we built a EfficinetNet model for the purpose of detecting deep fake images. Both the models are trained, validated and tested with a wide range of datasets and the performance metrics are analyzed. For deep fake audio detection, a Convolutional Neural Networks (CNN) model is built and the video datasets used are converted to frames. Later the model is trained, developed and evaluated. Also, the performance metrics are measured.

CHAPTER 2

LITERATURE SURVEY

1. **Deepfake Video Detection: Challenges and Opportunities (2024):**

   This paper explores a hybrid approach using CNN, RNN, and GAN to extract spatial and temporal features from videos. It achieved a high accuracy of 90% but struggles with poor generalization and high computational complexity.
   Algorithm: CNN, RNN, and GAN.
   Accuracy: 90%.
   Limitation: High computational cost and poor generalization.

2. **Deepfake Video Detection through Optical Flow-Based CNN (2019):**

   Using optical flow-based CNN, this paper detects inter-frame motion inconsistencies to identify deepfakes. While it provides a fair accuracy of 75.46%, the method is sensitive to dataset size and requires fine-tuning.
   Algorithm: Optical flow-based CNN.
   Accuracy: 75.46%.
   Limitation: Sensitive to dataset size, needs fine-tuning.

3. **A Survey on Deepfake Video Detection (2021):**

   This survey highlights deepfake detection using CNN, RNN, and Transformer by analyzing spatial, temporal, and physiological inconsistencies. It recorded a relatively lower accuracy of 65%, mainly due to generalizability issues across datasets.
   Algorithm: CNN, RNN, and Transformer.
   Accuracy: 65%.
   Limitation: Generalization challenges.

4. **Deepfake Video Detection via Predictive Representation Learning (2022):**

   This paper employs Latent Pattern Sensing (LPS) with self-supervised predictive learning to identify spatiotemporal inconsistencies in deepfakes. It achieved an accuracy

of 86.9%, but the performance drops with highly realistic deepfakes. It also struggles with generalization,

Algorithm: LPS.

Accuracy: 86.9%.

Limitation: Less effective on realistic deepfakes.

5. **ID-Reveal: Identity-aware DeepFake Video Detection (2021):**

This method uses 3D morphable models (3DMM) and a Temporal ID Network to extract identity-based facial features for deepfake detection. It offers 75.6% accuracy but is limited by high computational cost.

Algorithm: Temporal ID Network and 3DMM.

Accuracy: 75.6%.

Limitation: High computational cost.

6. **Deepfake Video Detection Using Convolutional Vision Transformer (2021):**

Combining Convolutional Neural Network (CNN) and Vision Transformer (ViT), this paper uses local feature extraction and attention-based classification to detect deepfakes. It reaches 91.5% accuracy but lacks generalization across different deepfake generation techniques.

Algorithm: CNN and ViT.

Accuracy: 91.5%.

Limitation: Poor generalizability.

7. **A Survey on Deep Fake Video Detection (2021):**

This work uses SVM and Decision Tree to analyze text-based patterns for detecting deepfake content. It achieves 88.7% accuracy but suffers from high computational complexity.

Algorithm: SVM and Decision Tree.

Accuracy: 88.7%.

Limitation: High computational complexity.

8. **Deep Fake Video Detection Using Transfer Learning Approach (2022):**

This work uses SVM and Decision Tree to analyze text-based patterns for detecting deepfake content. It achieves 88.7% accuracy but suffers from high computational complexity.

Algorithm: CNN and RNN.

Accuracy: 91.03%.

Limitation: High computational complexity.

9. **Deep Fake Video Detection through Optical Flow-based CNN (2019):**

Utilizes optical flow and CNN to identify pixel-level manipulations in video frames. It yields an accuracy of 89.5% but is computationally intensive and less efficient.

Algorithm: CNN.

Accuracy: 89.5%.

Limitation: High computational demand, less efficiency.

10. **Deepfake Video Detection: Challenges and Opportunities (2024):**

Employs YOLO for object detection and EfficientNet for classification to detect deepfakes. Achieves good accuracy but struggles with fine features and slightly low performance.

Algorithm: YOLO and EfficientNet.

Accuracy: 84.83%.

Limitation: Misses fine-grained features.

11. **Deep-Fake Detection for Human Face Images and Videos (2022):**

This model focused on deep fake detection of biometric fingerprints. The biometric fingerprints are taken in image form. A GAN- based model was used for detecting deep fake biometric fingerprints images.

Algorithm: GAN.

Accuracy: 90%.

Limitation: Relatively small-scale datasets, unavailability of a standardized benchmark for testing, and the fast-evolving techniques of deep fake generation.

12. Deep Fake Detection Using Tensor Decomposition-Based Neural Network (2021):

This paper uses a tensor decomposition-based deep neural network that integrates news content, social context, and echo chambers is described, which merges the most diverging sources of data together but is limited with datasets and is not capable of real-time detection.

Algorithm: ResNet-50.

Accuracy: 99.96%.

Limitation: Dependency on computational resources.

13. Convolutional Vision Transformer (CViT) for Deepfake Detection (2021):

The framework CViT combined CNNs for local feature extraction with Vision Transformers for global and local feature correlations. This approach is excellent in dealing with spatial and temporal inconsistencies.

Algorithm: CViT.

Accuracy: 93.75%.

Limitation: It has weaknesses in artifacts of some types of deepfake and is sensitive to the quality of the face detection preprocessing.

14. Multi-Algorithm Framework for Deepfake Detection (2023):

This paper explores a mixture of GANs, CNNs, RNNs, and LSTMs in detecting deepfakes based on aspects of temporal as well as spatial inconsistency. It uses a variety of datasets and improved feature extraction methods.

Algorithm: GAN, CNN, RNN and LSTM.

Accuracy: 97.98%.

Limitation: The model is vulnerable to new forgery methods.

CHAPTER 3

PROPOSED METHODS

3.1 DEEP LEARNING

Deep learning describes a portion of machine learning which employs artificial neural networks like deep neural networks to undertake tasks that would be quite complicate to solve without such a system by automatically learning the features of the underlying data. It has become useful in areas like image recognition, natural language processing and lately in combatting deep fakes. It is worth noting that deep learning is particularly effective in the analysis of high-dimensional data and its subtle and complicated patterns which is applicable in the case of media manipulations. The hyperparameters (or parameters) used are as follows,

- **Learning Rate:** The step size for updating the weights of a model during optimization.
- **Patch Size:** The size of the image patches processed within ViT model.
- **Batch Size:** Number of training examples passed before updating model weights.
- **Epoch:** One complete pass through the entire training dataset.

3.2 DATASET DESCRIPTION

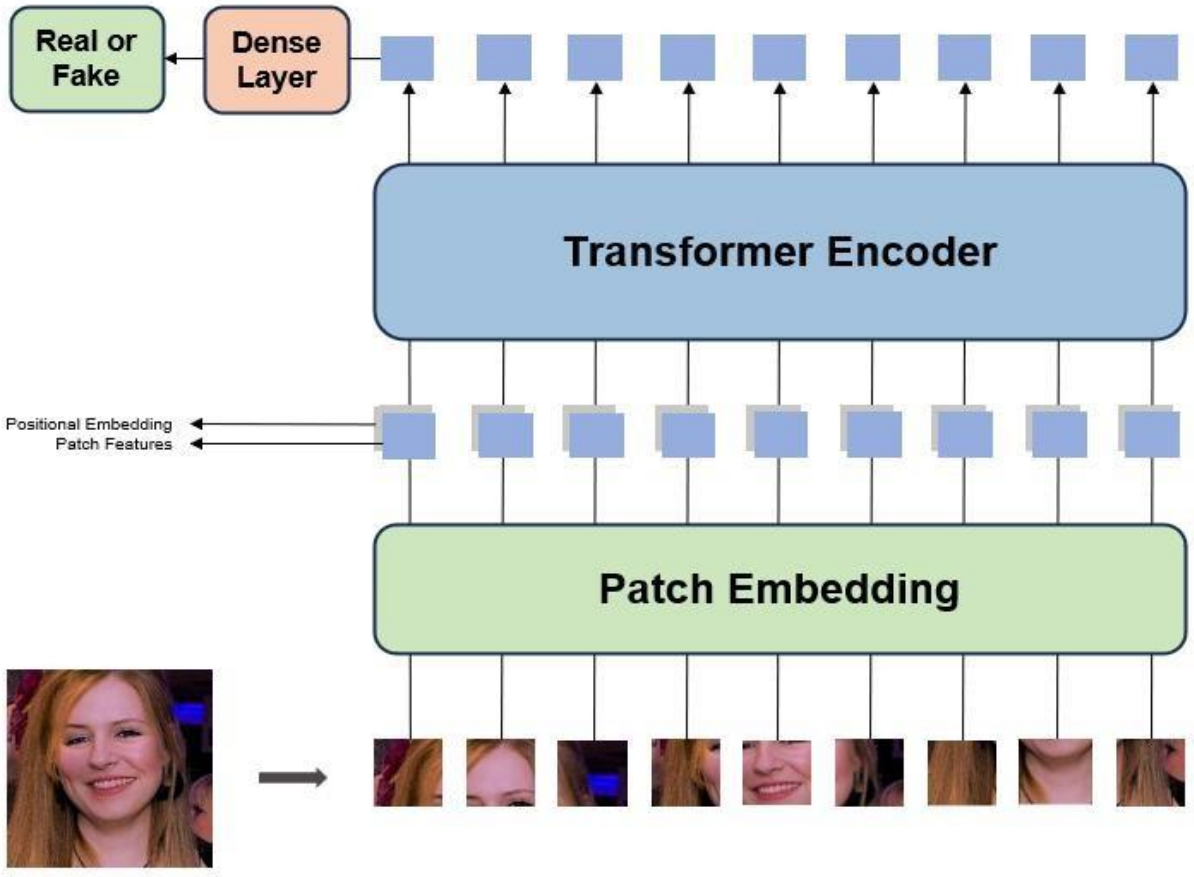
The work uses a publicly available dataset named “Deep Fake and Real Images” from Kaggle and “Spoofed and Bonafide Audio” from ASVspoof. Deep Fake and Real Images dataset contains images for training, validating and testing. The dataset for training has a total of 30,000 images, for validating has a total of 20,000 and for testing has a total of 10,000. The Spoofed and Bonafide Audio contains audios for training, validating and testing. The dataset for training has a total of 570 videos, for validating has a total of 162 videos and for testing has a total of 81 videos.

