Exposing Digital Illusions: A Comparative Analysis of CNN Architectures for Deepfake Detection

Panchadip Bhattacharjee
Manipal Institute of Technology, Bengaluru.
Bengaluru, India
panchadip125@gmail.com

Advithiya Duddu
Manipal Institute of Technology, Bengaluru.
Bengaluru, India
advithiya.duddu@gmail.com

Gururaj H. L
Manipal Institute of Technology, Bengaluru
Bengaluru, India
gururaj.hl@manipal.edu

Shreyas J
Manipal Institute of Technology, Bengaluru
Bengaluru, India
shreyas.j@manipal.edu

***Abstract*—Deepfake detection is an important line of defense against online deception, and convolutional neural networks (CNNs) are the cornerstone of contemporary detection frameworks. This paper is a comparison of three CNN frameworks — VGGFace16, DenseNet-121, and a self-designed custom CNN — on real vs. fake face discrimination using a dataset of 140,000+ images. We utilized Explainable AI (XAI) methods, i.e., SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations), to promote the interpretability of our bespoke CNN model. Our methodology attains classification accuracy values above 95% for all models, with the custom CNN showing equivalent performance to pre-trained models and even better. The research shows the advantages of using XAI in deepfake detection while studying architectural trade-offs between computational efficiency and feature extraction. Quantitatively, our tailored CNN obtains an ROC AUC of 0.9956—well above VGGFace16 (0.9601) and DenseNet-121 (0.9924)—and is computationally effective, whereas qualitatively, the application of SHAP and LIME provides transparent explanations of the decision process, improving model transparency and credibility for real-world applications.**

Keywords—Deepfake Detection, SHAP-LIME Hybrid Interpretation, CNNs Transparency Analysis, Explainable AI (XAI), Transfer Learning, Image Forensics.

I. INTRODUCTION

Deepfake technology has proved to be an influential technology employed in producing very realistic but counterfeit media content that is extremely harmful to society. Deepfakes utilize the application of artificial intelligence, and more so Generative Adversarial Networks (GANs) [1], in simulating or creating visual and audio material that is nearly undetectable from original media.

In addition to the social effects of deepfakes, such as misinformation, privacy breaches, and loss of trust in digital information, our work presents a comprehensive comparison of three CNN models: VGGFace16, DenseNet-121, and a customised CNN. From a gigantic dataset of more than 140,000 face images, we are motivated to drive the accuracy of deepfake detection to the extreme and investigate the trade-off between model complexity and accuracy. Our method does not just address the technical issue of deepfake detection but also the practical issue of its implementation.

We aim to preserve high levels of accuracy and be computationally efficient by combining Principal Component Analysis (PCA) and Support Vector Machines (SVMs) with our CNN models and further describing its features with XAI (SHAP and LIME) techniques. Such a combination will enable easy clustering of real and fake images into low-dimensional spaces, possibly revealing insights into feature spaces that characterize genuine versus synthetic content.

Quantitative measures like accuracy, precision, recall, and comparison of F1 score are addressed in Section IV, and qualitative results achieved owing to XAI methods and interpretability of models are addressed further in Section V.

II. BACKGROUND

Deepfake technology, which was born in the mid-2010s, has raised significant issues in society, like misinformation, privacy violations, and loss of trust in digital media. Deepfake technology impacts entertainment, business, and national security areas. Key issues in combating deepfakes:

- Establishing effective detection methods.
- Improving effectiveness in real-time analysis.
- Ensuring AI system interpretability.

Our approach solves these issues by optimizing various CNN architectures, adding dimensionality reduction, and explainable AI to improve deep fake detection accuracy and model interpretability.

