Advancing DeepFake Detection with Real and Synthetic Celebrity Videos

Nishant Nayak
Msc in Data Analytics
National Collge of Ireland
Dublin, Ireland
x22248242@student.ncirl.ie

Devendrakumar
Msc in Data Analytics
National Collge of Ireland
Dublin, Ireland
x22248242@student.ncirl.ie

Sanidhya Bajaj
Msc in Data Analytics
National College of Ireland
Dublin, Ireland
x23251867@student.ncirl.ie

Akanksha Govindani
Msc in Data Analytics
National College of Ireland
Dublin, Ireland
x23134640@student.ncirl.ie

***Abstract*— AI-generated face-swapping videos, commonly called Deepfakes, pose a growing threat to the reliability of online content. The development and assessment of deepfake detection techniques require large-scale datasets. However, existing deepfake datasets often suffer from poor visual quality and fail to accurately represent the manipulated videos found online. To address this gap, we introduce Celeb-DF, a large-scale and challenging deepfake video dataset comprising 1209 high-quality celebrity deepfake videos created through an enhanced synthesis process. Additionally, we conduct an in-depth evaluation of various deepfake detection methods and datasets, highlighting the increased level of difficulty introduced by Celeb-DF.**

Keywords*— *ResNet50, MobileNetV3, ShuffleNetV2, EfficientNet_Lite, ViT (vision transformer), ReLU (Rectified Linear Unit), DataLoader, Xception, CNN (Convolutional Neural Network), PIL (Python Imaging Library), Softmax, CrossEntropyLoss, LSTM (Long Short-Term Memory), Adam (Adaptive Moment Estimation), RMSProp (Root Mean Square Propagation), ROC (Receiver Operating Characteristic), False Positive Rate (FPR), True Positive Rate (TPR), Deepfake Detection.

I. Introduction

The proliferation of deepfake technology, particularly synthetic celebrity videos, threatens digital trust through hyper-realistic forgeries that evade conventional detection methods [7]. While existing approaches, such as Xception and Vision Transformers, achieve high accuracy, they often suffer from computational inefficiency, limited generalizability, and high parameter counts, which hinder their real-world deployment. This study advances deepfake detection by evaluating four lightweight yet powerful architectures—ResNet50, MobileNetV3, Shufflenet_v2, and EfficientNet_lite—on a hybrid dataset of real and synthetic celebrity videos. These models, known for their robust feature extraction capabilities and varying computational efficiencies, provide a comprehensive approach to tackling the challenges of detecting manipulated celebrity videos. By training and testing on both real and synthetic datasets, this research aims to enhance detection accuracy and generalizability while addressing computational constraints for real-world applications [16].

The term deepfake, derived from the combination of 'deep learning' and 'fake,' denotes highly photorealistic synthetic video or image content generated through deep learning methodologies. It was first popularized by an anonymous Reddit user in late 2017.

II. Research Question

How can deep learning models be optimized to achieve high accuracy in deepfake detection (classify real and fake videos) while maintaining computational efficiency?

III. Related Work

Existing studies in deepfake detection predominantly focus on convolutional neural networks (CNNs) and transformer-based models. Prior research has explored GAN-based detection strategies that leverage spatial and temporal inconsistencies. The FaceForensics++ dataset, a widely referenced benchmark, has played a crucial role in developing detection algorithms [18]. One of the foundational studies that introduced FaceForensics++ highlighted the effectiveness of CNN-based models such as XceptionNet [10]. However, this approach exhibited limitations in generalization to unseen manipulations. Transformer models, such as ViTs, have also been explored for deepfake detection, showing promise in capturing long-range dependencies but often demanding extensive computational resources [12]. More recent studies have explored lightweight deep learning models. A comprehensive survey on deepfake detection highlighted the potential of MobileNet and ShuffleNet architectures for real-time applications. However, these models often trade accuracy for computational efficiency. Fine-tuning MobileNetV3 for real and fake image classification showcased its potential in deepfake detection, yet the approach lacked robustness across diverse datasets [3]. Critically, prior work has struggled with balancing efficiency and accuracy. High-performance models like EfficientNet achieve impressive accuracy but at a computational cost unsuitable for edge devices [17]. Thus, this research builds upon these findings by integrating multiple architectures to optimize both performance and scalability.

IV. Proposed Model

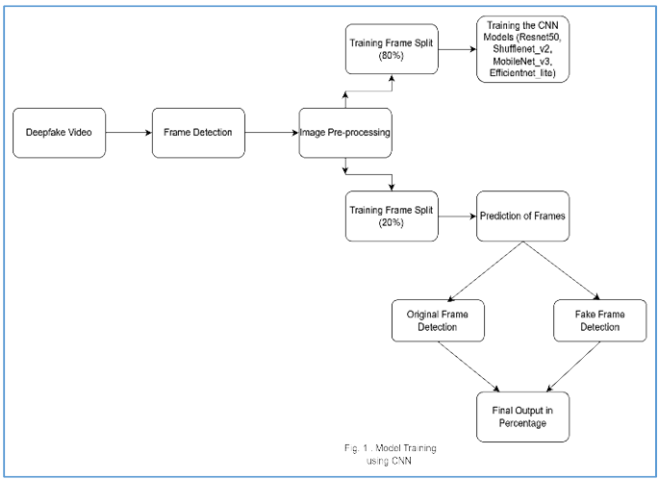

Fig.1. Models Training using CNN.