The Image and Video dataset consist of both fake and real datasets. The Audio datasets contain Logical Access and Physical Access. In logical access, audio samples are generated from systems where access is provided through logical means, like voice authentication

systems that might be vulnerable to manipulation using software-based spoofing techniques. The most common voices in these cases are simulated or artificially created voices mimicking real users. Physical Access refers to situations where audio samples produced from systems that possess physical means; in other words, the spoofed voices are generated based on the physically captured audio or attempts to influence real-world speech samples (for example, through hardware devices or recording from a real environment).

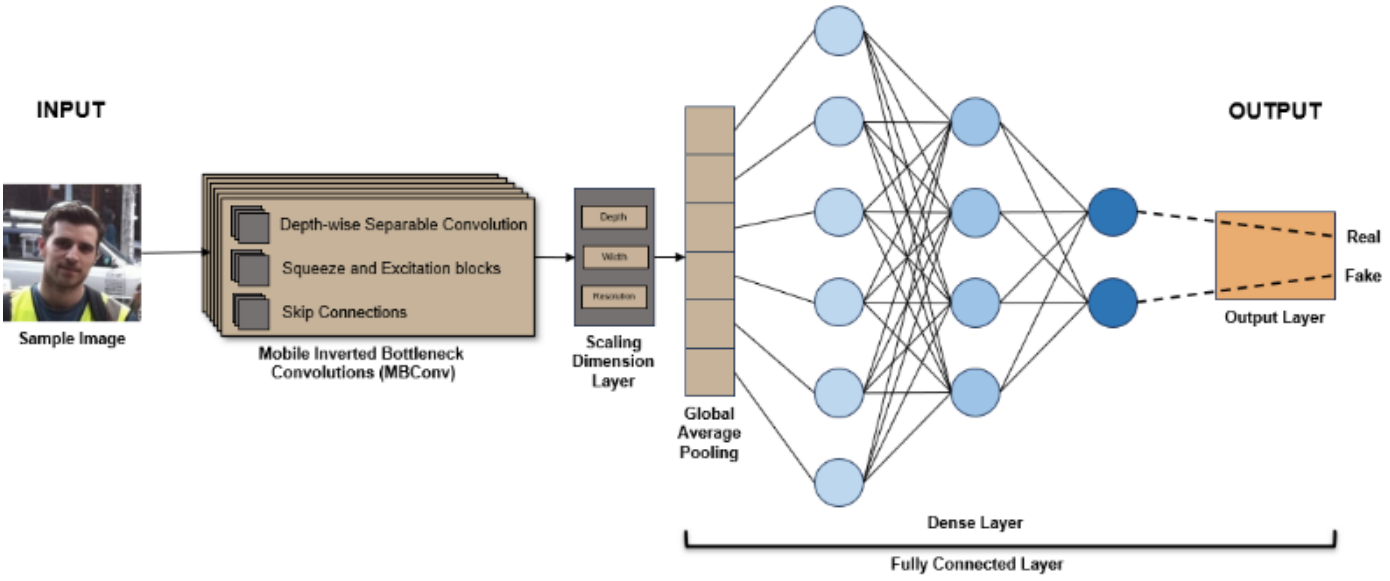
3.3 Deep Fake Detection Using ViT:

ViT is a relatively new field in deep learning that aims to process images as sequences of patches; this is like the token in natural processing. Unlike CNN, it uses self-attention mechanisms to capture overall patch level relationships within an image. This capability makes ViT considerably more powerful for detecting deep fakes because these kinds of irregularities often spread across spatial regions within an image. ViT achieves this by embedding patches, applying multi-head self-attention, and refining through feed-forward networks.



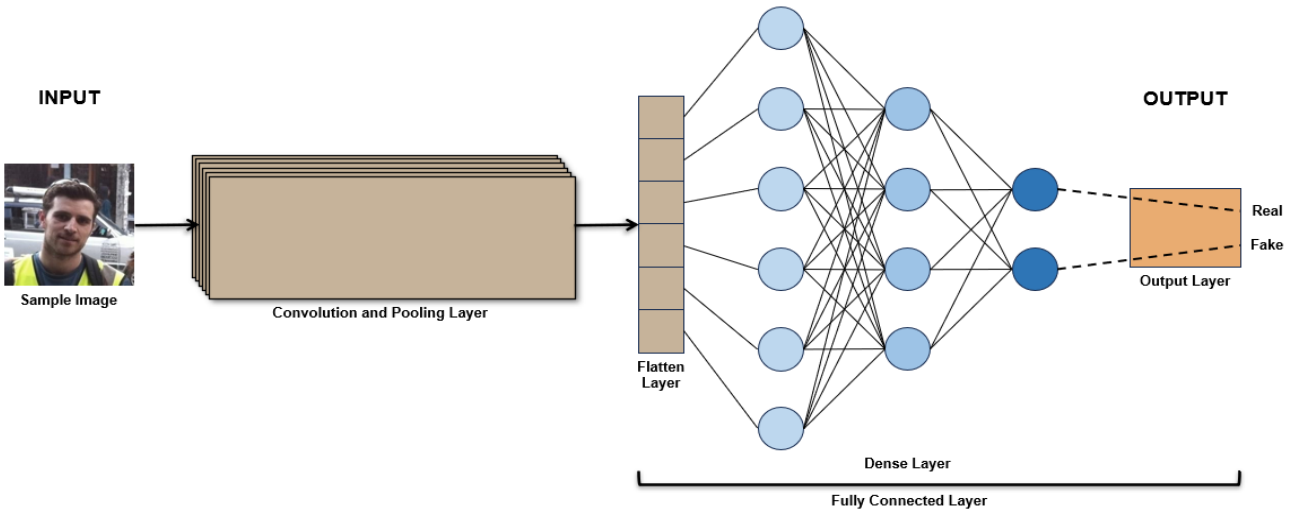
3.4 Deep Fake Detection Using EfficientNet:

EfficientNet is a cutting-edge convolutional neural network model that scales depth, width, and resolution equally by applying a compound scaling technique. Optimized scaling is beneficial for capturing delicate features of images and, at the same time, achieving high performance at fewer parameters. In the task of deep fake detection, EfficientNet is efficient in detecting delicate visual artifacts, including unnatural lighting, boundary misalignments, or inconsistencies in skin textures. Its effective feature extraction ability enables it to identify manipulated content with high accuracy, even in the case of subtle manipulations. The network design incorporates mobile inverted bottleneck convolutions and squeeze-and-excitation blocks to improve learning and accuracy in multiple layers.



3.5 Deep Fake Detection using CNN:

Images have adopted the use of Convolutional Neural Networks (CNN) for the purpose of deep fake detecting. For manipulative images, CNN divide images into rows and grids to capture very minute details of the images such as the texture patterns, edges and even the inconsistencies of the images. In cases where images have been faked, CNN for images are able to notice flaws and mistakes like odd pixel configurations and ineffective blending which arise from the faked image creation process. This mode of separation occurs in a various number of successive layers which include convolution, pooling and dense networks.



3.6 Detection of Deep Fake video:

In deep fake video detection, frames of the video are first isolated and handled as static images, which are processed through CNN models to determine if the content is fake. These models find spatial inconsistencies in face expressions, lighting, or blend errors between frames. Multiple frame predictions are combined to make a final decision for the video level to improve accuracy. Such sequence-by-sequence comparison and subsequent temporal merging assist in the detection of these fine manipulations happening over sequences, like lip-sync inconsistency or artificial facial movement imposed by deep fake generation systems.

3.7 DEEP FAKE IMAGE DETECTION USING ViT

3.7.1 Architecture Overview:

Fake-image detection uses the Vision Transformer (ViT) model. It splits the images into small patches. These patches are processed using self-attention layers and classify the images by aggregating the output. The transformer architecture is used to model global relationships between images.