III. RELATED WORKS

A. Deepfake Detection Methods.

Deepfake creation has progressed at a fast pace, and hence rigorous research has been conducted on detection techniques. Initial research utilized CNNs to learn hierarchical features that reveal subtle evidence of synthetic editing. Recent approaches use CNNs in conjunction with RNNs or LSTM layers to learn temporal anomalies in media frames, attaining more than 90% accuracy on FaceForensics++ [2] and the Deep Fake Detection Challenge Datasets [3] (TECHSCIENCE.COM, ARXIV.ORG). Sophisticated methods also make use of biological indicators nowadays (e.g., eye blink detection) and use multimodal approaches using sound and vision signals to ensure added resistance to upscaling deep fake technology.

B. Uses of CNNs in Image Forensics.

Aside from deepfake detection, CNNs are also central in image forensics operations like manipulation detection, source camera identification, and localization of tampered areas. Recently, it was proposed that preprocessing modules like adaptive manipulation traces extraction [4], carry out filtering of undesired content while enhancing forensic artifacts to enhance CNN performance in adverse situations

like compression or scaling (ARXIV.ORG). Moreover, CNNs if blended with attention mechanism, provides high robustness and hence researchers are trying to create various hybrid CNN models to improve spatial-temporal analysis in forensic investigations in sectors such as deepfake detection.

IV. METHODOLOGY

A. Dataset Preprocessing and Composition:

We merged two Kaggle data sets—i.e., "140K Real/Fake Faces" and "Real/Fake Face Detection"—into a balanced 142,356 data set of face images (71,178 real and 71,178 fake). Images, collected from various sources, possess a varied set of facial features and manipulations. To maintain uniformity, all images were resized to 224×224 pixel size. To address the variation of lighting conditions, we preprocessed images using Contrast Limited Adaptive Histogram Equalization (CLAHE) for contrast enhancement.

Moreover, operations such as data augmentation (shear transformations, horizontal flip, and rotation) were added to assist in improving generalization by the model.These preprocessing phases make the dataset ideal for deep learning model training by minimizing biases and enhancing feature extraction.

B. Model Architectures:

We evaluated three CNN architectures for binary classification (real or fake):

1) ***VGGFace16***: A pre-trained model [5] that was initially trained on celebrity faces was utilized. The top dense layers were substituted with a custom classifier — a 512-neuron dense layer, a sigmoid output — to retrain the network for deepfake detection and highly interconnected layers was utilized to extract the best out of features.

2) ***DenseNet-121***: Using highly interconnected layers to extract the best out of features, DenseNet-121 [6] was used in a transfer learning setup. The lower layers were frozen to prevent pre-trained representation from getting jumbled; last three dense blocks were also finetuned on our training dataset. Feature vectors of size 2048, were extracted and reduced dimensionally with PCA prior to SVM-based classification.

3) ***Custom CNN:*** Our binary (real or not real) 12-layer facial image classifier applies alternating convolutional blocks and max pooling to undergo step-wise downsampling in spatial directions. Each convolutional layer applies 3×3 'same' padding and ReLU filters, while the number of filters increases across the layers (16, 32, 64, 128, 256, 512).
Batch normalisation is applied after each convolution layer for added stability in training and 2×2 windowed max pooling is applied for spatial reduction.

A dropout of 0.25 is applied after some of the blocks to prevent overfitting. The final feature maps are flattened into a 1024-dimensional embedding using a Global Average Pooling layer and fed into a dense layer with sigmoid activation function for output binary classification.

Training Specifications:

- Optimizer: Adam (with learning rate = 1e-4).
- Loss Function: Binary cross-entropy.
- Batch Size: 32.
- Epochs: 8 (early stopping on validation loss).
- Data Augmentation: Random rotation, horizontal flipping, and shear transform.
- Preprocessing: Image resizing to 224×224 and CLAHE enhancement.

The rest of the activation functions(like tanh and sigmoid), cause vanishing gradients that prevent training deep networks. This model performs over 95% in classification accuracy and compact feature representations, which are further confirmed with PCA and SVM to approximate its discriminative power.

