International Journal of Sciences: Basic and Applied Research (IJSBAR)

ISSN 2307-4531
(Print & Online)

https://gssrr.org/index.php/JournalOfBasicAndApplied/index

Deepfake Detection and Analysis Using Fusion Model

Devang Mulye[a], Parakh Pawar[b], Aditya Yadav[c], Poornima S[d]*

[a,b,c,d]*Department of Information Technology, SIES Graduate School of Technology, Navi Mumbai, 400706, India*

[a]*Email:devangmulye@gmail.com*

[b]*Email:parakhpawar623@gmail.com*

[c]*Email:adiiyadav120@gmail.com*

[d]*Email:spoorni82@gmail.com*

Abstract

In today's digital era, deepfake technology presents both innovation and peril, with hyper-realistic synthetic media capable of widespread deception. This research delves into deepfake detection, focusing two deepfake detection models, namely, the VIT image classifier and Meso4 model. Utilizing convolutional neural networks, VIT analyzes images, while Meso4 scrutinizes at a mesoscopic level. A comparative analysis evaluates their effectiveness in discerning authentic from manipulated content. Using the Open Forensics dataset and self-generated content, VIT achieves remarkable accuracy, while Meso4 encounters challenges, such as limited generalization, task-dependent accuracy levels, etc. Therefore, in the proposed work additional features like, Eye movement error detection, Skin texture inconsistency detection and Facial feature inconsistency detection are integrated into a customized model, which results significantly in augmenting accuracy and computation speed compared to above compared models. This research work emphasizes the need for advancing unbiased deepfake detection methods, urging vigilance in safeguarding privacy and security amidst pervasive digital deception.

Keywords: Machine Learning; Image Recognition; Convolution Neural Network; Deepfake Detection.

Received: 9/15/2024
Accepted: 11/15/2024
Published: 11/25/2024

----

* Corresponding author.

1. Introduction

Detecto - Deepfake Detector explores deep learning models for detecting deepfakes, examining Meso4 and ViT models' strengths and limitations. In recent years, deepfake technology's rise has sparked both fascination and concern due to its potential for misinformation [2]. Deepfake detection is crucial for maintaining digital media integrity, combating misinformation's harmful effects. Various detection models, including DeepFace Lab and DeepFake Detection, employ diverse methodologies to identify deepfake manipulation. Despite their strengths, continual refinement is necessary to address emerging challenges and adversarial tactics. Detecto proposes a user-centric approach, offering a mobile application empowering individuals to analyze media for signs of manipulation. By integrating Vision Transformers (ViT) and a custom model, Detecto delivers accurate deepfake detection while prioritizing user choice and customization. Through rigorous evaluation and real-world validation, Detecto aims to contribute to deepfake detection advancement and promote a safer digital environment.The research made along the literature survey contributed with vast number of details regarding various other models, datasets and their pros and cons [1]. This study tackles the impact of deepfake media on public discourse and legal proceedings, emphasizing the need for accurate detection in digital forensics. It compares face-detection classifiers—Custom CNN, VGG19, and DenseNet-121—using an augmented dataset of real and fake faces. Data augmentation improves performance and reduces computational resources. Preliminary results show VGG19 achieves the highest accuracy at 95%, highlighting its potential for detecting deepfake media and aiding in identity protection [8]. This study addresses deepfakes' threats to privacy, democracy, and security. It emphasizes urgent detection methods using machine learning and deep learning techniques like SVM, random forests, and CNNs. It also highlights the need for policies, regulations, individual actions, and technological advancements to combat deepfakes [12]. This work presents a novel Deepfake detection approach using a Convolutional Vision Transformer (CVT), combining CNN and ViT components. The CNN extracts features, processed by the ViT's attention mechanism for categorization. Trained on the DFDC dataset, the model achieves 91.5% accuracy, an AUC of 0.91, and a loss of 0.32, demonstrating competitive performance [22]. This paper presents an efficient vision transformer for DeepFake detection, addressing CNN limitations. It combines CNN features with patch-based positioning for local and global image feature capture, and a distillation token for better generalization. The model outperforms SOTA, achieving higher AUC and F1 scores on the DFDC test dataset. Ensemble learning further enhances performance, with high AUC and F1 scores on the Celeb-DF (v2) dataset, demonstrating its effectiveness.These findings underscore the pervasive threat of deepfake technology, highlighting its potential for misinformation, fraud, and reputation damage. The rapid evolution of deepfake tools necessitates continuous improvement in detection techniques. Combining advanced algorithms like CNNs, Vision Transformers, and ensemble learning, along with integrating human judgment, enhances detection robustness. A holistic approach, including technological advancements, public education, and regulatory measures, is essential to mitigate societal risks and safeguard digital content and identity.

2. Methodology

The proposed workflow for deepfake detection is divided into two phases. Phase 1 involves analyzing two models: Meso4, which uses convolutional layers and heatmap thresholds for efficient and interpretable detection, and ViT, which employs a Vision Transformer with self-attention mechanisms for high accuracy and generalization. The Phase 2 introduces an ensemble model that combines the strengths of Meso4 and ViT. This model enhances detection by analyzing key facial features such as eye movement, skin texture, and facial expressions, thereby improving accuracy and robustness in identifying deepfakes.

