

# AI-Powered Deepfake Detection Using CNN and Vision Transformer Architectures

Md Sifatullah Sheikh<sup>1</sup>, Urmi Kirtonia<sup>1</sup>, Nuzath Tabassum Arthi<sup>1</sup>, Md Al-Imran<sup>1\*</sup>

<sup>1</sup>Department of Computer Science and Engineering, East West University,  
A/2 Aftabnagar, Dhaka 1212, Bangladesh

Emails: {mdsifatullahsheikh, urmikirtonia32, tabassumarthi2001}@gmail.com, al.imran@ewubd.edu

**Abstract**—The increasing use of artificial intelligence-generated deepfakes creates major challenges in maintaining digital authenticity. Four AI-based models, consisting of three CNNs and one Vision Transformer, were evaluated using large face image datasets. Data preprocessing and augmentation techniques improved model performance across different scenarios. VFDNET demonstrated superior accuracy with MobileNetV3, showing efficient performance, thereby demonstrating AI’s capabilities for dependable deepfake detection.

**Keywords**—Deepfake Detection, DFCNET, VFDNET, MobileNetV3, ResNet50

## I. INTRODUCTION

The development of deepfakes altered the landscape of digital media, whereby it enabled the creation of hyper-realistic content. There have been reported cases where creative applications were lauded, whereas the lack of unethical use has permeated. 2017 [1] saw widespread misuse of such technology with fake celebrity content and non-consensual altered porn, igniting grave concerns about AI and personal privacy issues.

A 2020 report from Sensity AI (formerly Deepttrace) [2] reported that 96% of online deepfake videos were pornographic and created without consent, which mainly targeted women before being distributed through adult websites and the dark web. The inappropriate use of deepfake technology leads to both psychological trauma and reputational harm while resulting in decreased public confidence in digital content. The use of deepfakes extends beyond privacy threats because they present significant dangers to democracy and security by supporting political manipulation along with identity theft and scams, and widespread disinformation. Fake media [3] spread without control endangers legal systems because it makes digital evidence validation more difficult to perform, according to Chesney and Citron.

To counter the growing threat presented by more realistic deepfakes, this study suggests a machine learning and deep learning-based method to recognize modified face photos. We specifically examine the efficiency of the Deepfake Convolutional Network (DFCNET), MobileNetV3, ResNet50, and Vision Fake Detection Network (VFDNET) in discriminating between real and modified information. This presents the key research question: **Which of these models performs the most**

**accurately and strongly in detecting deepfake images using various alteration techniques?** Our goal is to contribute to the development of accurate detection systems that will help restore public trust in digital media.

## II. LITERATURE REVIEW

Deepfake detection research is the main theme of this part, which focuses on the performance of models trained on specific datasets.

The dense CNN (D-CNN) developed by Patel et al. (2023) [4] obtained 97.2% accuracy throughout their real and fake image dataset. The D-CNN outperformed MesoNet variants by achieving 99.33% on GDWCT and 99.17% on StarGAN, while performance reduced to 94.67% for high-resolution images from StyleGAN2.

Vision Transformers (ViT) combined with 40,000 Kaggle images became the tool of choice for Ghita et al. (2024) [5] as they reached 89.91% accuracy. The approach showed potential but required extensive computational resources and achieved a moderate accuracy level without testing across different datasets.

DenseNet-121 and VGG16 models from Alkishri et al. (2023) [6] achieved 99% detection accuracy when they processed 140K Real and Fake Faces data with a DFT-based frequency analysis method. The detection system experienced reduced accuracy when handling low-resolution images due to insufficient frequency information.

The research by Zhang et al. (2022) [7] presents an ensemble learning technique that merges gradient and frequency and texture characteristics. Using a training dataset of CelebA and five GAN-based fakes, the model reached 97.04%

## III. METHODOLOGY

For deepfake detection, this study focuses on vision transformers and convolutional neural networks. Major preprocessing processes and feature analyses were carried out, as seen in Figure 1.