The workflow of Vision Transformer model is as follows,

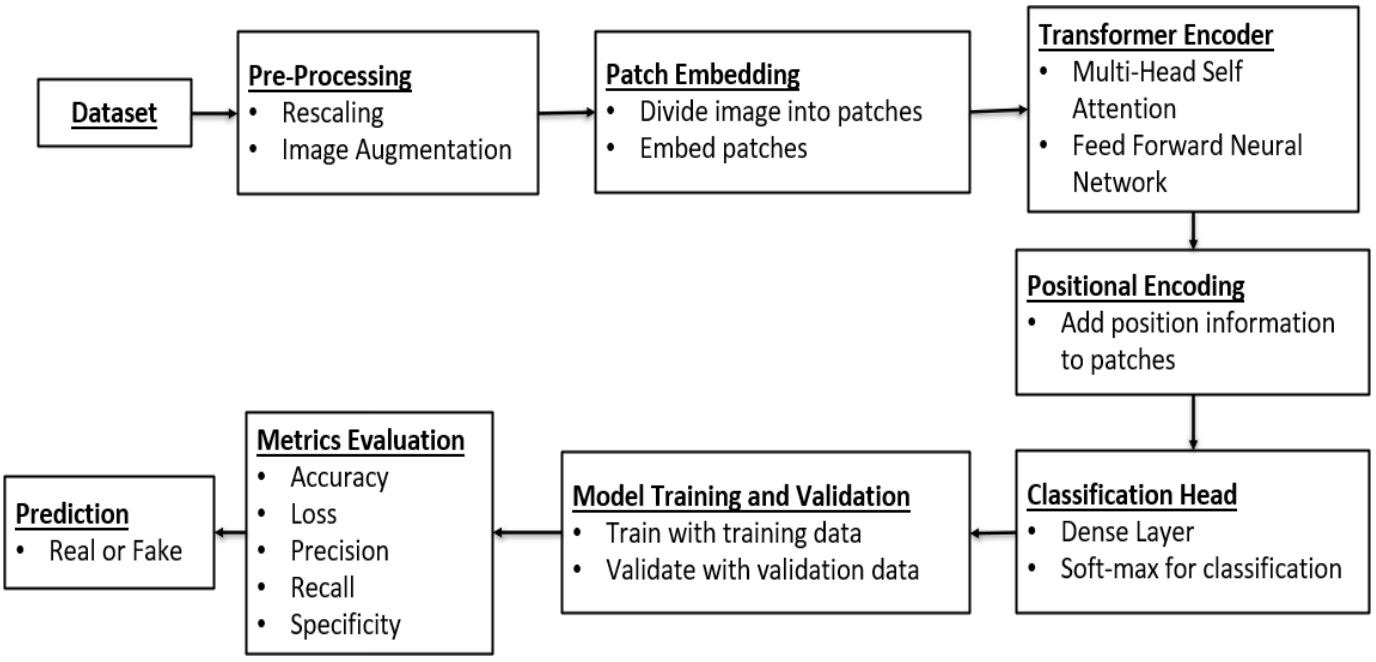


Figure 2 – Workflow of ViT approach for image detection

The collected image datasets are initially preprocessed by rescaling and image augmenting. The preprocessed image datasets are fed to a ViT model which contains a transformer encoder. The images from the datasets are divided into patches prior to the encoder. The outputs are optimized, and the parameters are tuned (or adjusted). Then the model is trained and validated using training and validating datasets. Finally, the model is tested using testing datasets and various metrics are evaluated. The metrics to be evaluated are Accuracy, Loss, Precision, Sensitivity (Recall), Specificity F1-Score, Matthews Correlation Coefficient (MCC), Balanced Classification Rate (BCR) and Cohen-Kappa.

3.7.1.1 ViT Layers:

The Vision Transformer model consists of the following 16 layers,

Patch Embedding Layer (Layer 1):

- It divides the image into patches for embedding purposes.
- It maps each patch to a high-dimensional embedding vector.

Positional Encoding Layer (Layer 2):

- It adds positional information to patches, which helps to preserve the spatial relations.
- Help the model to order the patches in sequence as it occurs in an image.

Multi-Head Self-Attention and Feed Forward Network Sub-layer (Layers 3–10):

- It learns the relationship among patches with attention mechanism.
- It helps to capture different types of similarities and fine-tune the feature embeddings and for non-linear transformation.

Layer Normalization (Layer 11):

- Stabilizes inputs for smoother training and better convergence.
- Scale attention and feed-forward outputs for normalization purposes.

Dropout Layers (Layer 12):

- It prevents overfitting by randomly deleting units during training.
- Adds randomness which improves model generalization.

Residual Connections (Layer 13):

- It combines the original input and output for memorization.
- It helps to learn deeper features and retain input information.

Dense Layer-MLP Head (Layer 14):

- Maps final embeddings to a single representation vector.
- Prepare embedding for classification or regression tasks.

Output Dense Layer (Layer 15):

- Produces outputs using sigmoid activation and converts learned representations into interpretable outputs.

Final Layer Normalization (Layer 16):

- Normalizes and stabilizes the final output for consistent scaling and prediction.

3.7.2 Hyperparameter Tuning and Training:

The Vision Transformer model for image detection used highly tuned and trained hyperparameters to achieve better performance. Some of the parameters in consideration are as follows,

3.7.2.1 Hyperparameter Optimization:

- **Learning Rate:** $1e^{-4}$ – initial value and tuned during the experiment to achieve stable convergence.
- **Batch Size:** 32 – best tuned trade-off between computation and gradient stability.
- **Patch Size:** 16x16 – gives a reasonable feature which represents the image.
- **Number of Heads:** 8 multi-head self-attention used for robust feature extraction.
- **Dropout Rate:** The model used regularization to prevent overfitting.

3.7.2.2 Training Parameters:

- **Epochs:** 20 – to achieve robust performance and prevent overfitting.
- **Metrics:** Accuracy, Loss, Precision, Sensitivity Specificity, F1-Score, Matthews Correlation Coefficient (MCC), Balanced Classification Rate (BCR) and Cohen-Kappa.

3.8 DEEP FAKE IMAGE DETECTION USING EFFICIENTNET

3.8.1 Architecture Overview:

EfficientNet is a compact and effective deep learning model which has high performance with low model size and computational expense. For the detection of deep fake images in our implementation, EfficientNetB0 (compact yet effective variant of EfficientNet) is used as the feature extractor. EfficientNet differs from typical CNNs because it scales depth, width, and resolution in compound scaling mode. This results in enhanced accuracy and effectiveness in identifying patterns of deep fake images.

The workflow of EfficientNet model is as follows,

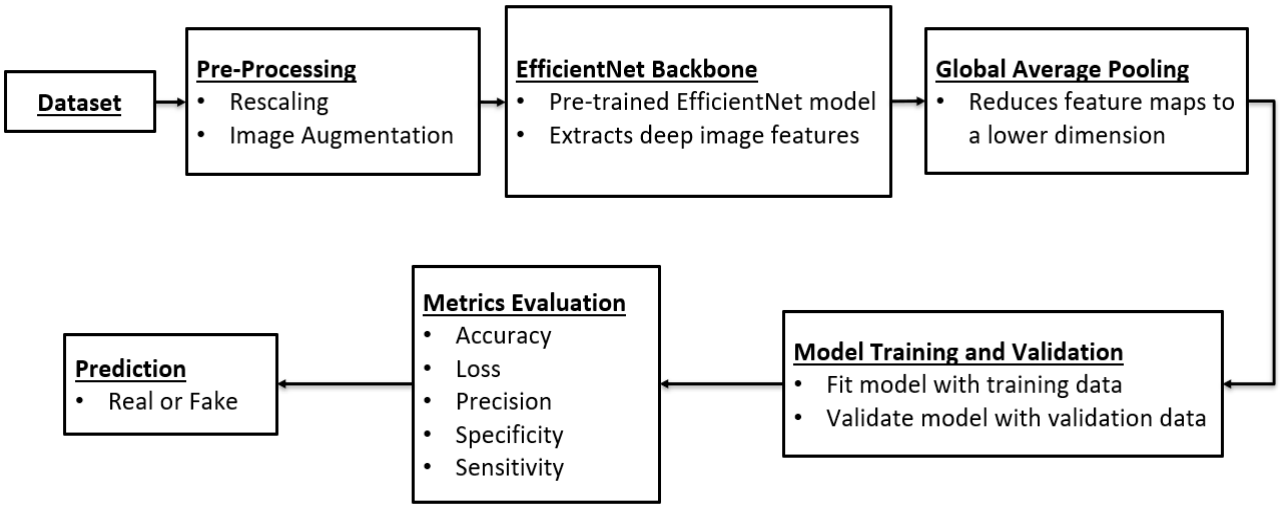


Figure 2 – Workflow of EfficientNet approach for image detection

The input data is preprocessed by resizing the images to 224x224 pixels (EfficientNetB0 requirement) and rotating and flipping the images for improved generalization. The preprocessed images are fed into the EfficientNetB0 backbone (ImageNet-trained), and high-level and discriminative features are obtained. The custom fully connected layers are provided with these features. The final model is trained, validated, and tested on respective datasets. The computed metrics are Accuracy, Loss, Precision, Sensitivity (Recall), Specificity, F1-Score, Matthews Correlation Coefficient (MCC), Balanced Classification Rate (BCR) and Cohen-Kappa.