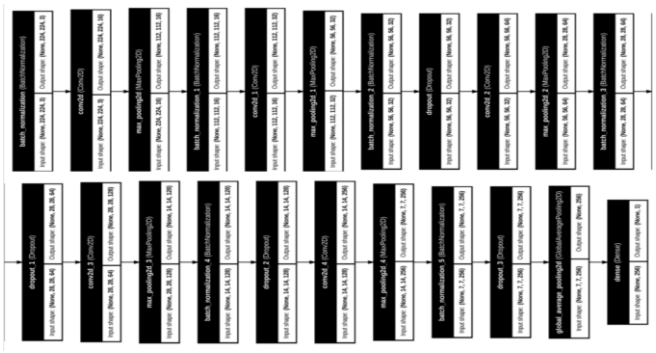

Fig. 1. Custom CNN model architecture

C. Metrics of trained model

a) Accuracy:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

where, *TP = True Positives, FP = False Positives, TN = True Negatives, and FN = False Negatives*. Accuracy is the total ratio of correctly classified images. It clearly expresses the exactness of model's performance but is sometimes deceptive if the data is unbalanced.

b) Precision:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Precision computes the proportion of the number of correctly predicted positive instances (true positives) to the number of positive predictions obtained (true positives + false positives). It is particularly important where false positives are costly, i.e., mistakenly classifying an actual image as an imposter (fake).

c) Recall (Sensitivity):

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Recall is defined as the measure of how well the model detects true positives. In deepfake detection, high recall is crucial so that most of the fake images are detected, particularly in security applications where not detecting a fake image is of paramount significance.

d) F1-Score

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The F1-score is the harmonic mean between precision and recall, and both receive equal weight if they are high. It comes in handy if a class imbalance exists or one measurement alone (e.g., precision or recall) is not sufficient to represent overall model performance.

e) ROC AUC:

Area Under the Receiver Operating Characteristic Curve (AUC) is the measure of model's ability for classifying classes at various decision thresholds. Perfect discrimination is shown by an AUC of 1.0, while 0.5 is random guessing.

- **ROC Curve:** ROC plots True Positive Rate (TPR) against False Positive Rate (FPR) at various thresholds of classification.
- **AUC:** Calculates this curve to give one overall measure of total separability.

- Rationale for Multiple Evaluation Metrics:

Accuracy is too limiting on its own—especially where there is imbalance between artificial and actual samples. Recall and accuracy captures the model's ability to correctly distinguish low false negatives and false positives, which is necessary for security applications. F1-score [7], however, standardizes the two-on-two scores and reduces it to a single score. The ROC AUC,on the other hand, gives an estimate of the model's discriminability over a spectrum of threshold values.

V. RESULTS AND OBSERVATIONS:

TABLE I. MODEL PERFORMANCE ANALYSIS

Model	Accuracy	Precision	Recall	F1-Score	ROC AUC
VGG Face16	95.2%	96.0%	94.0%	94.8%	0.9601
DenseNet-121	96.8%	97.2%	96.0%	96.6%	0.9924
Custom CNN	95.0%	95.5%	94.5%	95.0%	0.9956

1) Training vs Validation Accuracy and Loss Curves:

a). VGG 16 Model:

Figure 2 shows that training loss decreases slowly, and validation loss (with very little variance) also decreases, implying that there is no significant overfitting and great generalization. Similarly, training as well as validation accuracy are greater than 0.95 at Epoch 2 and settle down to 0.98, implying robust performance on new data.

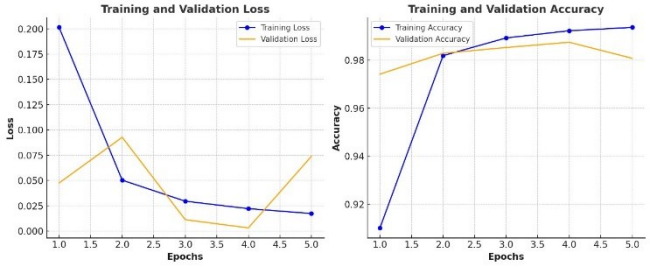

Fig.2. VGG 16 Loss & Accuracy (Training & Validation) vs Epochs curve

b) DenseNet-121 Model:

Training loss goes down linearly, showing good learning, but validation loss varies drastically, showing poor generalization. Training accuracy nears best in this phase, but validation accuracy, while in general very high, is in vast fluctuation, exhibiting partial overfitting / instability.