2.1. Phase 1: Analysis of Meso4 and ViT model

2.1.1. Meso4 model

The Meso4 model [3], short for Mesoscopic Model 4, is a deep learning architecture (Fig 1) specifically designed for the task of detecting deepfake images. It operates at a mesoscopic level, which refers to a scale intermediate between microscopic (fine-grained details) and macroscopic (overall structure). The architecture of the Meso4 model typically consists of several convolutional layers followed by pooling layers and fully connected layers [3].

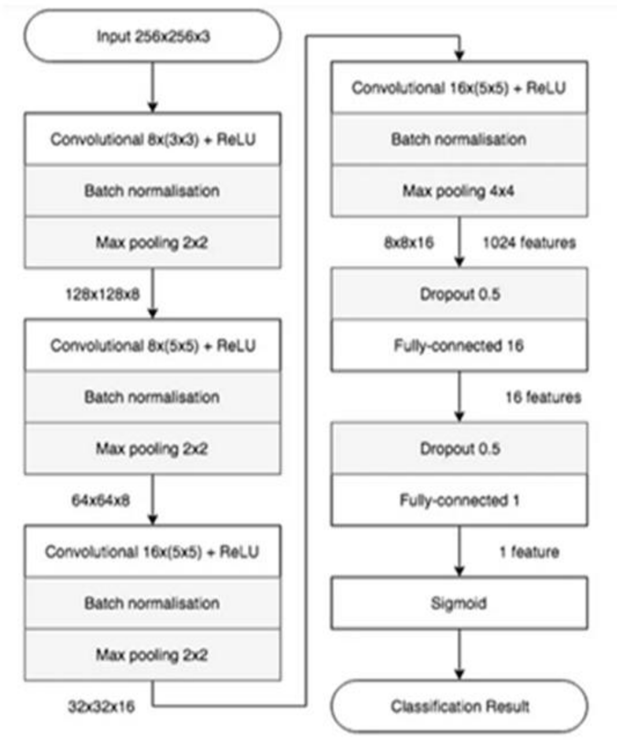

Figure 1: Meso4 Architecture [13].

2.1.2. Meso4 model Architecture

- Input Layer: Takes an input image as a matrix of pixel values, standardized to a fixed size.
- Convolutional Layers: Core of the Meso4 model, performing convolution operations to extract features. Typically includes four layers, each with multiple filters generating feature maps.
- Activation Functions: Introduces non-linearity post-convolution using functions like ReLU or Leaky ReLU to learn complex patterns.

- Pooling Layers: Reduces spatial dimensions of feature maps via max pooling, retaining important information.
- Fully Connected Layers: Performs high-level feature extraction and classification. The output layer has a single neuron with a sigmoid activation, indicating the probability of the image being a deepfake.
- Training and Optimization: Trained on labeled images (authentic or manipulated). Uses loss function minimization and optimization techniques like SGD or Adam to improve performance. The Meso4 model captures mesoscopic-level features for effective deepfake detection, balancing computational efficiency and accuracy.

2.1.3. Meso4 model stepwise workflow

- Load the Image: First, we load the image that we want to analyze.
- Convert Image to NumPy Array: Next, we convert this image into a NumPy array, which represents the image in matrix form.
- Generate Heatmap: We then generate a heatmap from this matrix. The heatmap helps us visualize important areas of the image.
- Determine Threshold Value: Using the heatmap, we derive a threshold value. This value will be crucial for making decisions later on.
- Initialize the Model: After that, we initialize the Meso4 model and load its pretrained weights.
- Preprocess the Image: We preprocess the image to prepare it for prediction.
- Calculate Confidence Level: With the threshold value, we calculate the confidence level of the model's prediction.
- Classify the Image: Based on the confidence level, the model classifies the image as either genuine or manipulated.
- Accentuate Critical Regions: The heatmap helps us highlight critical or distinctive regions in the image.
- Enhance Interpretability: By fusing the heatmap with the model, we improve the model's ability to identify important regions, making its decisions easier to understand.
- Adjust Model Attention: The model dynamically adjusts its focus based on the heatmap information, which helps reduce false positives and negatives.
- Improve Model Robustness: This process makes the model more resistant to adversarial attacks, enhancing its accuracy and resilience in detecting deepfakes.

2.1.4. ViT Model

The Vision Transformer (ViT) model is a state-of-the-art deep learning architecture specifically designed for image classification tasks (Fig 2). Unlike traditional convolutional neural networks (CNNs), which rely on convolutional operations for feature extraction, the ViT model utilizes self-attention mechanisms to capture long-range dependencies within images [20]. Here's a detailed breakdown of the architecture of the ViT model:

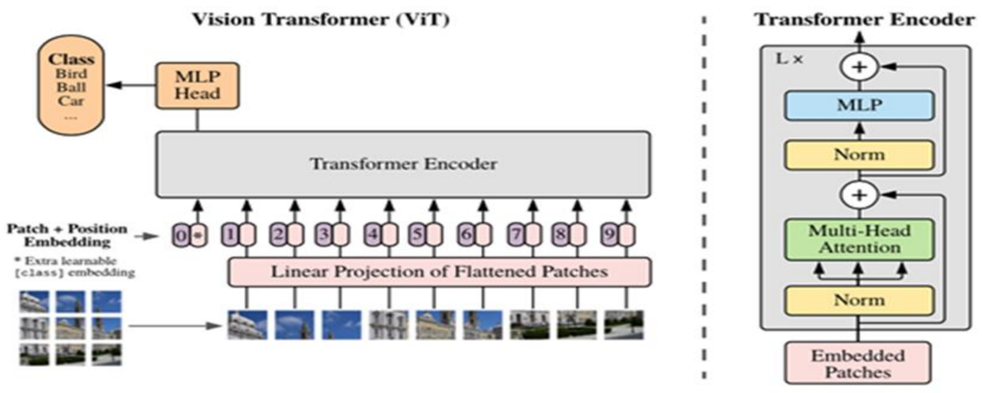

Figure 2: ViT model Architecture [14]