3.8.1.1 Effective Feature Extraction using EfficientNet:

EfficientNet introduces the compound scaling approach to scale the depth (number of layers), width (number of channels), and resolution (size of the input image) in a similar proportion. In our implementation, EfficientNetB0 is employed as the basic network:

- **Base Model:** EfficientNetB0 (with ImageNet pre-trained weights)
- **Input Size:** 224x224 pixels (default input size for B0)
- **Include Top:** False (not to include ImageNet classifier head)
- **Trainable Layers:** Fine-tuning allowed upper layers to be trained to detect deep fake features.

EfficientNetB0 accurately detects spatial and semantic features with few redundant computations. Good generalization is further boosted with a pretrained model and fine-tuning on the deep fake data.

Mobile Inverted Bottleneck Blocks (MBConv):

- Multiple MBConv blocks are stacked together.
- Each MBConv block consists of Depthwise Separable Convolution which reduces computation, Squeeze-and-Excitation (SE) block.
- These blocks enhance important features and Expansion & Pointwise Convolution block which captures complex patterns.

Fully Connected Layers:

After EfficientNet extracts the feature maps, the output is flattened and passed through custom dense layers for classification:

- **Global Average Pooling Layer:**
  - **a)** Reduces the feature maps into a single vector per feature map
  - **b)** Layer Type: GlobalAveragePooling2D

- **Dense Layer:**
  - **a)** 256 fully connected neurons with ReLU activation

b) Layer Type: Dense

- **Dropout Layer:**
  a) Dropout rate: 0.5 (to prevent overfitting)
  b) Layer Type: Dropout

Output Layer:

- Single neuron with Sigmoid activation function for binary classification.
- Layer Type: Dense

3.8.2 Training Process and Parameters:

3.8.2.1 Training Process

- **Forward Propagation:** Images pass through EfficientNet and custom layers to generate predictions.
- **Loss Calculation:** Binary cross entropy is used for measuring prediction error.
- **Evaluation:** The model is evaluated using the validation set with hyperparameter tuning.

3.8.2.2 Training Parameters:

- **Learning Rate:** 0.0001 – Lower learning rate ensures stable convergence during fine-tuning.
- **Batch Size:** 32 – Balanced for speed and gradient smoothness.
- **Epochs:** 20 – Chosen to prevent overfitting and ensure model robustness.
- **Metrics:** Accuracy, Loss, Precision, Sensitivity, Specificity, F1-Score, Matthews Correlation Coefficient (MCC), Balanced Classification Rate (BCR) and Cohen-Kappa.

3.9 DEEP FAKE IMAGE DETECTION USING CNN

3.9.1 Architecture Overview:

Our first implementation method for deep fake image detection using the Convolutional Neural Network (CNN) model consists of a total of 11 layers. It processes the entire image by passing it through multiple convolutional layers, which applies filters to capture spatial and hierarchical features. It uses operations of convolution, pooling and fully connected layers to extract the features and to classify the image.

The workflow and the layers of CNN used are as follows,

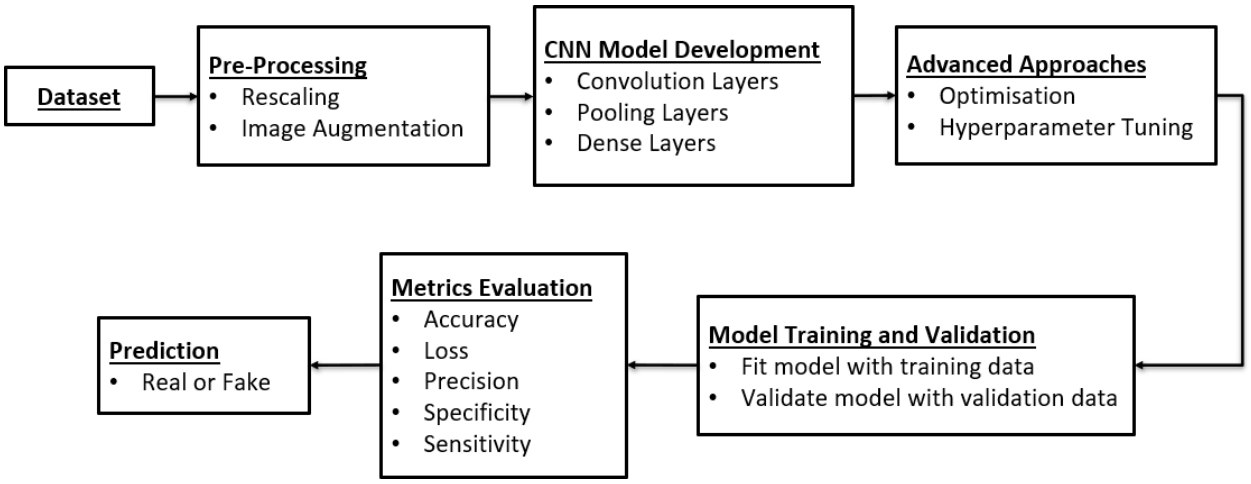


Figure 1 – Workflow of CNN approach for image detection

The collected image datasets are initially preprocessed which includes rescaling (the process of standardizing the image pixels, here the image is standardized to 224x224 pixels) and image augmenting (the process of adjusting the image to right proportion, which includes image rotation and flipping). The image datasets are then passed to a CNN model. The outputs are optimized, and the parameters are tuned (or adjusted). Then the model is trained and validated using training and validating datasets. Finally, the model is tested using testing datasets and various metrics are evaluated. The metrics to be evaluated are Accuracy, Loss, Precision, Sensitivity (Recall), Specificity, F1-Score, Matthews Correlation Coefficient (MCC), Balanced Classification Rate (BCR) and Cohen-Kappa.

3.9.1.1 Convolutional and Pooling layer:

In a CNN, convolutional layers pull out features of pictures by applying learnable filters on the input images, capturing spatial hierarchies like edges and textures. Pooling down the spatial dimensions is done to increase the efficiency of computational pipeline and reduces overfitting. Alternating between convolutional and pooling layers would allow the model to learn hierarchical features and complex patterns in deeper layers. Pooling also ensures translation invariance and reduces feature map size. Alternately using four convolutional and four pooling layers optimizes the extraction of features and maintains efficiency.

First Convolutional Layer:

- Applies 32 filters of size (3x3) with ReLU activation.
- **Layer Type:** Conv2D

First Max Pooling Layer:

- Reduces the spatial dimensions using (2x2) pooling.
- **Layer Type:** MaxPooling2D

Second Convolutional Layer:

- Applies 64 filters of size (3x3) with ReLU activation.
- **Layer Type:** Conv2D

Second Max Pooling Layer:

- Reduces the spatial dimensions using (2x2) pooling.
- **Layer Type:** MaxPooling2D

Third Convolutional Layer:

- Applies 128 filters of size (3x3) with ReLU activation.
- **Layer Type:** Conv2D

Third Max Pooling Layer:

- Reduces the spatial dimensions using (2x2) pooling.

- **Layer Type:** MaxPooling2D

Fourth Convolutional Layer:

- Applies 128 filters of size (3x3) with ReLU activation.
- **Layer Type:** Conv2D

Fourth Max Pooling Layer:

- Reduces the spatial dimensions using (2x2) pooling.
- **Layer Type:** MaxPooling2D

The model has 32 filters in its first convolutional layer, capturing the basic features like edges. The number of filters increases gradually to 64 and then to 128, which enables it to learn more complex patterns. The standard filter size is (3x3), commonly chosen for a fair balance between these two conflicting requirements: computational efficiency and capturing spatial local detail. The ReLU activation function has non-linearity which enables the model to learn complex patterns, without vanishing gradients. The range of ReLU function is zero to infinity. Max pooling is used in the convolutional layer to reduce the spatial dimension by 2x2, permits important features with reduced computational and to prevent overfitting.

3.9.1.2 Fully Connected Layers:

The fully connected layers in a neural network serve as the final stage to process features extracted by convolutional and pooling layers and make predictions.

Flattening Layer:

- Converts the multi-dimensional (here 2D) feature maps into 1D vectors and feeds it into fully connected layers.
- **Layer Type:** Flatten

Dense Layer:

- 512 fully connected neurons with ReLU activation function.
- **Layer Type:** Dense

Output Layer:

- Final output with 1 neuron and Sigmoid activation function.
- **Layer Type:** Dense

3.9.2 Training Process and Parameters:

Generally, in training, data is used for computations of errors based upon a pre-defined loss function. Optimally optimizing the network weights within the connected layers of the network results in classifying quite effectively in the model.