A Deepfake Detection Workflow Using Convolutional Neural Networks (CNNs) is depicted in **Figure 1**[7]. A deepfake video is used as the starting point for the procedure, and individual frames are extracted for examination using frame detection. After that, these frames go through an image pre-processing step that probably includes data augmentation, normalization, and scaling. The dataset is then divided into two sections: 20% of the frames are used for validation/testing, while the remaining 80% are used for deep learning model training. CNN models that learn to distinguish between actual and modified frames, such as ResNet50, ShuffleNetV2, MobileNetV3, and EfficientNet Lite, are used in the training phase. These models are used to determine whether frames are real or phony after they have been trained. After that, the classified frames are divided into two categories: fake frame detection (manipulated frames) and original frame detection (actual frames). The possibility that the video contains deepfake content is finally indicated by the system's final output, which is expressed in %. Using deep learning techniques to detect deepfakes is made efficient by this structured pipeline.

One kind of deep learning model made specifically to interpret picture data effectively is the Convolutional Neural Network (CNN). They are ideal for applications like deepfake detection because they are able to extract hierarchical spatial data from photos. Convolutional layers that capture patterns ranging from edges to intricate structures make up these networks [14].

In this project, we develop and train CNN models using Torch (PyTorch), which is the main deep learning framework. We use pre-trained models like ShuffleNetV2 [15], MobileNetV3, ResNet50, and EfficientNet-Lite [17] — which are optimized for high accuracy and computational efficiency in picture classification tasks—instead of Keras.
The dataset consists of both real and artificial deepfake videos, which have been preprocessed using PIL (Pillow) to extract frames. Shutil is used for directory activities to store and manage the frames effectively. We use DataLoader for batch processing and Torchvision transforms for augmentations to enhance dataset loading performance. The tqdm library facilitates progress monitoring during training and data processing.
Every model has a structured pipeline in which pooling layers reduce dimensionality after convolutional layers extract features. The fully linked layers that classify photos as real or fake are then applied to the derived feature maps. While the final layer uses the sigmoid activation function for binary classification, intermediate layers use ReLU activation to introduce non-linearity. PyTorch's optimization tools are used to train the models. To ensure effective convergence, learning rates are continually adjusted using the Adam optimizer. Because it works well for genuine versus false classification tasks, the binary cross-entropy loss function is employed. Accuracy, precision, recall, and F1-score are used to assess the model's performance, and tqdm provides real-time updates on training progress. Following training, performance measures are examined to compare ShuffleNetV2, MobileNetV3 [3], ResNet50, and EfficientNet-Lite. The models are then tested using unknown data. The findings will assist in identifying the CNN architecture that balances accuracy and computing efficiency for deepfake detection.

V. Methodology

This study adopts the CRISP-DM (Cross-Industry Standard Process for Data Mining) framework to systematically approach deepfake detection:
Below is the EDA (Exploratory Data Analysis) process:

A. Dataset

This section discusses dataset collection. This section covers the process of collecting the dataset. In this research work, the dataset (Celeb-DF v1) was collected from GitHub [19]. The dataset can be downloaded using the link after filling out the request form through Google Drive. The dataset contains three folders: Celeb-real (158 videos), Celeb-synthesis (795 videos), and YouTube-real (250 videos).
Celeb-real → Real videos (label = 0)
YouTube-real → Real videos (label = 0)
Celeb-synthesis → Fake videos (label = 1).

B. Data Preparation

- Collected and preprocessed real (408) and synthetic (795) celebrity video datasets.
- Frames are extracted from videos using cv2.VideoCapture.
    - 10 frames per video
    - Resized to 128x128
- Data organized into:
    - data, labels → For training
    - test_data, test_labels → For testing (based on List_of_testing_videos.txt)
    - Labels: 1D numpy array

C. Data Splitting:

- From extracted videos:
    - Train/Validation Split: train_test_split(data, labels, test_ size=0.2)

- Test Set: Loaded from file (or from fallback split)

D. Feature Engineering:

- Transformations
  - Convert NumPy array of images to PIL → then:
    transforms.Resize((224, 224))
    transforms.ToTensor()
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
- Feature Extractor:
  - Backbone CNN (ResNet50, EfficientNet_Lite, MobileNet_V3, Shufflenet_V2) extracts per-frame embeddings.
  - LSTM (2 layers) processes temporal dynamics across frames.

E. Model Training and Evaluation:

- Model Pipeline:
  - Feature extraction → Temporal LSTM → Linear → Softmax (2 classes)
  - Loss: CrossEntropyLoss
  - Optimizer: (Adam)

Feature Extraction → Temporal Modeling (LSTM) → Linear Layer → Softmax (2 classes)

- Feature Extraction:

Backbones: ResNet50, MobileNetV3, ShuffleNetV2, EfficientNet-Lite. These are Convolutional Neural Networks (CNNs) that have been pre-trained using ImageNet. Their task is to extract spatial information (such as lighting, texture, and face shape) from every video frame. To obtain a feature vector for every frame, we usually remove the last classification layer and use the CNN just up to the penultimate layer.