2.1.5. ViT Model Architecture

- Patch Embeddings: The input image is divided into fixed-size patches, linearly embedded into lower-dimensional representations. These are flattened into a sequence of tokens, each representing a patch.
- Positional Encodings: Added to patch embeddings to convey spatial relationships, using learned or predefined sinusoidal functions.
- Transformer Encoder Layers: Multi-head Self-Attention Mechanism: Allows the model to attend to different parts of the input, capturing global dependencies by computing attention scores between tokens.
- Feedforward Neural Network (FFNN): Applies non-linear transformations on each token using fully connected layers with ReLU activation.
- Layer Normalization and Residual Connections: Applied after each sub-module to stabilize training and facilitate gradient flow, mitigating the vanishing gradient problem.
- Classification Head: Final token representations are aggregated (mean pooling or classification token extraction) and fed into a feedforward neural network classifier with a softmax activation to assign class probabilities.
- Training and Optimization: Trained using supervised learning to minimize a loss function (e.g., cross-entropy loss). Uses optimization techniques like SGD or Adam to iteratively update model parameters. The ViT model uses self-attention mechanisms to capture global dependencies in images, achieving state-of-the-art performance in image classification.

2.1.6. ViT model stepwise workflow

- Start with Dataset Loading: The first step begins by loading dataset using the PIL library. This initial setup is crucial as it ensures efficient handling of data, which is essential for robust model training and evaluation.
- Mapping Labels and IDs: Once loaded, map labels and IDs within the dataset. Here, a label of zero denotes fake examples, while one indicates real instances. This mapping provides the foundational

ground truth necessary for supervised learning.

- Splitting for Training and Testing: With labels established, the dataset is methodically split into separate training and testing sets. This division allows us to assess how well this model generalizes beyond the data it's been trained on, a critical measure of its effectiveness.
- Integrating the Vision Transformer Model: Moving forward, integrated a pre-trained Vision Transformer (ViT) model into this workflow. This involves loading the model and setting up a processor tailored to its specifications, ensuring it can effectively extract features and classify images as required.
- Aligning Image Sizes: To optimize the ViT model's performance during both training and evaluation phases, align the sizes of input images with the expectations of the model. This alignment enhances the flow of information and computational efficiency.
- Enhancing Robustness with Image Transformations: The next step involves applying various image transformations. These modifications bolster the model's ability to handle diverse dataset variations, ultimately improving its robustness and ability to generalize effectively.
- Defining the Collate Function: A collate function is defined to streamline the process of batching data for model training. This function optimizes efficiency and ensures data is processed in a structured manner during training sessions.
- Loading Pre-trained Weights: Leveraging knowledge gained from previous training sessions, load pre-trained weights into ViT model. This step accelerates convergence during subsequent training phases, saving valuable time and computational resources.
- Training and Evaluation: With preparations complete, the model undergoes rigorous training and evaluation using a validation set. This iterative process provides crucial insights into the model's learning progress and its ability to accurately predict outcomes on unseen data.
- Real-Time Predictions: Demonstrating practical application, upload an image into the pipeline to showcase the ViT model's real-time prediction capabilities. This step underscores its potential for immediate decision support in various scenarios.
- Interpreting Results: Finally, results from the model are presented in a clear and concise manner, displaying scores and corresponding labels. This summary ensures easy interpretation of predictions, supporting informed decision-making based on the model's outputs.

2.1.7. Comparing the Two models

Meso4: While Meso4 offers advantages like efficiency, interpretability, and lower resource requirements, its accuracy can vary depending on the specific deepfake detection task (Table 1). This makes it a suitable choice for real-time applications that prioritize interpretability and speed over peak accuracy [15].

Table 1: Characteristics Comparison of ViT and Meso4 models

Feature	**Meso4**	**ViT**
Model Size & Complexity	Compact, Moderate	Large, High
Training Time & Inference Speed	Efficient, Fast	Slower, Time-consuming
Accuracy	Task-dependent	High
Generalization	Limited	Excellent
Resource Requirements	Low	High
Interpretability	High	Lower
Transfer Learning	Limited	Strong
Pre-trained Models	Not Common	Widely Available
Video Compatibility	Suitable	Limited

ViT: ViT excels in accuracy and generalization, making it powerful for handling diverse datasets. However, its larger size translates to longer training times and slower inference speeds. Additionally, ViT may not be ideal for video applications due to compatibility limitations. However, its strong transfer learning capabilities and the availability of pre-trained models enhance its versatility for various image classification tasks [20].

To address the limitations of each individual model, the study proposes a hybrid approach. This method leverages ViT's superior image processing for analyzing individual video frames while addressing computational concerns. Furthermore, a novel model is introduced that focuses on analyzing three key features crucial for deepfake detection: eye movement, skin texture and facial expressions inconsistencies. By analyzing and comparing these features across video frames, this model aims to boost detection accuracy. This facial feature analysis model can then be combined with ViT for further improvement in image classification within the context of deepfake detection [15].

2.2. Phase 2: Proposing an Ensemble Model

To enhance performance, a custom ensemble model is proposed in Fig 3, combining the strengths of the Meso4 and ViT models. This model addresses computational limitations and leverages ViT's superior image processing. It focuses on analyzing three key facial features to improve deepfake detection.