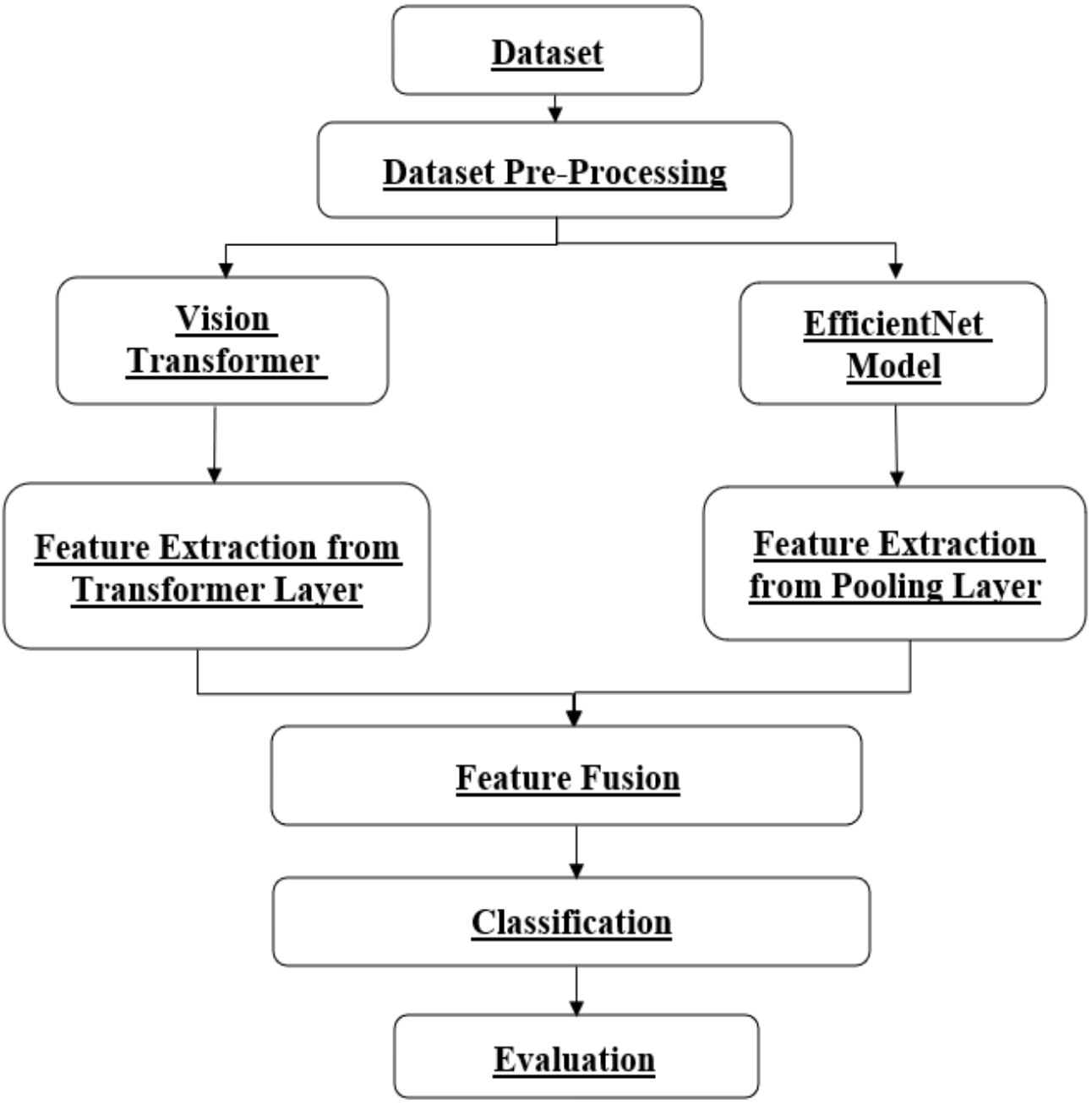
3.9.2.1 Training Process:

- **Forward Propagation:** It allows images to flow through convolutional, pooling, and fully connected layers and generates a prediction.
- **Loss Calculation:** The binary cross entropy loss function measures the difference between predicted and actual labels.
- **Evaluation:** Validate the model on the validation set using hyperparameter fine-tuning.

3.9.2.2 Training Parameters:

- **Learning Rate:** 0.001 **–** to achieve stability and simplified training process.
- **Batch Size:** 32 – best tuned trade-off between computation and gradient stability.
- **Epoch:** 20 **-** to achieve robust performance and prevent overfitting.
- **Metrics:** Accuracy, Loss, Precision, Sensitivity, Specificity, F1-Score, Matthews Correlation Coefficient (MCC), Balanced Classification Rate (BCR) and Cohen-Kappa**.**

3.10 Proposed Workflow which implements both ViT and EfficientNet:



3.10.1 Architecture Overview

A. Vision Transformer (ViT) Model:

- **Embedding Layer:** Each patch is converted into a higher-dimensional space using a linear layer. Positional information is encoded to preserve the spatial details.
- **Transformer Layer:** Consists of multiple blocks such as Multi-Head Self-Attention (MHSA) which helps the model focus on relevant regions of the image. Feedforward

Network (FFN) which applies non-linearity to each input independently.

B. EfficientNet Model:

- **Mobile Inverted Bottleneck Blocks (MBConv):** Multiple MBConv blocks are stacked together. Each MBConv block consists of Depthwise Separable Convolution which reduces computation, Squeeze-and-Excitation (SE) block which enhances important features and Expansion & Pointwise Convolution block which captures complex patterns.
- **Global Average Pooling (GAP):** This layer converts spatial feature maps into a single feature vector. It also reduces overfitting by removing unnecessary spatial details. The output is a 1280-dimensional feature vector.
- **Fully Connected Layer:** This layer maps the extracted features to the output classes. It uses a Sigmoid activation function.

C. Classification Stage:

The various Classifiers used are as follows,

- **Linear SVM:** SVM or Linear SVM separates features linearly.
- **Polynomial SVM:** Uses a polynomial kernel for non-linear separation.
- **RBF SVM:** Uses an RBF kernel for non-linear decision boundaries. It is used to handle non-linearly separable data.
- **Sigmoid SVM:** Uses a sigmoid kernel function to convert the input data into a higher dimensional space.
- **XGBoost:** A Decision Tree-based classifier. Gradient-boosted decision trees for classification.
- **Random Forest:** A Decision Tree-based classifier. Builds multiple decision trees during training and combines their outputs.

D. Specification:

- **Feature Extraction Models:** Vision Transformer, EfficientNet.
- **Feature Extraction Input Shape:** (224, 224, 3).

- **Optimizer:** Adam's Optimizer.
- **Classification Models:** Linear SVM, Polynomial SVM, Radial Base Function SVM, Sigmoid SVM, XGBoost, Random Forest.
- **Activation Function (Extractors):** Pre-trained (specific layers used).
- **Pooling:** Global Average Pooling (EfficientNet).
- **Optimizers for Feature Extraction:** Pre-trained weights (no optimizer for inference).
- **Evaluation Metrics:** Accuracy, Loss, Precision, Sensitivity, Specificity, F1-Score, Matthews Correlation Coefficient (MCC), Balanced Classification Rate (BCR) and Cohen Kappa.

3.11 SYSTEM CONFIGURATIONS

3.11.1 Software Requirements:

Windows 10 or above, Python 3.10 and other installed libraries and packages such as NumPy, Matplotlib, Librosa, TensorFlow, Keras, Scikit-Learn.

3.11.1.1 Python:

Python is a high-level programming language that is known for its simplicity and readability. It supports several paradigms in programming, including the paradigms of procedural, object-oriented, and functional programming, thus suitable for most applications. The VS Code interpreter along with Python 3.10 is used for the project. The motivation for choosing Python 3.10 version was that it provided the perfect balancing point between modern features and stability such that it was an excellent choice to use while getting into deep fake detection and was also compatible with audio processing libraries. Our project required the creation of a virtual environment. It ensured isolation and controlled workspace for our project. It served the purpose of avoiding conflicts between library versions.

3.5.1.1 PIP:

PIP is the most convenient, standard package management tool for Python. PIP makes it

effective to install, update and manage the dependencies of a Python library. PIP was one of the main tools for our project, which let us install all required libraries and frameworks. Without PIP, it would have been very difficult to install and manage the required libraries. PIP's reliability and versatility made it an indispensable tool for our project.

3.11.1.2 Libraries and Packages:

The following are the libraries and packages used for our project.

TensorFlow:

TensorFlow is one of the most popular open-source frameworks for running various machine learning and deep learning applications. This project used TensorFlow to train both CNN and ViT models, which provide tools for model design, optimization, and evaluation.

Librosa:

Librosa is a Python library for audio and music analysis, it is crucial to our project's audio preprocessing. It is used for the conversion of the audio signals into spectrograms, which helps us with deep fake audio detection.

Keras:

Deeply integrated with TensorFlow, Keras is a higher-level API for neural networks. This is used to design CNN and ViT architectures, define a custom metric such as sensitivity and specificity, and make the model-evaluation process go smoother.

Scikit-Learn:

This is the all-inclusive library for machine learning and supported the entire project; it provided tools for evaluation metrics, including confusion matrices, classification reports, and accuracy calculations and loss calculation.

NumPy:

NumPy is the cornerstone of numerical computing for Python. Its ability to handle arrays

and matrices efficiently made it an important feature for data preprocessing and manipulation. In our project, NumPy smoothed processes such as normalization over spectrograms.

Matplotlib:

Matplotlib is a high-quality visualization library used mainly for plotting accuracy, loss, sensitivity, and specificity graphs for conditions during training and validation. Such visualizations give insight into model performances and allow tuning of parameters.

3.11.2 HARDWARE REQUIREMENTS

Processor: Intel core i5 or above.

Core: 64-bit, quad-core, 2.5 GHz minimum per core

RAM: 16 GB or more

Hard Disk: 30 GB of available space or more.