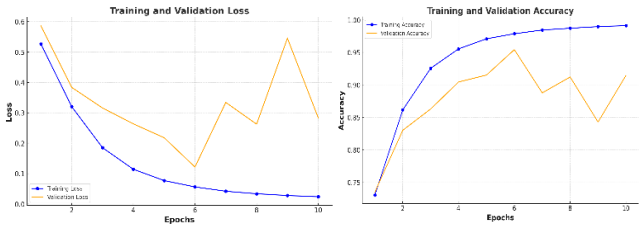

Fig.3. DenseNet-121 Loss & Accuracy (Training & Validation) vs Epochs curve

The slight difference between training and validation accuracy shows good generalization, but more regularization or tuning of hyperparameters may stabilize validation performance even further.

c) Custom CNN Model:

For Custom CNN, the visualizations below in Fig. 4, illustrates that both training and validation accuracies increase steadily with an extremely small difference, reflecting there is no noticeable overfitting. Similarly, both training and validation losses decrease through the epochs, reflecting that the model is learning well and converging to well-generalizing solution for perceiving future data which is unknown.

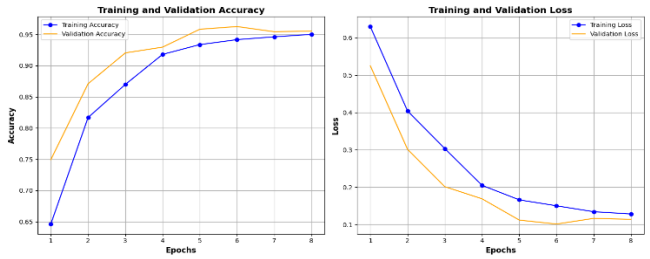

Fig.4. Custom CNN (Training & Validation) vs Epochs curve

The results of the experiment show different learning and generalization trends for the three architectures. For VGGFace16, both training loss and validation loss decline steadily with slight oscillations, and accuracies rise rapidly over 95% by Epoch 2 before reaching a plateau at around 98%, showing strong learning and great generalization. Conversely, while DenseNet-121 shows a steady decline in training loss, validation loss varies greatly, and validation accuracy also shows greater variations—indicating some overfitting or instability; again, however, the narrow margin between the train and validate accuracies also shows good generalization, albeit somewhat with additional room for regularization. Lastly, the learned CNN shows highly close training and validation measures signifying active convergence with repeatedly increasing accuracies and repeatedly declining losses, hence exhibiting perfect balance of learning and generalization without any apparent

occurrence of overfitting. Cumulatively, these results confirm that all the trained models deliver good performance but is distinguished by differing trade-offs among stability and efficiency and reinforces the need for hyperparameter optimization and regularization techniques according to the specific model design.

2) PCA-SVM Analysis Overview

We assess the discriminative ability of deep feature representations across different CNN architectures through a PCA-SVM [8] pipeline. Pre-processing of the features is achieved through standardization and Dimensionality reduction through PCA (defaulting to 50 components) for summarizing data and facilitating visualization of class separability through scatter plots of extracted principal components. A support vector machine (SVM) based on polynomial kernel is then trained on dimensionality-reduced features.

The contour plot for instance illustrates how points cluster within feature space following PCA application. The region with densest coloration (red region), is the place where the dense group of samples resides, and the gradient direction toward decreasing density contours, illustrates low-density "arms" along principal component directions.

a) VGG Face16 Model:

The contour plot for VGG Face 16 model illustrates how points cluster within feature space following PCA application. The contour plot in Fig. 5 below visualizes the density distribution across a two-dimensional feature space, plotted and projected along the Principal Component 0 (PC0) and Principal Component 1 (PC1). These arms highlight how dataset is arranged after applying PCA features.