- Eye Movement Error Detection: Identifies inconsistencies in eye movement patterns.
- Skin Texture Detection: Uncovers manipulation signs through skin irregularities.
- Facial Expression Detection: Detects unnatural facial expressions.

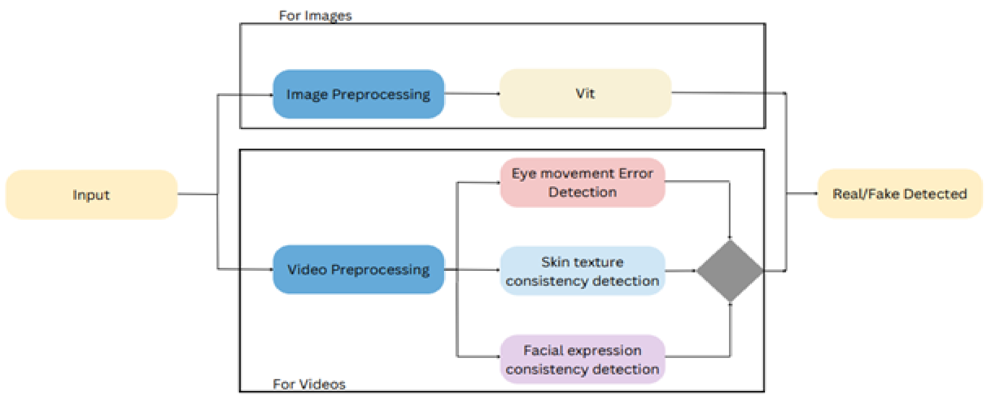
For Images
Image Preprocessing
Vit
Input
Eye movement Error Detection
Video Preprocessing
Skin texture consistency detection
Facial expression consistency detection
For Videos
Real/Fake Detected

Figure 3: Proposed Ensemble model diagram

The model analyzes and compares these features across video frames for enhanced detection accuracy. Trying to integrate this custom model with ViT improves image classification, despite increased computational demands, offering superior accuracy in identifying deepfake content.

2.2.1. Eye movement error detection

This feature detection work detects potential deepfakes in a video by analyzing the blinking rate using pre-trained classifiers from OpenCV, which leverage Haar feature detection to identify faces and eyes. Initially, the classifiers for faces and eyes are loaded. The code then processes each video frame by converting it to grayscale for better eye detection and applying noise reduction to improve image quality. The face classifier identifies faces, and within these faces, the eye classifier searches for eyes. When eyes are first detected, a timer is initiated to measure the duration between blinks. The code tracks frames where eyes are not detected after being seen earlier, counting these moments as blinks. After processing the entire video, the blinking rate is calculated by dividing the total blinks by the elapsed time. This rate is compared to a predefined threshold, as deepfakes might struggle to realistically simulate natural blinking patterns, potentially resulting in a lower blinking rate than expected in a real human. If the calculated rate falls below the threshold, the code raises a flag suggesting a possible deepfake. The threshold value can be adjusted based on the specific use case and desired sensitivity for deepfake detection [16].

2.2.2. Algorithm for eye movement error detection code

a. Load pre-trained classifiers for face and eye detection from OpenCV.

- Define a threshold for blinking rate (e.g., blink_threshold).
- Initialize variables:
- blink_count to track the number of detected blinks.
- total_frames_processed to count the total frames analyzed.

- elapsed_time to measure the time elapsed for blinking duration.

b. Process Video:

- For each frame in the video:
- Convert the frame to grayscale.
- Apply a noise reduction filter to improve image quality.

c. Face Detection:

- Use the face classifier to identify faces in the frame.
- If a face is detected:
- Focus on the region of interest (ROI) containing the face

d. Eye Detection:

- Utilize the eye classifier to detect eyes within the ROI.
- If eyes are detected:
- If it's the first time eyes are detected, start a timer to measure blinking duration.
- If eyes are not detected after being seen earlier, count it as a blink and increment blink_count.

e. Calculate Blinking Rate:

- After processing all frames:
- Calculate the elapsed time from the timer.
- Calculate the blinking rate by dividing blink_count by the elapsed time.

f. Detect Deepfake:

- Compare the blinking rate to the predefined threshold (blink_threshold).
- If the blinking rate is below blink_threshold:
- Raise a flag indicating a potential deepfake presence.

2.2.3. Skin Texture Consistency Detection

This feature detection work uses libraries like OpenCV, NumPy, and PIL for robust image processing and deep learning to analyze skin texture consistency in videos. A trained Haar Cascade classifier detects individuals within video frames, forming the basis for skin texture analysis. Each frame is examined using this classifier, and skin texture is analyzed with techniques like Gaussian blur and Laplacian filtering to identify variations. OpenCV's VideoCapture function loads the video, and a while loop processes each frame to detect individuals and assess skin texture. Performance is optimized with parallel processing via ThreadPoolExecutor. The code tracks skin texture consistency across frames, calculating percentages of consistent and inconsistent textures. These metrics guide the classification of the video's authenticity, helping to identify potential manipulation by highlighting skin texture variations indicative of digital deception.