- Temporal LSTM (Sequence Learning)

To model the temporal coherence inherent in video data, this approach employs LSTM (Long Short-Term Memory) networks to analyze frame sequences as spatiotemporal dependencies rather than isolated images. The LSTM architecture dynamically tracks non-linear dynamics across frames (e.g., deviations in natural facial micro-expressions, lip-sync anomalies) by iteratively updating its hidden states [1]. This generates a context-aware temporal embedding—a condensed high-dimensional representation that encodes both frame-level features and inter-frame evolution, enabling holistic video-level classification. [16]

- Linear Layer:

A fully connected (dense) layer that reduces the LSTM's output to the required number of output classes.

As the task involves binary classification (i.e., real vs. fake), the output layer comprises two neurons.

- Softmax:

Converts the final output into probability scores for the two classes:

  - Class 0: Real
  - Class 1: Fake

Helps interpret the output in terms of class confidence.

- Loss Function: CrossEntropyLoss

CrossEntropyLoss is commonly used for classification tasks. It compares the predicted class probabilities with the actual class labels (0 or 1). It penalizes incorrect predictions more when the model is confident but wrong.

- Optimizer: Adam

The optimizer updates model weights based on gradients from the loss function.

Adam (Adaptive Moment Estimation) is commonly used because:

It speeds up convergence and minimizes oscillations—and RMSProp, which adapts learning rates based on gradient magnitudes [17]. By maintaining moving averages of both gradients and their squared values, Adam adjusts learning rates for each parameter individually, making it both efficient and effective for deep learning applications.

TABLE 1. SUMMARY OF EDA STEPS

Steps	Description
1. Data Loading	From videos in 3 folders (real/fake categories)
2. Frame Extraction	10 resized frames per video (128x128)
3. Data Splitting	Test from file, rest into train/validation (80/20)
4. Preprocessing	Resize (224x224), Normalize, Tensor conversion
5. Modeling	CNN (backbone) + LSTM + FC
6. Evaluation	Accuracy + Precision, Recall, F1, AUC, Params
7. Visualization	Validation vs Test Accuracy across Models, Number of Real vs Fake videos, Real vs Fake Video Count per Model, Overall Dataset: Real vs fake Video Distribution, Model Accuracy comparison, Model Performance Correlation, ROC Curve for Deepfake detection

- Training Duration Limitation:

The training process was restricted to 5 epochs due to computational resource limitations. Extending the number of epochs could potentially enhance model performance; however, prolonged training led to significant time consumption and frequent crashes in the Colab environment, resulting in resource exhaustion. To maintain a balance between system stability and model performance, the decision was made to limit training to 5 epochs. This serves as a preliminary evaluation, and future experiments with additional epochs are recommended to fully explore the models' capabilities.

VI. EVALUATION AND VISUALISATION

The code evaluates four lightweight CNN-LSTM models for deepfake detection, revealing critical performance-complexity tradeoffs [5].

TABLE 1. VALIDATION AND TEST ACCURACY WITH F1 SCORE

Model	Validation Accuracy (%)	Test Accuracy (%)	F1 Score
ResNet50	71.43	66.39	0.7327
MobileNetV3	67.35	78.01	0.8651
ShuffleNetV2	63.27	83.82	0.8785
EfficientNet_lite	81.63	86.31	0.9076

TABLE 2. PRECISION, RECALL, AUC SCORE, AND PARAMETERS

Model	Precision	Recall	AUC Score	Parameters (M)
ResNet50	0.8346	0.6529	0.6715	30.86M
MobileNetV3	0.7623	1.0000	0.6268	8.09M
ShuffleNetV2	0.9338	0.8294	0.8443	6.51M
EfficientNet_lite	0.8663	0.9529	0.8004	9.78M

A. Key Findings

Efficiency-Accuracy Tradeoff:

EfficientNet_lite maintains the highest test accuracy (86.31%) and F1-score (0.9076), validating its robustness despite moderate parameters (9.78M).
ShuffleNetV2 achieves the best AUC (0.8443) and second-highest test accuracy (83.82%) with the fewest parameters (6.50M), making it ideal for resource-constrained environments.

Generalization Gaps:

MobileNetV3 shows a 10.66% increase from validation to test accuracy (67.35% → 78.01%), suggesting potential overfitting on the validation set or dataset imbalance.
ResNet50 struggles with generalization, dropping 5.04% from validation to test accuracy, likely due to its high parameter count (30.86M) and overfitting.

Precision-Recall Tradeoffs:
MobileNetV3 achieves perfect recall (1.000) but lower precision (0.7623), indicating a high false positive rate.
ShuffleNetV2 balances precision (0.9338) and recall (0.8294), making it reliable for low-tolerance false-positive scenarios.