Display: Dual XGA (1024 x 768) or higher resolution monitors

Operating System: Windows 10 or above.

CHAPTER 4

RESULTS & DISCUSSIONS

4.1 EVALUATION METRICS AND PARAMETERS

The Loss and Accuracy were calculated directly from predictions and using the pre-defined loss functions, binary cross entropy, and the accuracy metric provided by TensorFlow/Keras. Precision, Sensitivity, Specificity and F1-Score were defined as custom metrics using the mathematical formulas of True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN).

The values were calculated using TensorFlow operations, such as (tf.round) and (tf.keras.backend.sum). Here, True Positive indicates how many times the algorithm has correctly identified the real image as real while, the True Negative indicates how many times the algorithm has identified the fake image as fake. False Positive denotes the number of times the algorithm has miscalculated the real image for fake while False Negative denotes the number of times the algorithm miscalculated fake image for real.

Performance Metrics	Equation
Accuracy	
Loss	
Precision	$Pre = \frac{TP}{TP + FP}$
Sensitivity (Recall)	$Sen = \frac{TP}{TP + FN}$
Specificity	$Spc = \frac{TN}{TN + FP}$

BCR	$BCR = \frac{Sen + Spc}{2}$
MCC	$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$
Cohen's Kappa	
F1-Score	$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall}$

Table 1 – Performance metric equations

Precision is in fact the ratio of the number of truly positive observations predicted to the total number of positive observations predicted. Specificity is also known as True Negative Rate that computes the model's ability to identify and correct for the negative classes. Sensitivity is also called Recall or True Positive Rate that calculates the model's ability to identify and correct for positive classes. The harmonic means of Precision and Recall is in fact what defines the F1-score. This measure is balanced regarding both false positives and false negatives, especially when the dataset is imbalanced. Instead of taking an unweighted mean of the metric for each class, macro-average calculates a metric for each class individually and then takes the unweighted mean. The weighted average takes the metric for each class and weighs it by the number of samples in that class.

The confusion matrix provided detailed insights into the classification performance of the deep learning models and contains parameters such as True Positives (TP), True Negatives (TN), False Positives (FP), False Negatives (FN). From the confusion matrix, key metrics such as precision, sensitivity, specificity and f1-score were derived. These metrics highlighted the model's strength in distinguishing deep fake images or audios from authentic ones.

$$Confusion\ Matrix = \begin{pmatrix} TN & FP \\ FN & TP \end{pmatrix}$$

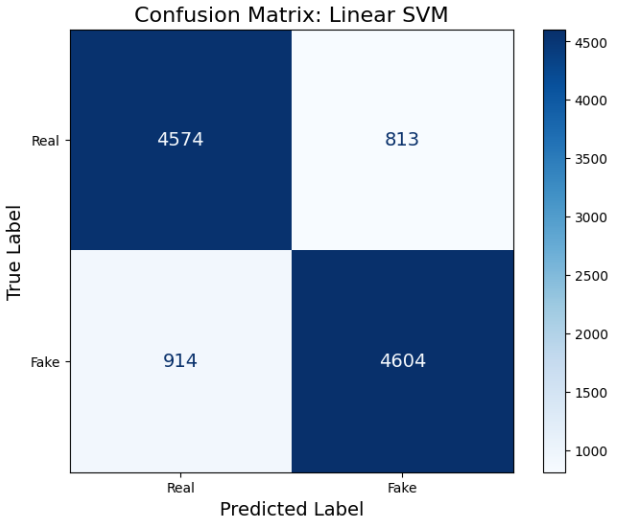
4.2 VIT AND EFFICIENTNET RESULTS FOR IMAGE DETECTION

The Vision Transformer and EfficientNet model is a particular computer vision model that uses transformer and EfficientNet model architectures in the detection of deep fake images. While running on the dataset, the ViT model and EfficientNet model efficiently displayed the capability in both feature extraction and classification accuracy. The results obtained by the ViT model and EfficientNet model are shown. Having calculated key metrics like accuracy, precision, recall, and F1-score, it has proven that the model is great at determining real and fake images.

4.2.1 Result and Report Classification

1) Linear SVM Results:

The following figure shows the confusion matrix of Linear SVM classifier.



Confusion matrix: Linear SVM

The following table displays the performance metrics of Linear SVM classifier.

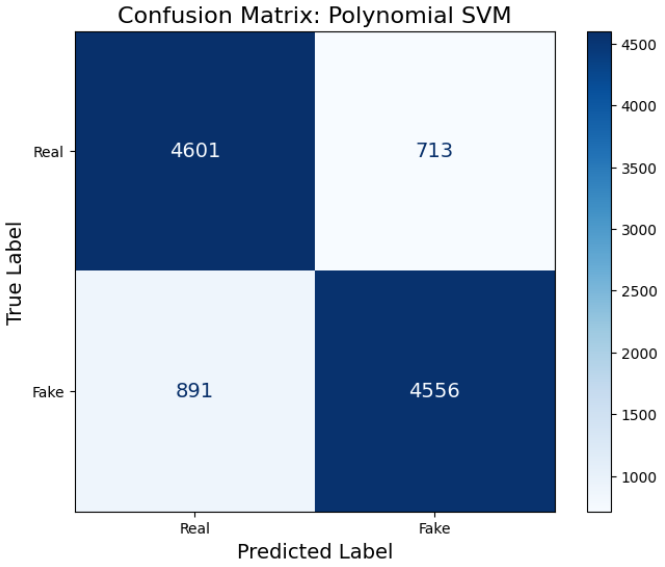
Metrics	Value
Accuracy	84.16%
Loss	15.84%
Precision	0.8499

Sensitivity	0.8344
Specificity	0.8491
MCC	0.6834
BCR	0.8417
Cohen's Kappa	0.6833
F1-Score	0.8421

Performance Metrics: Linear SVM

2) Polynomial SVM Results:

The following figure shows the confusion matrix of Polynomial SVM classifier.



Confusion matrix: Polynomial SVM

The following table displays the performance metrics of Polynomial SVM classifier.

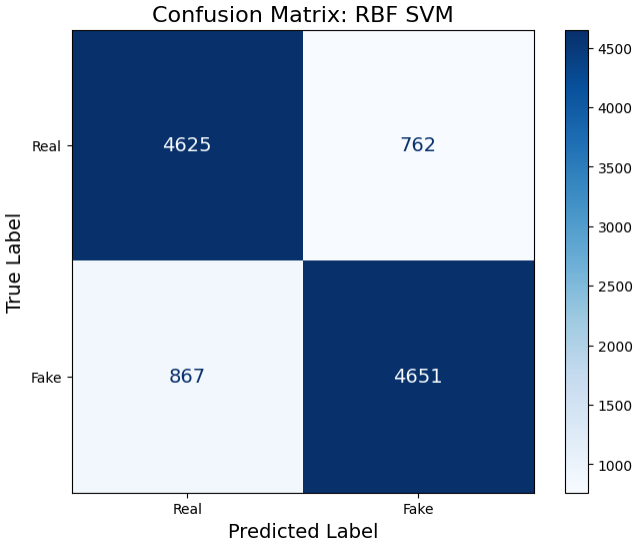
Metrics	**Value**
Accuracy	83.97%
Loss	16.03%
Precision	0.8417
Sensitivity	0.8364

Specificity	0.8430
MCC	0.6794
BCR	0.8397
Cohen's Kappa	0.6794
F1-Score	0.8390

Performance Metrics: Polynomial SVM

3) Radial Base Function (RBF) SVM Results:

The following figure shows the confusion matrix of RBF SVM classifier.



Confusion matrix: RBF SVM

The following table displays the performance metrics of RBF SVM classifier.

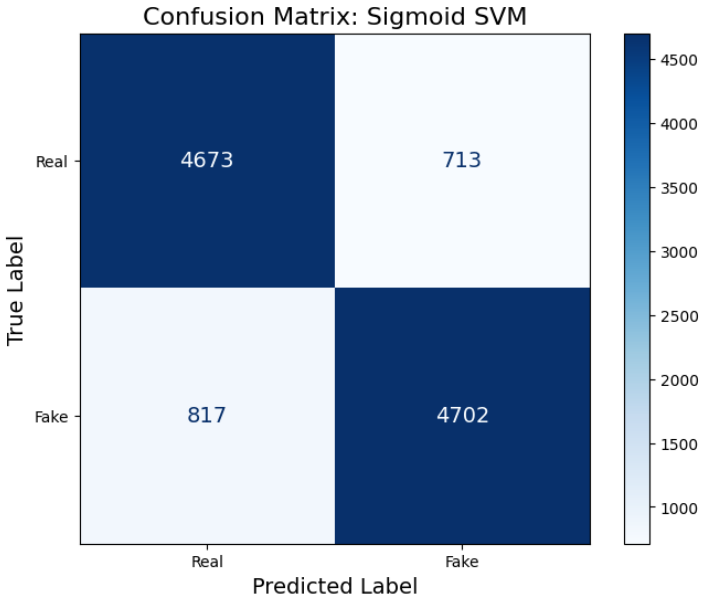
Metrics	Value
Accuracy	85.06%
Loss	14.94%
Precision	0.8592
Sensitivity	0.8429
Specificity	0.8585

MCC	0.7014
BCR	0.8507
Cohen's Kappa	0.7013
F1-Score	0.8510

Performance Metrics: RBF SVM

4) Sigmoid SVM Results:

The following figure shows the confusion matrix of Sigmoid SVM classifier.