2.2.4. Algorithm for skin texture Consistency detection in videos

a. Initialization

- Load pre-trained Haar Cascade classifier for full-body detection.
- Define parameters such as threshold values for consistency assessment.
- Initialize variables to track consistent and inconsistent texture predictions.

b. Video Loading:

- Utilize OpenCV's VideoCapture function to load the input video.
- Prepare the video for analysis, ensuring seamless access to frame data.

c. Frame Processing Loop

- Iterate through each frame of the video:
- Apply the pre-trained Haar Cascade classifier to detect individuals within the frame.
- For each detected individual:
- Analyze skin texture using advanced techniques such as Gaussian blur and Laplacian filtering to highlight variations.
- Evaluate consistency in skin texture across frames to track counts of consistent and inconsistent predictions.

d. Consistency Assessment

- Assess texture variations within consecutive frames to determine consistency.
- Track counts of consistent and inconsistent texture predictions to unravel dynamics of skin texture nuances throughout the video.

e. Video Classification

- Calculate percentages of frames with consistent and inconsistent skin texture predictions.
- Utilize these metrics as indicators for video authenticity.
- Guide the classification process to determine whether the video is authentic or potentially manipulated, based on the interplay between skin texture variations and digital deception.

2.2.5. Facial expression inconsistency detection

This feature detection work uses a pre-trained Vision Transformer (ViT) model to analyze facial expression consistency in videos for deepfake detection. It imports essential libraries like OpenCV, Torch, PIL, and transformers for advanced image processing and model loading. The code loads a pre-trained ViT model from the "trpakov/vit-face-expression" checkpoint for high-performance image classification. Each video frame is processed into a PIL image, pre-processed by the ViT feature extractor, and analyzed for facial expressions. Using parallel processing via ThreadPoolExecutor, the code efficiently examines multiple frames simultaneously. It tracks the consistency of facial expressions between consecutive frames, tallying counts of consistent and inconsistent predictions. It then calculates the percentages of these consistencies, comparing them to a predefined threshold (89%). If consistency falls below this threshold, the video is flagged as fake; otherwise, it is deemed authentic, highlighting the role of facial expression dynamics in detecting digital deception.

2.2.6. Algorithm for facial expression Consistency detection in videos

a. Initialization:

- Load pre-trained ViT model for facial expression classification.
- Define a threshold for consistency assessment (e.g., 89%).
- Initialize variables to track consistent and inconsistent expression predictions.

b. Video Loading:

- Utilize OpenCV's VideoCapture function to load the input video.
- Specify the file path of the video to be processed.

c. Frame Processing Loop:

- Iterate through each frame of the video using a while loop:
- Convert the frame to a PIL Image.
- Preprocess the frame with ViT for facial expression prediction.
- Predict facial expressions for the frame.

d. Consistency Check:

- Assess consistency between predicted facial expressions in consecutive frames:
- Track counts of consistent and inconsistent expression predictions.
- Compare facial expression predictions between consecutive frames to determine consistency.

e. Video Classification:

- Calculate the percentage of frames with consistent facial expression predictions.
- Classify the video as fake or real based on the predefined threshold:
- If the percentage of consistent predictions exceeds the threshold, classify the video as real.

If the percentage falls below the threshold, classify the video as potentially manipulated or fake.

2.2.7. Integrating the models to form a Multimodal Deepfake Detection System

This feature detection work adopts a multimodal approach to deepfake detection, employing various functionalities synergistically to classify videos as "real" or "fake." Here's how it works:

• Individual Feature Analysis:

- Separate functions like detect_eye_blink, detect_skin_texture, and detect_facial_expression are defined.

- Each function analyzes video frame-by-frame, extracting relevant information (e.g., detect_eye_blink identifies faces and eyes, monitoring blinking patterns).

• Independent Processing:

- Functions operate independently, providing separate "real" or "fake" classifications.

- The code utilizes a divide-and-conquer approach, combining weighted insights from each function to make the final decision.

This approach enhances deepfake detection accuracy by leveraging diverse analysis methods, catering to both images and videos effectively.

3. Experimentation and Results

Dataset Details: This section outlines the datasets used in comparing the Meso4 and ViT classifiers and enhancing the custom model. Each dataset was selected to meet specific model requirements:

Meso4 Dataset: A curated dataset of 2,043 images categorized by difficulty. It includes 960 fake and 1,081 real images for balanced training, testing, and validation.

ViT Dataset (Open Forensics): Comprising 190,305 images, it includes 70,000 fake and 70,000 real images for training, with separate sets for testing (5,492 fake, 5,413 real) and validation (19,600 fake, 19,800 real).

Ensemble Model Testing Dataset (Kaggle Deepfake Images): Includes 3,462 fake images, 700 real images, 3,462 fake videos, and 700 real videos for training, validation (678 fake images, 136 real images, 678 fake videos, 136 real videos), and testing (790 image directories, 790 video files).

DFDC Dataset: Testing utilized 5,000 deepfake and 5,000 real videos, with training (15,000 deepfake, 15,000 real) and validation (2,500 deepfake, 2,500 real) sets.

These diverse datasets enabled robust training and evaluation of the Meso4 and ViT models in image and video classification tasks. Results are summarized in Table 2, ensuring fair comparison based on unique training and testing conditions. The research is divided into two phases: analyzing Meso4 and ViT models for deepfake image detection in phase one, and implementing a fusion model for deepfake video detection in phase two. The ViT model excelled on its native dataset, accurately identifying 99% of fake images but struggled on a random dataset, achieving only 73% accuracy for fake images and 30% for real ones. In contrast, the MESO-4 model performed well on its own dataset, with 82% accuracy for fake images and 72% for real ones, but struggled with a random dataset, achieving 100% accuracy for fake images but 0% for real ones. These results highlight the divergent capabilities of both models and emphasize the need for thorough evaluation methods (Table 3).