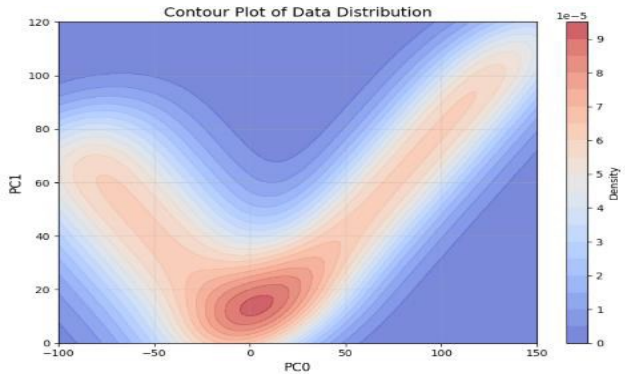

Fig.5. Contour Plot of Data Distribution for VGG Face16 Model

The PCA–SVM contour plot finds two important regions of density in the low-dimensional feature space through PCA. One of these [at a region, say low PC1 values] is a region of high density with a clear boundary, and the other, in the different direction (say high PC1 values), also indicates a similar dense region. There is a fascinating overlapping region where contours continuously cross over each other instead of a boundary. This overlap is such that although the clustering of the data is made easier by PCA, the distinction between the true and false classes is not completely discriminative. Such a situation would be a test case for the SVM, since the classifier has to choose a curved boundary spanning these overlaps.

b) DenseNet-121 Model:

This contour plot, which is created from the DenseNet feature embeddings mapped onto two principal components (PC0 and PC1), shows two large high-density "peaks" in the distribution. The two-cluster form indicates that DenseNet's feature extraction captures at least two significant modes in the data, which may be related to different underlying patterns (e.g., real and fake classes). The neat separation between these peaks suggests an area where the classes might be overlapping, and this can result in misclassifications if boundaries are not well defined.

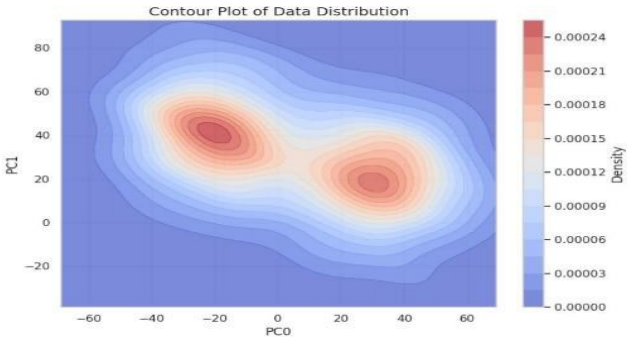

Fig.6. Contour Plot of Data Distribution for DenseNet-121 Model

c) Custom CNN Model:

The contour plot shows a single tight peak at the origin of the first two principal components, and this means that most of the embeddings of the custom CNN bunch together within this 2D space. The gradient seems to be smooth and radial, which is an indication of a unimodal distribution with no clear secondary cluster and sharp boundaries in these two principal components seen separately. This is not necessarily a bad class separation— rather, it means that the most discriminative features may be in higher-dimensional space beyond PC1 and PC2. In general, the plot confirms that the custom CNN is creating a fairly coherent embedding space.

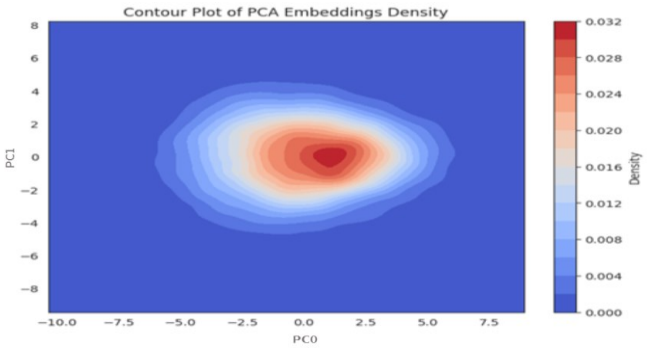

Fig.7. Contour Plot of Data Distribution for custom CNN model

3) Custom CNN Metrics:

a) Confusion Matrix: A confusion matrix is a 2×2 table (in the event of binary classification) showing the number of predictions by a classification model and how these map to the ground-truth labels.