Confusion matrix: Sigmoid SVM

The following table displays the performance metrics of Sigmoid SVM classifier.

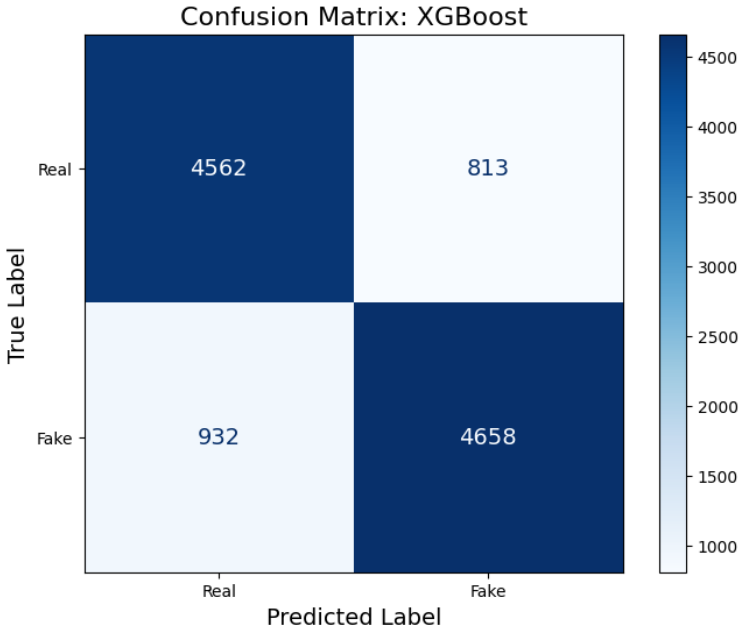
Metrics	**Value**
Accuracy	**85.97%**
Loss	14.03%
Precision	0.8683
Sensitivity	0.8520
Specificity	0.8676
MCC	0.7195

BCR	0.8598
Cohen's Kappa	0.7194
F1-Score	0.8601

Performance Metrics: Sigmoid SVM

5) XGBoost Results:

The following figure shows the confusion matrix of XGBoost classifier.



Confusion matrix: XGBoost

The following table displays the performance metrics of XGBoost classifier.

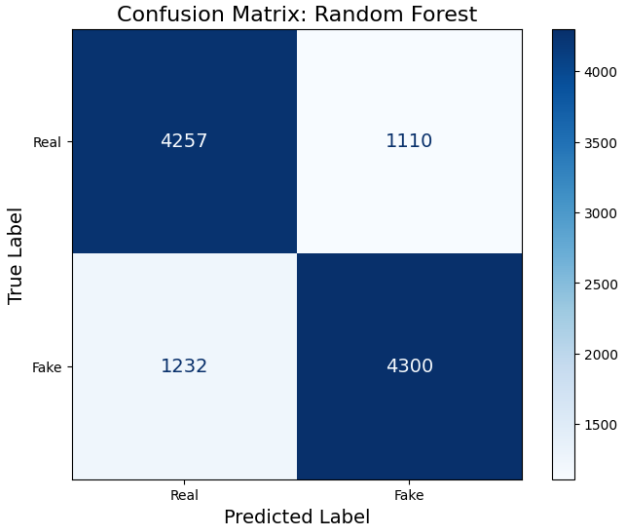
Metrics	Value
Accuracy	84.09%
Loss	15.91%
Precision	0.8514
Sensitivity	0.8333
Specificity	0.8487
MCC	0.6819

BCR	0.8410
Cohen's Kappa	0.6817
F1-Score	0.8422

Performance Metrics: XGBoost

6) Random Forest Results:

The following figure shows the confusion matrix of Random Forest classifier.



Confusion matrix: Random Forest

The following table displays the performance metrics of Random Forest classifier.

Metrics	**Value**
Accuracy	78.51%
Loss	21.49%
Precision	0.7948
Sensitivity	0.7773
Specificity	0.7932
MCC	0.5704

BCR	0.7852
Cohen's Kappa	0.5703
F1-Score	0.7860

Performance Metrics: Random Forest

4.1 CNN RESULTS FOR VIDEO DETECTION

This section presents the results for the CNN model on video-based deep fake detection. CNN, with its high potential in the extraction of features, was used to analyze the spatial patterns contained in images and to classify them as real or fake. Metrics such as Accuracy, Loss, Precision, Sensitivity, Specificity, MCC, BCR, Cohen Kappa and F1-Score were used to measure how well the model performs. Additionally, a confusion matrix was created to dig deeper into the classification performance of the model, including the number of correctly and incorrectly classified instances into real and fake videos.

4.1.1 Results and Classification Report:

The Test Accuracy and the Test Loss result are as follows,

Accuracy	98.74%
Loss	1.26%

Accuracy and Loss of CNN model for video detection

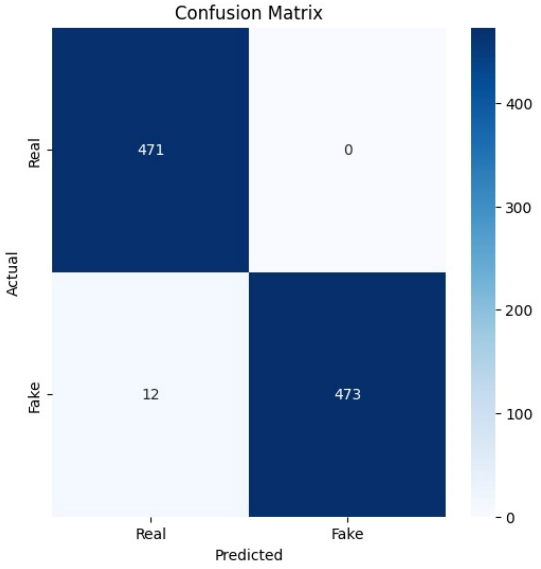
We obtained an accuracy of about 98.74% and a loss of 1.26%, which shows the robustness of our model. The report for Precision, MCC, BCR, Cohen Kappa and F1-Score are as follows,

Sensitivity	1.00
Specificity	0.8491

Precision	1.00
MCC	0.9749
BCR	0.9876
Cohen Kappa	
F1 – Score	0.9874

Precision, Sensitivity, Specificity, MCC, BCR and F1-Score of CNN model for audio detection

The confusion matrix is as follows,



Confusion Matrix of CNN model for video detection

From the above confusion matrix, true positive, false positive, false negative and true negative can be inferred as follows,

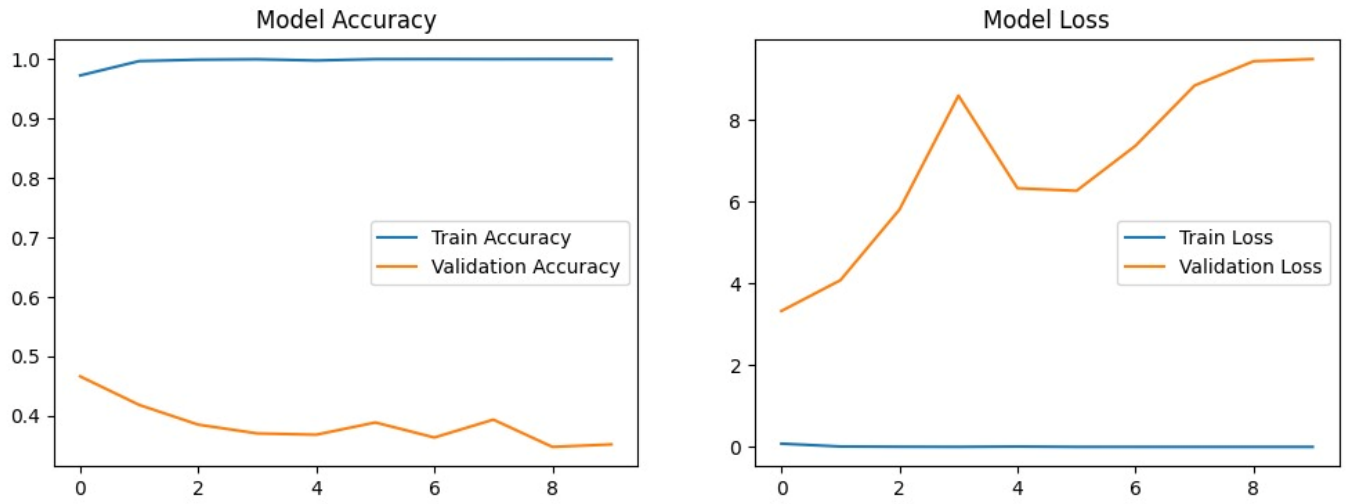
$$True\ Positive(TP) = 473$$

$$False\ Positive(FP) = 0$$

$$False\ Negative(FN) = 12$$

$$True\ Negative(TN) = 471$$

4.1.2 Results Graphs:



Accuracy and Loss graph of CNN model for videos

The graph displays the performance metrics Accuracy and Loss of the CNN model during Training and Validation stages. It shows the metrics against epochs. The training accuracy is always high throughout all the epochs, whereas the validation accuracy fluctuates significantly and decreases over time. The training loss is low and stable, but the validation loss is changing and continues to increase. Both training and validation accuracy have stable values across.