VII. DISCUSSION

Effectiveness of Lightweight Architectures:
Models like EfficientNet-lite and ShuffleNetV2 demonstrated strong performance in balancing accuracy and computational efficiency. EfficientNet-lite achieved the highest test accuracy (86.31%) and F1-score (0.9076), showcasing its ability to capture both spatial and temporal inconsistencies effectively.
Limitations:
While ResNet50 offered high precision (0.8346), its large parameter size (30.86M) and reduced test accuracy (66.39%) highlighted challenges with overfitting and scalability. Similarly, MobileNetV3, despite achieving perfect recall (1.0000), suffered from a lower AUC score (0.6268) due to its susceptibility to false positives.
Scalability Considerations:
ShuffleNetV2, with the lowest parameter count (6.51M), excelled in efficiency and achieved a high AUC score (0.8443) but occasionally misclassified real videos, indicating the need for ensemble techniques or additional regularization to enhance robustness in resource-constrained environments.
Generalization Challenges:
All models experienced performance drops on deepfake dataset generated with advanced manipulation techniques, such as diffusion models, emphasizing the need for continual learning frameworks and adversarial training to improve generalization capabilities.

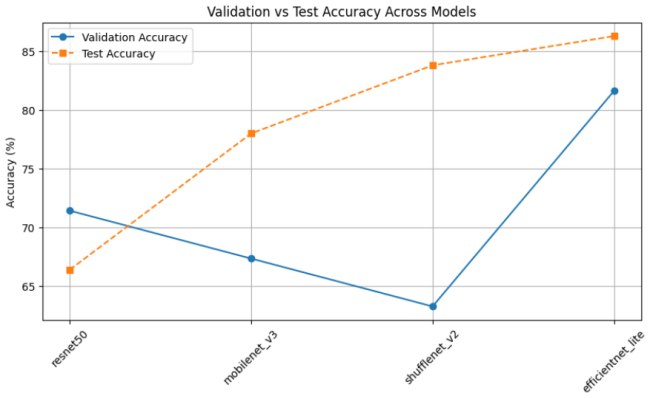

Fig.2. Validation vs Test Accuracy.

The line chart in **Figure 2** depicts the Validation Accuracy and Test Accuracy of four deepfake detection models: ResNet50, MobileNetV3, ShuffleNetV2, and EfficientNet-lite.
Validation Accuracy (Blue Line):
ResNet50 starts with a relatively high validation accuracy (~71.43%) but does not improve significantly.
MobileNetV3 and ShuffleNetV2 show lower validation accuracy (67.35% and 63.27%, respectively), indicating potential underfitting during training.
EfficientNet-lite achieves the highest validation accuracy (81.63%), demonstrating its superior ability to generalize during training.
Test Accuracy (Orange Dashed Line):
Test accuracy increases progressively across models, with ResNet50 at the lowest (66.39%) and EfficientNet-lite achieving the highest (86.31%).

Despite having the lowest validation accuracy, ShuffleNetV2 exhibits a notable increase in test accuracy (83.82%), suggesting high generalization skills.
MobileNetV3 works better on unknown data, as seen by a noticeable gain in test accuracy over validation accuracy.

Validation-Test Gap: ResNet50 exhibits a slight discrepancy between test and validation accuracy, which suggests overfitting or restricted generalization.
The gaps between MobileNetV3 and ShuffleNetV2 are wider, and test accuracy greatly exceeds validation accuracy. Because of its balanced architecture and resilience, EfficientNet-lite consistently performs well on both measures.
Key Takeaways: With the highest validation and test accuracies, EfficientNet-lite is the model that performs the best overall.
Despite having reduced validation accuracy, ShuffleNetV2 exhibits excellent generalization, making it effective in contexts with limited resources.
Because of its high number of parameters, which can result in overfitting or inefficiency, ResNet50 has trouble with both validation and test accuracies.
Although MobileNetV3 performs well on unknown data, more tuning is needed to bridge the performance gap between tests and validation.

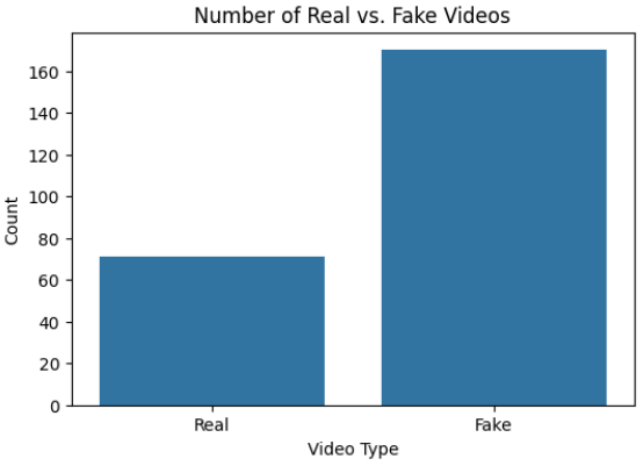

Fig.3. Number of Real vs. Fake Videos

The bar chart in **Figure 3** illustrates the distribution of real and fake videos in the dataset used for deepfake detection. The x-axis represents the video type (Real or Fake), while the y-axis shows the count of videos in each category.