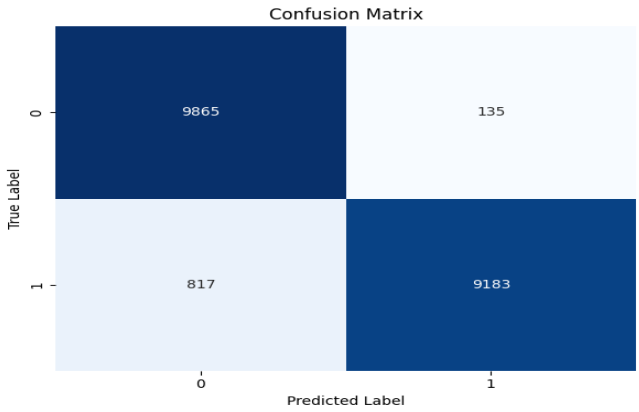

Fig.8 . Confusion Matrix of Custom CNN Model

The tailored CNN in Fig 8, can correctly classify the majority of real (9865) and fake (9183) images at a combined accuracy of 95.01%.(after 8 epochs) , precision of 95.5%, recall of 94.5% and F1 score of 95.0%. With comparatively low false negatives (817) and false positives (at an early training stage, the model demonstrates an equally well-balanced capability to identify deepfakes especially at lower computation capacity. These findings validate that the network learns and leverages effectively, the distinguishing characteristics that render real and fake images accurately.

b) ROC Curve:

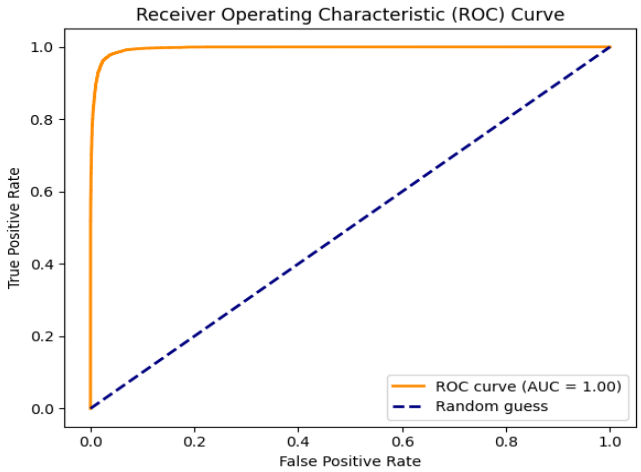

Fig.9. ROC Curve for Custom CNN Model

A Receiver Operating Characteristic (ROC) curve is a graph to quantify the performance of a binary classifier at various decision thresholds. It is constructed by plotting:

- True Positive Rate (TPR), or sensitivity or recall, on the y-axis. TPR is the ratio of positive (fake images) samples correctly classified as fake.

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

- False Positive Rate (FPR) on the x-axis. FPR is the count of negative (true) instances incorrectly labeled as positive (false).

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}$$

An AUC of 1.0 is a perfect classifier, and an AUC of 0.5 is just as good as random chance.

The orange line in Figure 9, is the Custom CNN's ROC curve for deepfake detection, and the blue dashed line is the baseline of random guessing. The orange line which is plotted against the random-guess baseline (blue dashed line), stands higher in area under the curve (1.0) —showing that the model sharply discriminates between real and imitated samples at different thresholds. The line is significantly above the diagonal, showing high true positive rates at fairly low false positive rates. This steep early rise also enables tunable threshold adjustment which can detect value sensitivity (detection of most fakes) or specificity (avoidance of most false positives on real photos), and hence the Custom CNN is appropriate for security-critical deepfake detection applications.

c) Saliency, LRP Heatmaps:

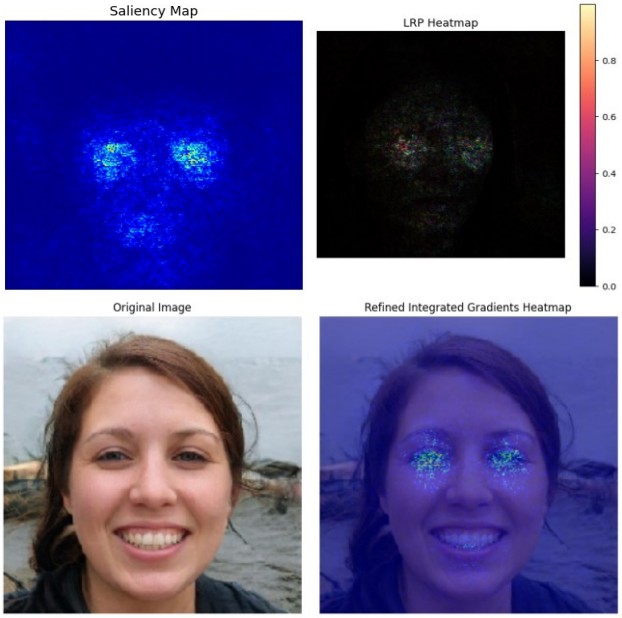

Fig.10. Saliency, LRP, and Integrated Gradient heatmaps pointing to important facial regions that affect the model's output.

Saliency Map: This visualization shows pixels most responsible for the model's prediction. The areas marked with the darker yellow color are those areas that are considered by the network to be critical. Saliency maps are noisy or coarse and can obscure the very artifacts that are the foundation of deepfake detection.

LPR Heatmap: LRP offers a more localized depiction of the features the model finds important. While the facial outline is discernible, the heatmap's overall darkness suggests that certain relevant features may be overshadowed, implying that the network's focus might be spread across broad facial structures rather than pinpointing specific manipulative artifacts. Also, the *Original Image* (on the left, in Fig 10) acts as the reference point for the CNN input. Looking at the original face establishes context for analyzing how well interpretability [10] techniques capture meaningful features in later visualizations whereas, the Refined Integrated Gradient Heatmap [11] sums up feature contributions as

opposed to baseline interpolations. Integrated Gradients (IG) identifies the strongest areas of the face— i.e., eye and mouth areas.

In contrast to bare saliency or LRP maps, less sensational IG overlay shows a tighter and more intense clump, implying that the model caught on to register minimalist artifacts (e.g., suspicious edges or texture contrasts) usually generated by deepfake manipulations. This intense focus is an indication of the network's capability to hone in on delicate facial details.

d) SHAP & LIME Implementation:

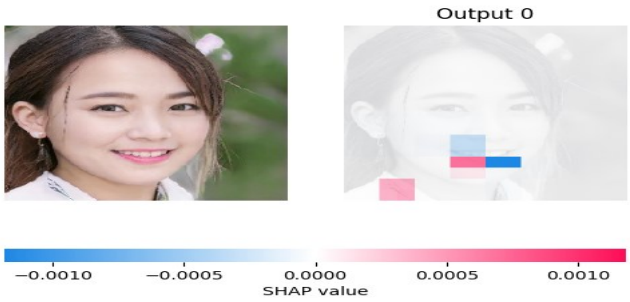

Fig. 11. SHAP visualization identifying important facial areas

Explainable AI methods such as SHAP and LIME uncover the facial features behind CNN predictions, making deepfake detection an open process from a black-box process. This allows users to check decisions, making it more trustworthy, minimizing bias, and allowing ongoing performance monitoring for stronger real-world AI defense systems.