### A. Preprocessing Pipeline for Enhanced Model Generalization

Data sets require specific preprocessing steps such as normalizing pixel values [0, 1] and resizing images  $224 \times 224$  to maximize model performance. The preprocessing procedure converts RGB images to grayscale batches of sixteen. Real

\*Corresponding author: Md Al-Imran (al.imran@ewubd.edu)

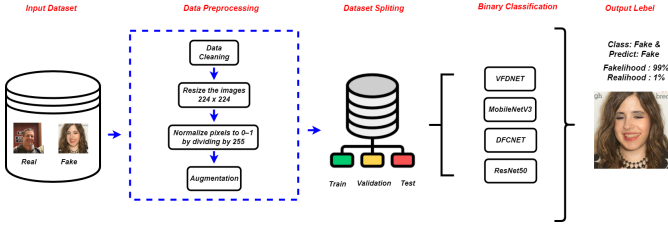


Fig. 1: Schematic Workflow of the Proposed Method

and fake data are classified as binary under labels 0 and 1. Data generalization improves through image augmentation techniques, including rotation, scaling, resizing, and flipping, which also decrease overfitting. Implementing histogram analysis guarantees consistent data distributions between training, validation, and test data sets.

### B. Fundamental CNN Layers

Convolutional Neural Networks (CNNs) are composed of essential layers that extract, reduce, and categorize input data. This section describes the basic layers that make up the foundation of CNN architectures.

1) *Pooling Layer*: The pooling layer reduces the size of the image while keeping the key features. The output is called a pooled feature map. The inclusion of these layers is important because not only do they help reduce overfitting, but they also reduce the cost of computation.

2) *Convolutional Layer*: Convolutional layer activations can be expressed as a tensor of size  $Height \times Width \times F_m$ , where  $F_m$  represents the number of feature maps. The algorithm divides [8] the input picture into  $Height \times Width$  sections and uses  $D$ -dimensional feature maps to identify visual patterns from each sector. Each filter performs a convolution on the input images to discover a particular feature and adds a bias term, as represented in Equation (1).

$$h^{k+1} = \sigma (W^k \otimes h^k + b^k) \quad (1)$$

The convolution operation [9] in this formula  $\otimes$  is denoted by  $h^k$  for input feature map  $k$  and by  $W^k$  for filter weight matrix and by  $b^k$  for bias term and by  $h^{k+1}$  for output feature map together with  $\sigma$  for the activation function.

3) *Fully Connected (FC) or Dense Layer*: A dense layer within a neural network is a layer where every neuron in the present layer is connected to every neuron in the previous layer, thus, every input node is connected to every output node. Figure 2 represents a network that has two FC layers, where input and output neurons are shown. The two layers are indicated as Fully Connected 1 as FC1 and Fully Connected 2 as FC2. Suppose that  $x$  is the output of FC1 where  $x \in \mathbb{R}^{N_1 \times 1}$ . Assume that  $W$  represents the weight matrix of FC2 where  $W \in \mathbb{R}^{N_1 \times N_2}$  and  $w_i$  is the  $i$ -th column vector of  $W$ . The weight vector of the corresponding neuron  $i$ -th in the FC2 layer is represented by each column  $w_i$ . Therefore, the output of FC2 [10] is given by  $W^T x$ .

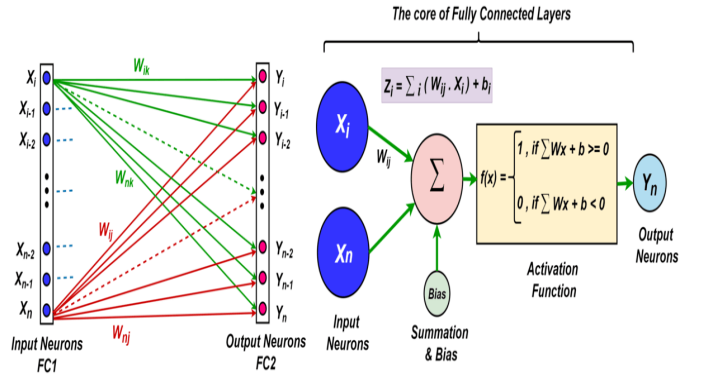


Fig. 2: Fully Connected (FC) layer with two layers.

4) *Flatten Layer*: The neural network uses a flattening layer to transform the output data produced by the previous network segment into a one-dimensional matrix. The flattening layer can filter individual neurons. The flattened layer decreases the number of weights that exist between the convolutional and fully connected layers.