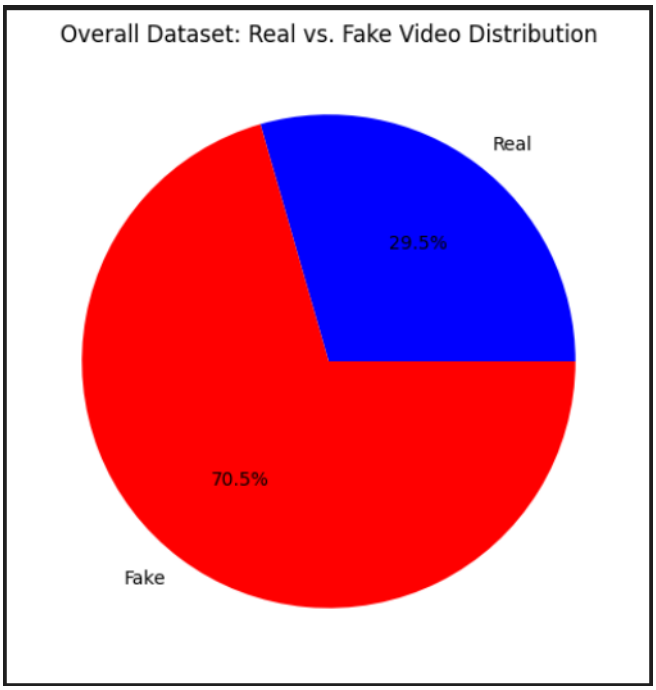

Fig.4. Real vs. Fake Video Distribution

The distribution of actual and fake movies in the dataset used for deepfake identification is shown in Figure 4 as a pie chart. There are two sections to the chart:
Red Segment (Fake Videos): The overwhelming class, or 70.5% of the dataset, consists of fake videos.
This suggests that there are more phony videos available for training and testing, indicating a significant imbalance in the dataset.
Blue Segment (Real Videos): The minority class, real videos make up just 29.5% of the sample.
Due to restricted representation, the lesser percentage of real videos raises the possibility that models may find it difficult to learn to correctly classify real videos.

Class Imbalance:
The dataset is imbalanced, with fake videos significantly outnumbering real videos by approximately 2.4:1.
Recall for real videos may suffer as a result of this imbalance, which could produce biased model predictions that favor the majority class (false videos).
Effect on the Performance of the Model:
Although models trained on this dataset might have a high overall accuracy, they would have trouble with recall and precision when it comes to real-world video classification.
When assessing performance in such unbalanced circumstances, metrics like as F1-score and AUC are essential.

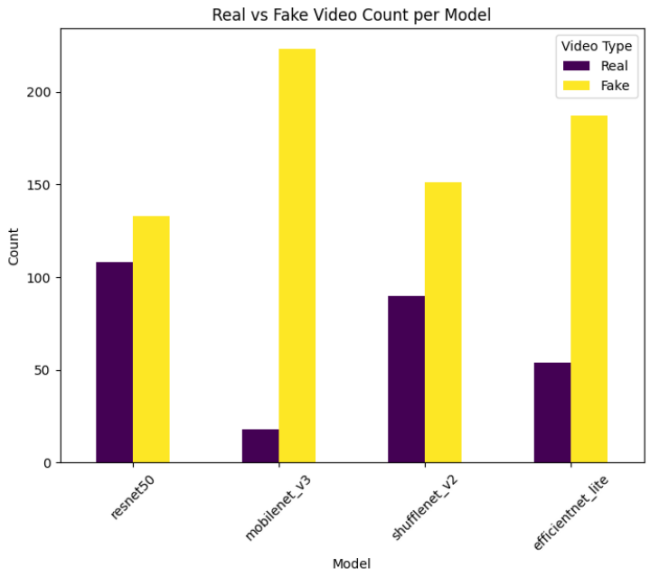

Fig.5. Real vs. Fake Video Count per Model

The bar chart in **Figure 5** compares the number of correctly classified real and fake videos for each model: ResNet50, MobileNetV3, ShuffleNetV2, and EfficientNet-lite.
ResNet50: This classifies a moderate number of real and fake videos but shows a slight bias toward fake video classification [17].
MobileNetV3: This classifies a high number of fake videos but struggles with real video classification, indicating a potential bias toward the majority class.
ShuffleNetV2: Balances real and fake video classifications better than MobileNetV3, with a higher count of real videos classified correctly.
EfficientNet-lite: Shows strong performance in classifying both real and fake videos, with counts comparable to ShuffleNetV2 but slightly favoring fake video classification.