4.1 RESULTS COMPARISON

Our model achieves an accuracy of 97.96% and 97.76% in deep fake image detection using CNN and ViT respectively, which is higher than the accuracy of most of the model, referred from the literature paper. When comparing the CNN and ViT model for image detection, both models showed very similar results, that is similar accuracy percentage on same datasets. In the case of deep fake audio detection, our model acquired an accuracy of 87.65%, which is slightly lower than the models referred from the literature paper.

CHAPTER 5

5.1 SUMMARY

The project works on deep fake images and videos detection techniques. The initial implementation of the project used Convolutional Neural Networks (CNNs) for detection in images, which had satisfactory results. In the second stage, building on this foundation, a Vision Transformer (ViT) model and EfficientNet model was implemented, using its better attention mechanisms to improve the accuracy of the model. Then, the project scaled up to detect audio deep fakes, where audio files were converted into images through spectrograms.

Data sets such as Kaggle and ASVspoof have been pre-processed to separate real and fake data for training, validating, and testing models. Improving the accuracy of the models, overfitting, and generalization across the datasets have been the core challenges. Ultimately, the objective is high reliability in deep fake identification, which will lead to countering misuse through misinformation and digital forgery.

Future Scope: The project can be extended to video content by using spatial-temporal models like 3D CNNs or Transformers. Moreover, further hyperparameter tuning can enhance the audio analysis as well. Additionally, developing a system in real time and applying such methodologies on social media also provides great practical applications.

5.2 CONCLUSION

This project has kind of been able to tackle the problem of detection against deep fakes by using advanced deep learning models for images and audio detection. It used techniques such as CNN and ViT for classification in images. Also, it provided the identification of an audio deep fake through conversion to a spectrogram and then based on ViT analysis. This is shown with such a model, showing flexibility. Results show that as deep fakes possess great threats on social media, it is already possible to apply more advanced models with domain-specific preprocessing techniques to fill in this gap. Future improvements shall be aimed at increases in accuracy, optimization of the computational costs, and enhancement of the capacity of systems to detect other synthetic media.

CHAPTER 6

REFERENCES

[1] Mohan, C. R., Prabhakar, P. J., Karthik, N., Kumar, P. M., Rithwik, K., & Prabhas, M. M. S. Enhanced Deepfake Detection on Social Media: Applying CountVectorizer and Random Forest for Optimal Tweet Classification, International Journal of Research in Engineering, IT and Social Sciences, ISSN 2250-0588, Impact Factor: 6.565, Volume 14 Issue 06, June 2024, Page 124-132.Suratkar, S. and Kazi, F., 2023. Deep fake video detection using transfer learning approach. Arabian Journal for Science and Engineering, 48(8), pp.9727-9737.

[2] Pashine, Samay, Sagar Mandiya, Praveen Gupta, and Rashid Sheikh. Deep Fake Detection: Survey of Facial Manipulation Detection Solutions, DOI 10.48550/arXiv.2106.12605, June 2021.

[3] Rafique, Rimsha, Rahma Gantassi, Rashid Amin, Jaroslav Frnda, Aida Mustapha, and Asma Hassan. Deep fake detection and classification using error-level analysis and deep learning, Scientific Reports, SN 2045-2322, Volume 13 Issue 01, DO 10.1038/s41598-023-34629-3, May 2023.

[4] Ismail A, Elpeltagy M, S Zaki M, Eldahshan K. A New Deep Learning-Based Methodology for Video Deepfake Detection Using XGBoost, Sensors, Volume 21, DOI 10.3390/s21165413, August 2021.

[5] Dolhansky, Brian, Joanna Bitton, Ben Pflaum, Jikuo Lu, Russ Howes, Menglin Wang, and Cristian Canton Ferrer. Adversarial Threats to DeepFake Detection: A Practical Perspective, DOI 10.48550/arXiv.2011.09957, November 2020.

[6] Soudy, Ahmed Hatem, Omnia Sayed, Hala Tag-Elser, Rewaa Ragab, Sohaila Mohsen, Tarek Mostafa, Amr A. Abohany, and Salwa O. Slim. Deepfake detection using convolutional vision transformers and convolutional neural networks. SN 1433-3058, Volume 36 Issue 31, DO 10.1007/s00521-024-10181-7, November 2024.

[7] Kumar, Manoj, and Hitesh Kumar Sharma. A GAN-Based Model of Deepfake

Detection in social media, Procedia Computer Science, DOI 10.1016/j.procs.2023.01.191, January 2023, Page 2153-2162.

[8] Jacob Mallet, Laura Pryor, Rushit Dave, Mounika Vanamala. Deepfake Detection Analyzing Hybrid Dataset Utilizing CNN and SVM, Proceedings of the 2023 7th International Conference on Intelligent Systems, DOI 10.1145/3596947.3596954, July 2023, Pages 7-11.

[9] Deng, Liwei and Suo, Hongfei and Li, Dongjie. Deepfake Video Detection Based on EfficientNet-V2 Network, Computational Intelligence and Neuroscience, Volume 2022, DOI 10.1155/2022/3441549, April 2022, Pages 1-13.

[10] Sara Abdali, M. Alex O. Vasilescu, Evangelos E. Papalexakis. Deepfake Representation with Multilinear Regression, ArXiv, August 2021.

[11] Kaur, Achhardeep, Azadeh Noori Hoshyar, Vidya Saikrishna, Selena Firmin, and Feng Xia. "Deepfake video detection: challenges and opportunities." *Artificial Intelligence Review* 57, no. 6 (2024): 159.

[12] Amerini, Irene, Leonardo Galteri, Roberto Caldelli, and Alberto Del Bimbo. "Deepfake video detection through optical flow based cnn." In *Proceedings of the IEEE/CVF international conference on computer vision workshops*, pp. 0-0. 2019.

[13] Yu, Peipeng, Zhihua Xia, Jianwei Fei, and Yujiang Lu. "A survey on deepfake video detection." *Iet Biometrics* 10, no. 6 (2021): 607-624.

[14] Ge, Shiming, et al. "Deepfake video detection via predictive representation learning." *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 18.2s (2022): 1-21.

[15] Cozzolino, Davide, Andreas Rössler, Justus Thies, Matthias Nießner, and Luisa Verdoliva. "Id-reveal: Identity-aware deepfake video detection." In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 15108-15117. 2021.

[16] Wodajo, Deressa, and Solomon Atnafu. "Deepfake video detection using convolutional vision transformer." *arXiv preprint arXiv:2102.11126* (2021).

Plagiarism Checker X - Report

Originality Assessment

7%

Overall Similarity

Date: Nov 25, 2024 Matches: 400 / 5603 words Sources: 18	Remarks: Low similarity detected, consider making necessary changes if needed.	Verify Report:

Sources

1	**https://www.mdpi.com/2673-2688/5/3/56** INTERNET 2%
2	https://link.springer.com/article/10.1007/s00521-024-10181-7 INTERNET 2%
3	https://plat.ai/blog/confusion-matrix-in-machine-learning/ INTERNET 1%
4	https://medium.com/@surajagrahari330/understanding-convolutional-neural-networks-cnns-5440eef4fffd INTERNET <1%
5	https://medium.com/@damian.hill/the-power-of-python-a-comprehensive-guide-to-python-programming-a764ce730023 INTERNET <1%
6	https://rumn.medium.com/precision-recall-and-f1-explained-with-10-ml-use-case-6ef2fbe458e5 INTERNET <1%
7	https://www.geeksforgeeks.org/confusion-matrix-machine-learning/ INTERNET <1%
8	https://www.geeksforgeeks.org/why-python-is-a-high-level-language/ INTERNET <1%
9	https://medium.com/@juanc.olamendy/choosing-the-right-metrics-recall-precision-pr-curve-and-roc-curve-explained-682259961cbe INTERNET <1%
10	https://medium.com/@nirajan.acharya777/understanding-precision-recall-f1-score-and-support-in-machine-learning-evaluation-7ec935e8512e INTERNET <1%
11	https://ieeexplore.ieee.org/document/9577592 INTERNET <1%
12	https://mlres.net/understanding-cnn-layers-convolution-pooling-and-fully-connected-layers/ INTERNET <1%
13	https://packaging.python.org/en/latest/tutorials/packaging-projects/ INTERNET <1%
14	https://www.sciencedirect.com/science/article/abs/pii/S0885230820300474 INTERNET <1%

15	https://stackoverflow.com/questions/58355388/batch-and-subdivisions-in-yolov3 INTERNET <1%
16	https://techcommunity.microsoft.com/blog/machinelearningblog/train-vision-transformer-model-and-run-inference/4241945 INTERNET <1%
17	https://www.sciencedirect.com/science/article/pii/S1051200421002554 INTERNET <1%
18	https://towardsdatascience.com/comprehensive-guide-on-multiclass-classification-metrics-af94cfb83fbd INTERNET <1%

EXCLUDE CUSTOM MATCHES	ON
EXCLUDE QUOTES	OFF
EXCLUDE BIBLIOGRAPHY	OFF