### C. Commonly used Activation Functions

In neural networks, each neuron in the same layer has the same activation function. The model is trained by engaging gradient descent and backpropagating error signals through each neuron.

1) *Rectified Linear Unit* : A Convolutional Neural Network (CNN) implements ReLU activation functions to simplify operations within its neural network layers. The use of leaky ReLU activation functions [11] helps the network process negative values more efficiently, as depicted by Equation (2).

$$ReLU(z) = \max(0, z) \quad (2)$$

The function represents  $z$  as the neuron input. When  $z$  is above 0, the function delivers  $z$  back. The function generates an output of 0 whenever  $z$  remains equal to or lower than 0. The mathematical representation of the ReLU and Leaky ReLU functions uses the following equations (3) and (4):

$$ReLU(z) = \begin{cases} z, & \text{if } z > 0 \\ 0, & \text{if } z \leq 0 \end{cases} \quad (3)$$

$$Leaky-ReLU(z) = \begin{cases} z, & \text{when } z > 0 \\ 0, & \text{when } z \leq 0 \end{cases} \quad (4)$$

2) *Sigmoid Activation Function*: The S-shaped curve of the sigmoid activation function ranges between one and zero. It aids neural networks in learning from the data and is used for binary classification, but it is highly sensitive to input perturbations. Equation (5) and (6) denote that Sigmoid Activation function  $\mu(z)$  and Hyperbolic Tangent function  $\tau(z)$  can be expressed as:

$$\mu(z) = \frac{1}{1 + exp^{-z}} \quad (5)$$

$$\tau(z) = \frac{\exp^z - \exp^{-z}}{\exp^z + \exp^{-z}} \quad (6)$$

Where  $\exp^z$  means the exponential function, which grows swiftly as  $z$  increases, and  $\exp^{-z}$  means the same function but mirrored, meaning it decreases as  $z$  increases.

#### D. Framework for the detection of fake images

Four models, including three CNN-based and one transformer-based, were fine tuned to classify real and fake deepfake images.

1) **Deepfake Convolutional Network (DFCNET)**: The DFCNET model is an unknown function that uses various layers connected one after another. For each layer, we create a feature map  $z_i \in \mathbb{R}^{m_i \times n_i \times c_i}$ , which is obtained from the output of the previous layer  $z_{i-1}$ . The input image  $x$  serves as the first layer  $z_0$ , and the final layer generates the output image  $y$ .

Each input channel is convolved with a specific filter, summed pixel wise with a bias, and passed through a non linear function. Repeating this with various filters produces multiple output channels. The output  $a_i^j$  [12] of channel  $j$  is given in Equations (7) and (8):

$$a_i^j = \phi(f_{ij}(a_{i-1}) + \beta_{ij}) \quad (7)$$

$$f_{ij}(a_{i-1}) = \sum_{k=0}^{c_{i-1}} \mathcal{F}_{\kappa_{ijk}}(a_{i-1}^k) \quad (8)$$

Here,  $\phi$  is a nonlinear activation function.  $\beta_{ij} \in \mathbb{R}$  is the bias and  $f_{ij}$  convolves each channel of  $c_{i-1}$  with a filter and sums the results.

For incorporating nonlinearity, we built our bespoke CNN with three 2D convolutional layers and max-pooling with ReLU activation. Dropout was introduced after every convolutional layer as a safeguard against overfitting. Then, a deep layer that had 256 dimensions of ReLU-activated units was employed to represent the high level features. This layer utilized a sigmoid classification function to produce the binary classifier output. The training process involved the use of the Adam optimizer, which led to model evaluation based on binary cross-entropy loss and accuracy metrics.

2) **Vision Fake Detection Network (VFDNET)**: In image classification, the Vision Transformer (ViT), a pioneering transformer-based design, has performed similarly to or better than CNNs. Its ability to collect global contextual relationships through self-attention makes it ideal for tasks like bogus picture identification. In this paper, we also suggest a ViT-based model to solve the classification problem.