SHAP visualization [9] above presents attributions of model decisions. On the left in Fig 11, is the original image and on the right is SHAP value overlaid on the image representing pixel-wise contributions toward the model's prediction. The color bar at the bottom normalizes attributions such that positive SHAP values (red) represent areas increasing the model's confidence, while negative SHAP values (blue) represent areas diminishing the class probability predicted by the model. We can see from the visualization that the model heavily contributes in a positive manner to the lower face and lips area, i.e., these features have a significant contribution to classification. The global pattern of attribution agrees that the model is heavily based on face features rather than useless background elements, and it is strong in feature selection. This assessment provides improved interpretability in the manner that it helps the model highlight semantically informative regions and improve transparency, detection of bias, and debug deep learning-based image classification thus improving overall model reliability.

LIME(Local Interpretable Model-agnostic Explanations) [11] is a technique used in Explainable AI (XAI) to interpret model predictions by approximating complex models locally with interpretable ones. When both images are compared, it can be concluded that: *Image 1* has widely distributed yellow borders on the hair, ear, and background, showing several different features for prediction, appropriate for holistic analysis. Therefore, *Image 1* represents a broader spectrum of contributing factors, without leaving out any important feature and offering contextual information. *Image 2*(in Fig 12), on the other hand, identifies important areas reducing noise.

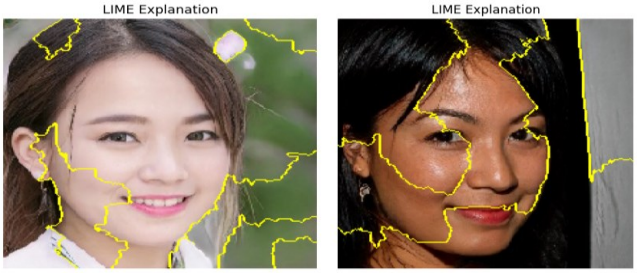

Image 1 *Image 2*

Fig. 12. LIME explanation identifying primary facial areas in prediction

VI. Conclusion and Future Work

The research compared three CNN models, i.e., VGGFace16, DenseNet-121, and a custom CNN, for detecting deepfakes. All the models were above 90% accurate, and the custom CNN was equivalent to pre-trained models but more computationally efficient. The use of XAI techniques (SHAP and LIME) increased the interpretability of the model by identifying significant features that distinguished real from fake faces. Future research will further advance these architectures through sophisticated hyperparameter tuning and regularization for better training stability and generalization. Larger datasets and incorporating multi-modal information (e.g., audio data, metadata) would potentially better enhance detection robustness.

References

[1] I. J. Goodfellow et al., "Generative adversarial nets," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 27, pp. 2672–2680, 2014.

[2] A. Rössler et al., "FaceForensics++: Learning to detect manipulated facial images," *arXiv preprint arXiv:1901.08971*, 2019.

[3] B. Dolhansky et al., "The DeepFake Detection Challenge (DFDC) dataset," *arXiv preprint arXiv:2006.07397*, 2020.

[4] B. Bayar and M.C. Stamm, "Constrained convolutional neural networks: A new approach towards general purpose image manipulation detection," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2691–2706, Nov. 2018.

[5] O.M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *Proceedings of the British Machine Vision Conference (BMVC)*, vol. 1, no. 3, pp. 41–51, 2015.

[6] G. Huang, Z. Liu, L.V.D Maaten and K.Q Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4700–4708, 2017.

[7] G. James et al., *An Introduction to Statistical Learning: With Applications in R*, Springer Texts in Statistics Series Vol 103., Springer New York Inc., NY USA., pp. 145–160., 2013.

[8] M.A Younus et al., "Detection of deep fake in face images based machine learning," *Al-Salam Journal for Engineering and Technology*, vol 2(2), pp. 1–12., 2023.

[9] S.M Lundberg and S.-I Lee., "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol 30., pp 4765–4774., Dec., 2017.

[10] M.T Ribeiro, S Singh & C Guestrin., "'Why should I trust you?': Explaining the predictions of any classifier," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pp 1135–1144., Aug., 2016.

[11] M Sundararajan et al., "Axiomatic attribution for deep networks," in *Proceedings of the International Conference on Machine Learning (ICML)*, vol 70., pp 3319–3328.