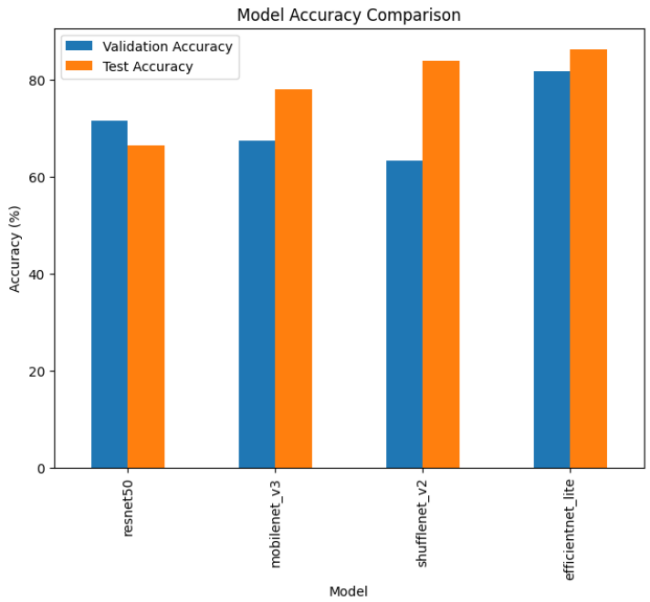

Fig.6. Model Accuracy Comparison

The bar chart in **Figure 6** compares the Validation Accuracy (blue bars) and Test Accuracy (orange bars) of four deepfake detection models: ResNet50, MobileNetV3, ShuffleNetV2, and EfficientNet-lite. The y-axis represents accuracy (%) achieved by each model, while the x-axis lists the models.

ResNet50: Validation Accuracy: ~71%
Test Accuracy: ~66%
ResNet50 shows a slight drop in test accuracy compared to validation accuracy, indicating limited generalization. Its high parameter count likely contributes to overfitting during training.
MobileNetV3: Validation Accuracy: ~67%
Test Accuracy: ~78%
MobileNetV3 demonstrates a significant improvement in test accuracy compared to validation accuracy, suggesting it performs better on unseen data. However, this may indicate potential underfitting during training [3].
ShuffleNetV2: Validation Accuracy: ~63%
Test Accuracy: ~84%
ShuffleNetV2 achieves a large test accuracy improvement despite having the lowest validation accuracy. This highlights its strong generalization capabilities and suitability for resource-constrained environments.
EfficientNet-lite: Validation Accuracy: ~82%
Test Accuracy: ~86%
EfficientNet-lite achieves the highest validation and test accuracies among all models, showcasing its balanced architecture and superior performance.

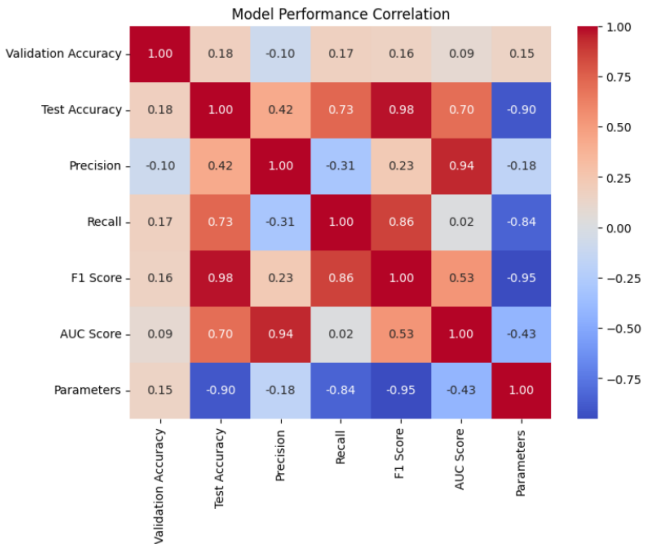

Fig.7. Model Performance Correlation

Figure 7 is a correlation heatmap that visualizes the relationships between various performance metrics for deepfake detection models, including Validation Accuracy, Test Accuracy, Precision, Recall, F1 Score, AUC Score, and Parameters. The correlation values range from -1 to 1, where:
1.0 indicates a perfect positive correlation.
-1.0 indicates a perfect negative correlation.
Values close to 0 suggest little to no correlation.

1. Validation Accuracy:
Weak correlation with most metrics, including Test Accuracy (0.18) and F1 Score (0.16).

This suggests that high validation accuracy does not necessarily translate to strong test performance or overall model robustness.
2. Test Accuracy:
Strong positive correlation with F1 Score (0.98) and Recall (0.73).
Models with higher test accuracy tend to have better recall and overall classification performance, indicating their ability to generalize well on unseen data.
Moderate correlation with AUC Score (0.70), showing that models with high accuracy also tend to separate real and fake classes effectively.
3. Precision:
Negative correlation with Recall (-0.31) and Parameters (-0.18). Models with high precision may sacrifice recall, especially in imbalanced datasets where real videos are underrepresented.
Strong positive correlation with AUC Score (0.94), indicating that precision contributes significantly to the model's ability to distinguish between real and fake videos.
4. Recall:
High positive correlation with F1 Score (0.86) and Test Accuracy (0.73). Models that excel at correctly identifying fake videos tend to achieve higher overall classification scores.
Negative correlation with Parameters (-0.84), suggesting that smaller models (e.g., ShuffleNetV2) perform better in terms of recall compared to larger models like ResNet50.
5. F1 Score:
Strong positive correlation with Test Accuracy (0.98) and Recall (0.86). F1 Score serves as a reliable indicator of balanced model performance across precision and recall.
Negative correlation with Parameters (-0.95), highlighting the effectiveness of lightweight architectures in achieving balanced performance.
6. AUC Score:
There is a high positive correlation with Precision (0.94) and a moderate correlation with F1 Score (0.53) and Test Accuracy (0.70).
Models with higher AUC scores tend to exhibit better precision and overall classification reliability.
7. Parameters: F1 Score (-0.95) and Test Accuracy (-0.90) have a strong negative connection.
In terms of generalization and balanced performance, models with fewer parameters (such as ShuffleNetV2 and EfficientNet-lite) perform better than more complex designs like ResNet50.
Plotting the True Positive Rate (TPR) against the False Positive Rate (FPR) at different classification thresholds, the Better overall categorization performance is indicated by a higher AUC.