As illustrated in Figure 3, the proposed VFDNET model first divides the input image into fixed-size patches, linearly projecting each into tokens to form a sequence. Three data augmentation techniques—AutoAugment\_transform, RandAugment\_transform, and Auto\_RandAugment\_transform—were applied to enhance model robustness. All input images were resized to  $224 \times 224 \times 3$ . The augmented datasets were

organized using the ImageFolder class and loaded via DataLoader for training and evaluation. The Adam optimizer was employed due to its adaptive learning rate capability. The transformer encoder [13] design uses many blocks that consist of multi-headed self-attention (MSA) combined with a feedforward network (FFN). The feedforward network implements a two-layer multilayer perceptron with the hidden layer expanding at a rate of  $r$ , followed by one GELU non-linearity applied to the first linear layer. Every block within the transformer receives layer normalization (LN), which supports residual connections made after each block. The VFDNET input,  $h_0$ , and the  $k$ th block transformation process can be formally described in Equation (9):

$$\begin{cases} \mathbf{h}_0 = [\mathbf{h}_{\text{class}} \parallel \mathbf{h}_{\text{patch}}] + \mathbf{h}_{\text{position}} \\ \mathbf{q}_k = \mathbf{h}_{k-1} + \text{MSA}(\text{LN}(\mathbf{h}_{k-1})) \\ \mathbf{h}_k = \mathbf{q}_k + \text{FFN}(\text{LN}(\mathbf{q}_k)) \end{cases} \quad (9)$$

where  $\mathbf{h}_{\text{class}} \in \mathbb{R}^{1 \times C}$  and  $\mathbf{h}_{\text{patch}} \in \mathbb{R}^{N \times C}$  are the class and patch tokens, respectively, and  $\mathbf{h}_{\text{position}} \in \mathbb{R}^{(1+N) \times C}$  is the position embedding.  $C$  is the size of the matching embedding dimension, and  $N$  is the number of extracted patch tokens.

3) **MobileNetV3**: The tuning process of MobileNetV3 [14] for mobile phone CPUs results from hardware Network Architecture Search (NAS) in combination with the NetAdapt algorithm. MobileNetV3 consists of two implemented models, which are MobileNetV3 Large and MobileNetV3 Small. Each of these models was specifically designed to address either high or low resource usage needs. The models have been customized and used to accomplish both object detection and semantic segmentation tasks. MobileNetV3 enhances its architecture to deliver better performance using minimal computational resources.

4) **ResNet50**: ResNet50 is a deep neural network architecture that consists of 50 weight layers. ResNet50 is a CNN architecture model that normalizes the concepts of residual learning and skip connections. ResNet50 belongs to the residual network family. The ResNet50 contains 48 convolution layers, which include a single maxpool layer and an average pool layer. This model processes data through an initial sequence of layers, which include low-level feature extraction and ReLU activation through convolution operations and batch normalization, and max pooling.

## IV. EXPERIMENTAL SET

In this section, we divide the discussion into four parts. The first part is to use the DFCNET, ResNet50, VFDNET, and MobileNetV3 networks to train and test the 140K real and fake faces dataset.

### A. Dataset Description

The dataset containing deepfake images is accessible through Kaggle from Xhlulu, who uploaded it, and contains 140K real and fake faces. The dataset contains 70000 photos of authentic faces, which come from the Flickr dataset that