Table 2: Testing datasets on Meso4 and ViT models

Dataset	No of tested Images	Model	Real Detected	Fake Detected	Accuracy
VIT dataset	100(fake)	Vit	1	99	99%
VIT dataset	100(real)	Vit	99	1	99%
Random dataset	100(fake)	Vit	27	73	73%
Random dataset	100(real)	Vit	30	70	30%
Meso4 dataset	100(fake)	Meso4	18	82	82%
Meso4 dataset	100(real)	Meso4	72	28	72%
Random dataset	100(fake)	Meso4	0	100	100%
Random dataset	100(real)	Meso4	0	100	0%

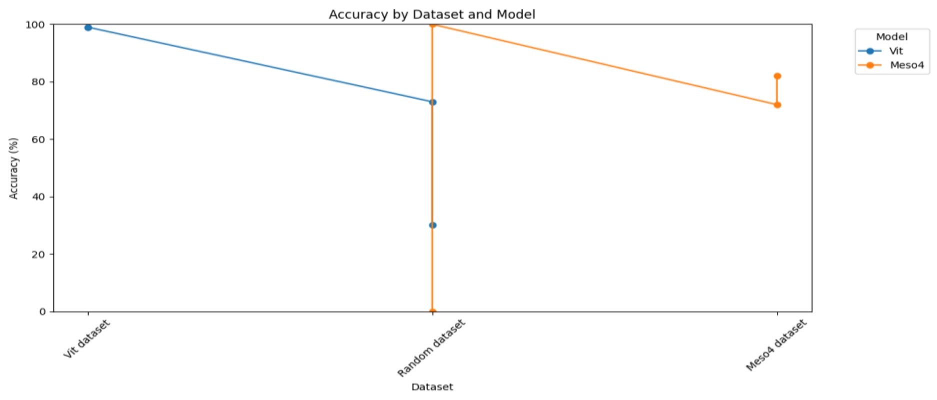

Figure 4: Accuracy of Dataset and Models

The graph (Fig 4) compares "Vit" and "Meso4" models on different datasets. "Vit" performs slightly better on its native dataset, while "Meso4" excels on the "Random dataset." This highlights dataset influence on model performance, showing "Meso4's" adaptability and robustness.

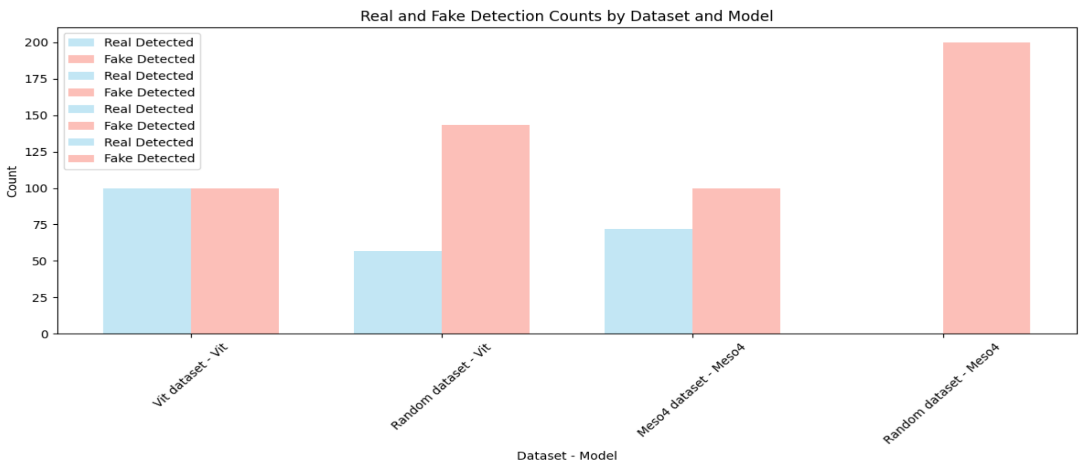

Figure 5: Analysis of Real and Fake Image Detection

The graph (Fig 5) shows dataset-model pairs. "Vit" excels in real image detection but struggles with fakes, especially on the "Random dataset." Conversely, "Meso4" outperforms on the "Random dataset" in real image detection. This reveals performance variations across datasets, emphasizing strengths and weaknesses in detecting real and fake image.