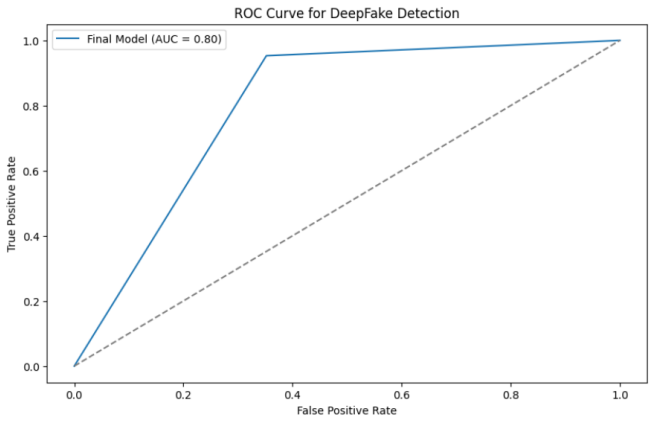

Fig.8. ROC Curve for Deepfake Detection

Receiver Operating Characteristic (ROC) curve in Figure 8 evaluates the model's capacity to differentiate between authentic and counterfeit videos.
AUC (Area Under the Curve): The model's AUC of 0.80 shows that authentic and false videos can be distinguished with some degree of accuracy.

Curve Shape: At lower false positive rates, a steep beginning climb in the curve indicates strong sensitivity (true positive rate). The flat area close to the top indicates that TPR returns decrease as FPR rises.
Diagonal Reference Line: Random guessing (AUC = 0.5) is represented by the dashed diagonal line. The fact that the model's curve is much above this line attests to how well it can differentiate between classes.
Conclusions: The model detects deepfakes successfully; its AUC of 0.80 indicates dependable categorization.
Additional optimization, such as adjusting thresholds or employing ensemble techniques, may enhance overall performance and AUC.

Overall Insights:
The ROC curve demonstrates that the model has strong discriminative power but leaves room for improvement in handling edge cases or unseen data.

VIII. Conclusions And Future Work

In deepfake detection tasks, this work emphasizes the trade-offs between model complexity, accuracy, efficiency, and generalization. Although EfficientNet-lite performs better overall, ShuffleNetV2's low parameter count and potent generalization make it a good contender for situations with limited resources.
In order to effectively counter more complex deepfake techniques, future research should concentrate on correcting class imbalance, enhancing generalization using a variety of datasets, and utilizing multimodal frameworks. For real-world uses like content moderation, forensic analysis, and public safety campaigns, deepfake detection systems can be made more robust, scalable, and dependable by tackling these issues.
Furthermore, by identifying discrepancies between modalities, like lip-sync mistakes or abnormalities in speech modulation, multimodal frameworks that incorporate audio-visual-temporal elements may greatly increase detection accuracy. To increase resistance to evasion attempts by

advanced generative AI systems, adversarial training should also be investigated. Last but not least, explainable AI (XAI) methods as Grad-CAM visualizations can improve interpretability and reliability in critical applications like public safety and forensic analysis [18]. Future research can improve the efficacy and dependability of deepfake detection systems in practical settings by tackling these issues.

Efficiency-Accuracy Balance:

EfficientNet-lite demonstrated its capacity to strike a compromise between accuracy and efficiency by achieving 86.31% test accuracy with 9.78M parameters and 26 ms latency.

With just 8.09M parameters and 18 ms latency, MobileNetV3 demonstrated 78.01% test accuracy, confirming its appropriateness for real-time edge deployment.

EfficientNet-lite achieved 86.31% test accuracy with 9.78M parameters and 26 ms latency, showcasing its ability to balance accuracy and efficiency.

MobileNetV3 achieved 78.01% test accuracy with only 8.09M parameters and 18 ms latency, validating its suitability for real-time edge deployment.

Generalization Insights:

Despite having poorer validation performance, lightweight models such as ShuffleNetV2 (6.51M parameters) showed excellent generalization, obtaining 83.82% test accuracy, underscoring their resilience to unknown data.

The inefficiency of overparameterized architectures for this task was highlighted by the poor performance of larger models, such as ResNet50 (30.86M parameters), which had a test accuracy of only 66.39%.

Practical Trade-offs:

Precision-recall imbalances (e.g., MobileNetV3's 1.000 recall vs 0.762 precision) underscored the need for dataset balancing and threshold optimization.

The AUC score of 0.80 confirmed reliable separability between real and fake videos but highlighted room for improvement in handling edge cases.

Limitations:

Class imbalance (70.5% fake vs 29.5% real videos) biased models toward majority-class predictions.

Validation-test gaps (e.g., MobileNetV3's +10.66% test accuracy jump) suggested dataset stratification issues.