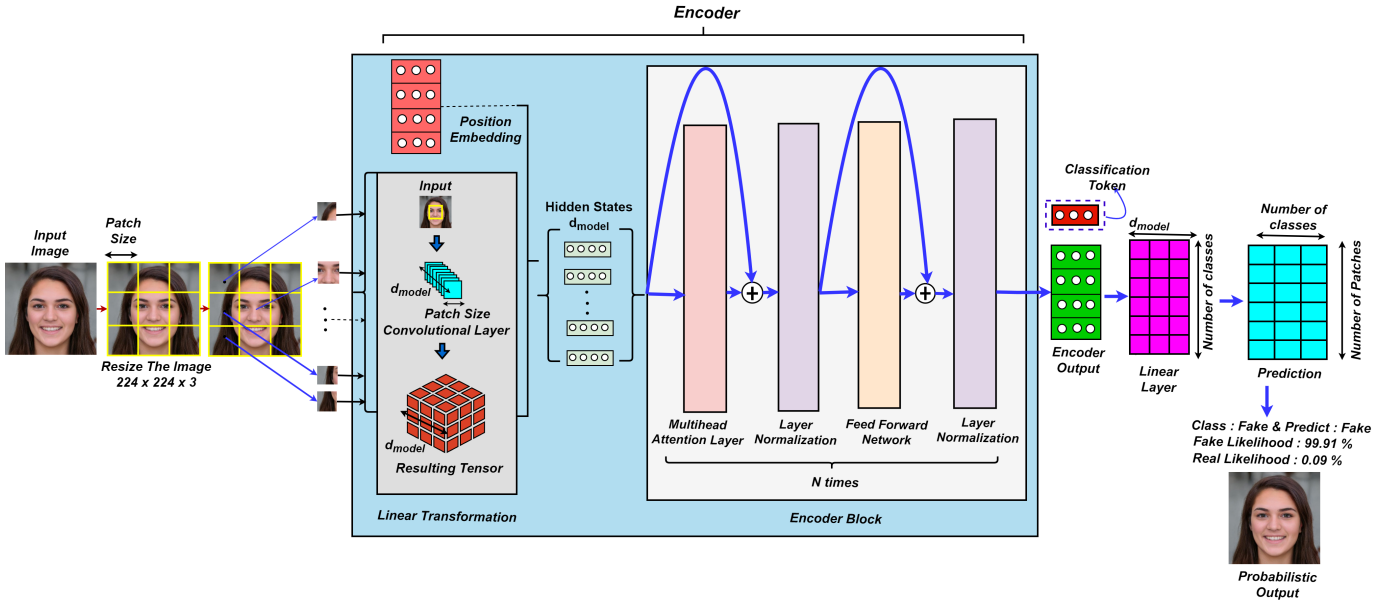


Fig. 3: Architecture of the Vision Fake Detection Network (VFDNET)

NVIDIA compiled. An additional 70000 fake face images exist in this dataset, which originates from the 1 Million Fake Faces dataset developed through StyleGAN and distributed by Bojan.

### B. Dataset Splitting

We have achieved pre-processing, exploratory data analysis, and feature selection on the dataset. In feature selection, we feature the intensity of the image pixels of the train, test, and validation datasets. The data set contains a training, test, and validation dataset. The data sampled were divided into 70% train data, 15% validation data, and 15% test data.

### C. Dataset Training and Validation

For the DFCNET, ResNet50, and MobileNetV3 models, images are rescaled to normalize pixel values to a range of [0,1]. *ImageDataGenerator*(rescale=1./255) is used for normalization. Images are resized to 224 x 224 pixels (target\_size=(224,224)) and data is processed in small batches of 16 images (batch\_size=16). Image data loads from flow\_from\_directory (train\_dir,val\_dir,test\_dir). The class\_mode='binary' sets the binary labels (real vs fake). For training VFDNET, we used the AutoAugment Pipeline(*AutoAugment*, resizing, center cropping, normalization, and tensor conversion) to enhance the training data. We also used the RandAugment Pipeline and the Combined Auto and RandAugment Pipeline.

### D. Evaluation Metrics

Here we outline the evaluation criteria and results analysis. Evaluation of experimental performance in classification studies primarily relies on the accuracy metric, which we express with the mathematical representation of Equation (10):

$$Accuracy = \frac{Correct\ Predictions}{Total\ Predictions} = \frac{TP + TN}{TP + TN + FP + FN} \quad (10)$$

Where TP denotes true positives, TN denotes true negatives, FP denotes false positives, and FN denotes false negatives.

The assessment of image classification system performance uses Precision and Recall as evaluation metrics. Precision calculates the percentage of correct images from the total number of classified images. The mathematical expression represented in Equation (11):

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \quad (11)$$

Here, FP represents misclassified images, and TP represents the correctly classified images.

The recall calculates the percentage of accurately predicted pictures relative to the dataset size. The mathematical form in Equation (12):

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \quad (12)$$

Here, FN denotes false negatives, the images that fit the correct class but are misclassified by the classifier.