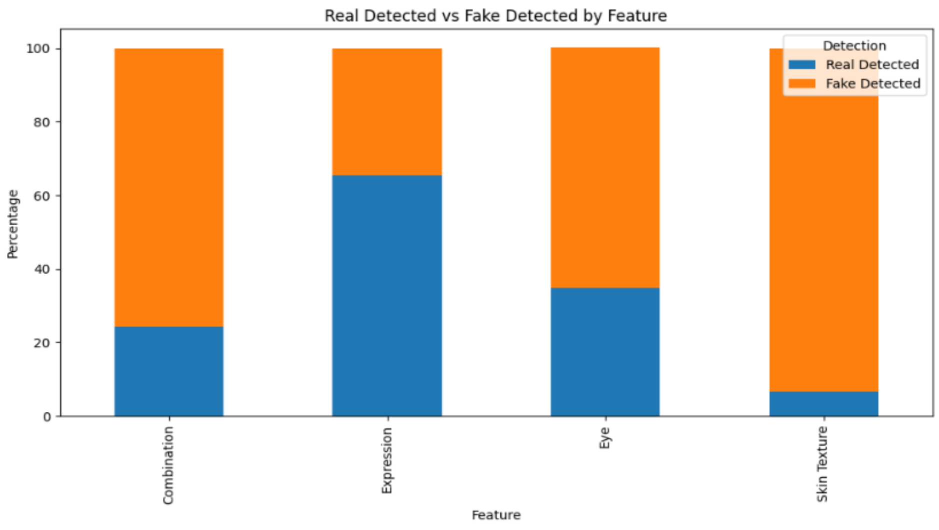

Figure 6: Real and Fake Detection Performance through Individual Features

The graph (Fig 6) compares real and fake video detection across facial features in "Deepfake Images" and "DFDC." It shows higher fake video detection in "Skin Texture" and "Expression" features, especially prominent in "Skin Texture." Conversely, "Eye" features detect more real videos. This reveals varying detection effectiveness across features, offering insights into accuracy distribution within each feature category across datasets.

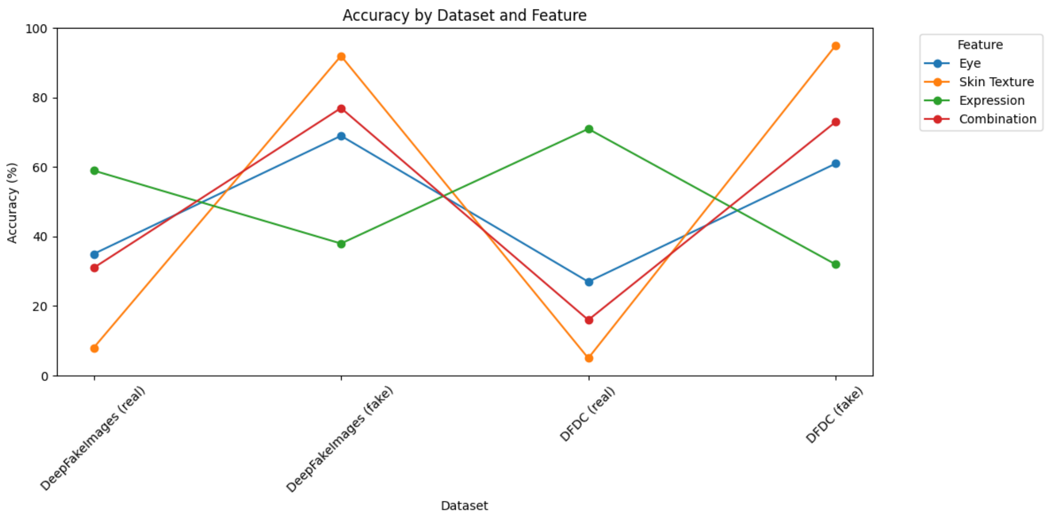

Figure 7: Analysis of accuracy of different features

Table 3: Performance of models with DFDC and Deepfake Images datasets

Dataset	**No. of Videos**	**Model**	**Real Detected**	**Fake Detected**	**Accuracy**
Deepfake Images	90(real)	Eye	32	58	35%
Deepfake Images	110(fake)	Eye	34	76	69%
DFDC	76(real)	Eye	21	56	27%
DFDC	292(fake)	Eye	111	181	61%
Deepfake Images	90(real)	Skin Texture	8	82	8%
Deepfake Images	110(fake)	Skin Texture	8	102	92%
DFDC	40(real)	Skin Texture	2	38	5%
DFDC	87(fake)	Skin Texture	4	83	95%
Deepfake Images	37(real)	Expression	22	15	59%
Deepfake Images	47(fake)	Expression	29	18	38%
DFDC	42(real)	Expression	30	12	71%
DFDC	50(fake)	Expression	34	16	32%
Deepfake Images	90(real)	Fusion	28	62	31%
DeepFakeImages	100(fake)	Fusion	23	77	77%
DFDC	77(real)	Fusion	12	65	16%
DFDC	90(fake)	Fusion	24	66	73%

The graph (Fig 7) shows real and fake video detection accuracy across facial features in "Deepfake Images" and "DFDC." Each line represents a feature like "Eye" or "Skin Texture," with points indicating accuracy. "Skin Texture" notably detects fake videos better, especially in "Deepfake Images." Conversely, "Expression" shows balanced accuracy. It offers insights into feature effectiveness, highlighting accuracy variations across datasets.

4. Findings

Model Performance Varied: Each model showed different accuracies across datasets. For example, the Eye Movement Error Detection Model performed better on DFDC than on Deepfake Images, while the Skin Texture Consistency Detection Model showed the opposite trend.

Dataset Influence: Unique dataset characteristics affected model performance. DFDC was more challenging for some models than Deepfake Images, leading to decreased accuracy rates.

Fusion Model Effectiveness: Combining Eye Movement, Skin Texture, and Facial Expression Detection showed mixed results. While it performed well in detecting fake videos in some cases, it struggled with real videos, especially on DFDC.

Importance of Evaluation: Comprehensive evaluation across datasets is crucial for understanding model efficiency. It reveals strengths, weaknesses, and areas for improvement.