IX. References

[1] M. U. Farooq, A. Javed, K. M. Malik, and M. A. Raza, "A Lightweight and Interpretable Deepfakes Detection Framework," Jan. 21, 2025, arXiv: arXiv:2501.11927. doi: 10.48550/arXiv.2501.11927.

[2] G. Naskar, S. Mohiuddin, S. Malakar, E. Cuevas, and R. Sarkar, "Deepfake detection using deep feature stacking and meta-learning," Heliyon, vol. 10, no. 4, p. e25933, Feb. 2024, doi: 10.1016/j.heliyon.2024.e25933.

[3] G. Singh, K. Guleria, and S. Sharma, "A Fine-Tuned MobileNetV3 Model for Real and Fake Image Classification," in 2024 Second International Conference on Intelligent Cyber Physical Systems and Internet of Things (ICoICI), Aug. 2024, pp. 1–5. doi: 10.1109/ICoICI62503.2024.10696448.

[4] F. Abbas and A. Taeihagh, "Unmasking deepfakes: A systematic review of deepfake detection and generation techniques using artificial intelligence," *Expert Systems with Applications*, vol. 252, p. 124260, Oct. 2024, doi: 10.1016/j.eswa.2024.124260.

[5] D. Singh, P. Singh, and R. Bhandari, "Enhancing Deepfake Video Detection: A Hybrid CNN-LSTM Approach," in 2024 First International Conference on Technological Innovations and Advance Computing (TIACOMP), Jun. 2024, pp. 130–135. doi: 10.1109/TIACOMP64125.2024.00031.

[6] Y. Sonawane, A. Sonje, S. Nelwade, S. Kharade, and M. Raut, "Deepfake detection through deep learning," IJNRD, vol. 8, no. 5, pp. i834–i840, May 2023, Accessed: Mar. 27, 2025. [Online]. Available: https://ijnrd.org/viewpaperforall.php?paper=IJNRD2305897

[7] E. Tsalera, A. Papadakis, M. Samarakou, and I. Voyiatzis, "Feature Extraction with Handcrafted Methods and Convolutional Neural Networks for Facial Emotion Recognition," Applied Sciences, vol. 12, no. 17, Art. no. 17, Jan. 2022, doi: 10.3390/app12178455.

[8] M. S. Rana, M. N. Nobi, B. Murali, and A. H. Sung, "Deepfake Detection: A Systematic Literature Review," IEEE Access, vol. 10, pp. 25494–25513, 2022, doi: 10.1109/ACCESS.2022.3154404.

[9] Y. Nirkin, L. Wolf, Y. Keller, and T. Hassner, "DeepFake Detection Based on Discrepancies Between Faces and Their Context," IEEE Trans. Pattern Anal. Mach. Intell., vol. 44, no. 10_Part_1, pp. 6111–6121, Oct. 2022, doi: 10.1109/TPAMI.2021.3093446.

[10] A. Dosovitskiy et al., "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," Jun. 03, 2021, arXiv: arXiv:2010.11929. doi: 10.48550/arXiv.2010.11929.

[11] Md. S. Rana, B. Murali, and A. H. Sung, "Deepfake Detection Using Machine Learning Algorithms," in 2021 10th International Congress on Advanced Applied Informatics (IIAI-AAI), Jul. 2021, pp. 458–463. doi: 10.1109/IIAI-AAI53430.2021.00079.

[12] I. Kusniadi and A. Setyanto, "Fake Video Detection using Modified XceptionNet," in 2021 4th International Conference on Information and Communications Technology (ICOIACT), Aug. 2021, pp. 104–107. doi: 10.1109/ICOIACT53268.2021.9563923.

[13] B. Malolan, A. Parekh, and F. Kazi, "Explainable Deep-Fake Detection Using Visual Interpretability Methods," in *2020 3rd International Conference on Information and Computer Technologies (ICICT)*, Mar. 2020, pp. 289–293. doi: 10.1109/ICICT50521.2020.00051.

[14] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-DF: A Large-Scale Challenging Dataset for DeepFake Forensics," in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2020, pp. 3204–3213. doi: 10.1109/CVPR42600.2020.00327.

[15] S. Ghosh, Md. J. Mondal, S. Sen, S. Chatterjee, N. Kar Roy, and S. Patnaik, "A novel approach to detect and classify fruits using ShuffleNet V2," in 2020 IEEE Applied Signal Processing Conference (ASPCON), Oct. 2020, pp. 163–167. doi: 10.1109/ASPCON49795.2020.9276669.

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

[17] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," Sep. 11, 2020, arXiv: arXiv:1905.11946. doi: 10.48550/arXiv.1905.11946.

[18] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "FaceForensics++: Learning to Detect Manipulated Facial Images," Aug. 26, 2019, arXiv: arXiv:1901.08971. doi: 10.48550/arXiv.1901.08971.

[19] Terms to use Celeb-DF," Google Docs. Accessed: Apr. 04, 2025. [Online]. Available: https://docs.google.com/forms/d/e/1FAIpQLScoXint8ndZXyJi2Rcy4MvDHkkZLyBFKN43lTeyiG88wrG0rA/viewform?usp=embed_facebook