5. Conclusions

This work represents a pivotal advancement in the fight against digital deception by adopting a multi-faceted approach that leverages the strengths and addresses the limitations of existing models like Meso4 and ViT. The proposed workflow is divided into two phases: Phase 1 analyzes the Meso4 and ViT models, where Meso4 offers efficiency and interpretability for real-time applications, while ViT provides high accuracy and generalization capabilities for diverse datasets. Phase 2 introduces an ensemble model that combines the strengths of both models, enhanced by a custom facial feature model targeting key areas such as eye movement, skin texture, and facial expressions. The results show that ViT excelled on its native dataset with 99% accuracy but struggled on random datasets, whereas Meso4 performed well on its dataset with 82% accuracy but failed on random datasets for real images. The fusion model achieved varied results, performing well in detecting fake videos but less so for real ones. Meso4's real-time capabilities and ViT's superior performance on diverse datasets are complemented by the enhanced detection accuracy from the custom model. However, performance varied significantly across datasets, and higher computational demands posed challenges for real-time implementation. Future work should focus on algorithm optimization, expanding datasets, enhancing real-time detection capabilities, and ensuring ethical use and regulatory compliance. In conclusion, this work offers a robust framework for identifying and combating digital deception by integrating Meso4 and ViT with a custom facial feature model, ultimately contributing to a more trustworthy and responsible digital future.

References

[1] Taeb, Maryam, and Hongmei Chi. "Comparison of deepfake detection techniques through deep learning." Journal of Cybersecurity and Privacy 2.1 (2022): 89- 106.

[2] Lyu, Siwei. "Deepfake detection: Current challenges and next steps." 2020 IEEE international conference on multimedia & expo workshops (ICMEW). IEEE, 2020.

[3] Khichi, Manish. "DEEPFAKE OR REAL IMAGE PREDICTION USING MESONET." PhD diss., 2021.

[4] Liang, J., Wang, D., & Ling, X. (2021, August). Image classification for soybean and weeds based on VIT. In Journal of Physics: Conference Series (Vol. 2002, No. 1, p. 012068). IOP Publishing.

[5] Westerlund, M. (2019). The emergence of deepfake technology: A review. Technology innovation management review, 9(11).

[6] George, A. S., & George, A. H. (2023). Deepfakes: The Evolution of Hyper realistic Media Manipulation. Partners Universal Innovative Research Publication, 1(2), 58-74.

[7] Turan, S. G. (2021). Deepfake and digital citizenship: A long-term protection method for children and youth. In Deep fakes, fake news, and misinformation in online teaching and learning technologies (pp. 124-142). IGI Global.

[8] Shahzad, H. F., Rustam, F., Flores, E. S., Luís Vidal Mazón, J., de la Torre Diez, I., & Ashraf, I. (2022). A Review of Image Processing Techniques for Deepfakes. Sensors, 22(12), 4556.

[9] Silva, S. H., Bethany, M., Votto, A. M., Scarff, I. H., Beebe, N., & Najafirad, P. (2022). Deepfake forensics analysis: An explainable hierarchical ensemble of weakly supervised models. Forensic Science International: Synergy, 4, 100217.

[10] Wang, Z., Guo, Y., & Zuo, W.(2022). Deepfake forensics via an adversarial game. IEEE Transactions on Image Processing, 31, 3541-3552.

[11] Joseph, Z., & Nyirenda, C. (2021, May). Deepfake detection using a twostream capsule network. In 2021 ISTAfrica Conference (IST-Africa) (pp. 18). IEEE.

[12] Wodajo, D., & Atnafu, S. (2021). Deepfake video detection using convolutional vision transformer. arXiv preprint arXiv:2102.11126.

[13] https://imgs.search.brave.com/7UnnKff9hMiaIIiwho-CYmjLmWP55BlPDv3aDeiyjEE/rs:fit:500:0:0/g:ce/aHR0cHM6Ly9naXRo/dWIuY29tL01hbGF5/QWdyL01lc29OZXQt/RGVlcEZha2VEZXRl/Y3Rpb24vcmF3L21h/aW4vaW1ncy9tb2Rl/bF9zY2hlbWF0aWMu/cG5n

[14] https://imgs.search.brave.com/YRwCih_N70FJHplKPoOq91gikmZEPbWrOh1-6KHYAhg/rs:fit:500:0:0/g:ce/aHR0cHM6Ly9lZGl0/b3IuYW5hbHl0aWNz/dmlkaHlhLmNvbS91/cGxvYWRzLzM1MDA0/Vml0LnBuZw

[15] Coccomini, D.A., Messina, N., Gennaro, C., Falchi, F. (2022). Combining EfficientNet and Vision Transformers for Video Deepfake Detection. In: Sclaroff, S., Distante, C., Leo, M., Farinella, G.M., Tombari, F. (eds) Image Analysis and Processing – ICIAP 2022. ICIAP 2022. Lecture Notes in Computer Science, vol 13233. Springer, Cham. https://doi.org/10.1007/978-3-031-06433-3_19

[16] Pashine, S., Mandiya, S., Gupta, P. and Sheikh, R., 2021. Deep fake detection: Survey of facial manipulation detection solutions. arXiv preprint arXiv:2106.12605.

[17] D. Pan, L. Sun, R. Wang, X. Zhang and R. O. Sinnott, "Deepfake Detection through Deep Learning," 2020 IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (BDCAT), Leicester, UK, 2020, pp. 134-143, doi: 10.1109/BDCAT50828.2020.00001.

[18] Ahmed, S.R.A. and Sonuç, E., 2023. Deepfake detection using rationale-augmented convolutional neural network. Applied Nanoscience, 13(2).

[19] Das, S., Seferbekov, S., Datta, A., Islam, M.S. and Amin, M.R., 2021. Towards solving the deepfake problem: An analysis on improving deepfake detection using dynamic face augmentation. In Proceedings of the IEEE/CVF.