The mathematical notation for F1-score shows it as the weighted average of precision and recall values. The mathematical representation of F1-Score is given by Equation (13):

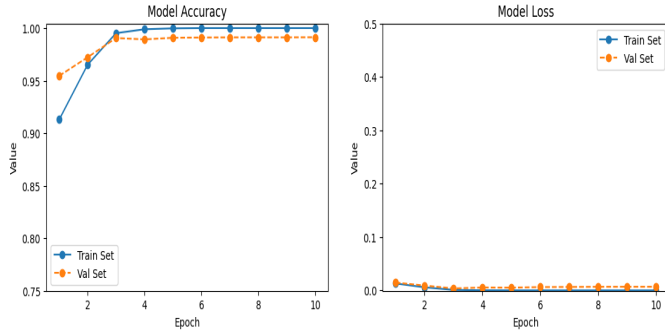
$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (13)$$

## V. RESULTS ANALYSIS

The following subsections present and discuss the experimental results and the comparative analysis.

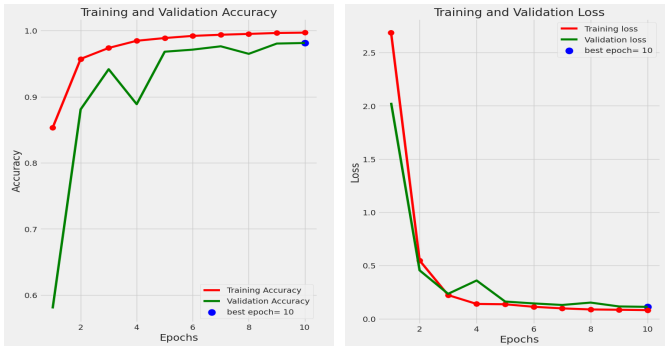
### A. Significance of Performance Across Epochs

Figure 4 and Table I collectively present the training and validation performance of the proposed models. The VFDNET outperformed others, achieving perfect training accuracy and the lowest validation loss of 0.0068, indicating strong



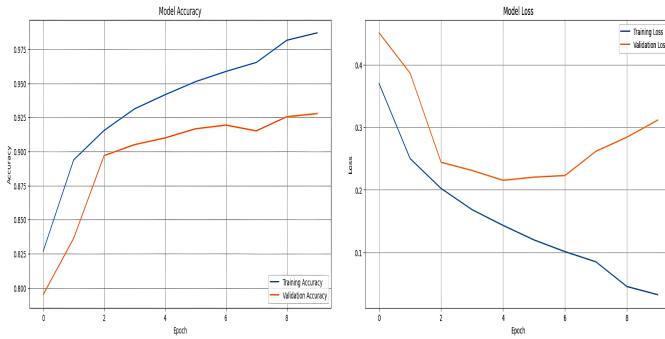
(a) VFDNET Accuracy

(b) VFDNET Loss



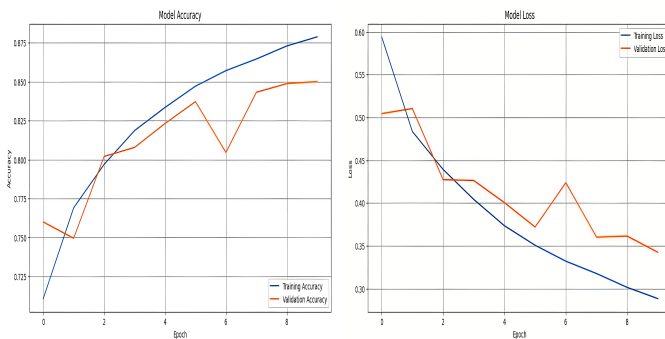
(c) MobileNetV3 Accuracy

(d) MobileNetV3 Loss



(e) ResNet50 Accuracy

(f) ResNet50 Loss



(g) DFCNET Accuracy

(h) DFCNET Loss

Fig. 4: Training and validation accuracy and loss curves for all employed models.

generalization and convergence stability. MobileNetV3 also demonstrated competitive performance with a high validation accuracy of 0.9814 and low loss. In contrast, ResNet50 and DFCNET exhibited higher validation losses and noticeable performance gaps, suggesting overfitting. The plotted learning curves further corroborate these findings, visually highlighting the superior learning dynamics of transformer-based and lightweight architectures.

TABLE I: Performance Benchmark Across Models

Proposed	Train Acc.	Train Loss	Val Acc.	Val Loss
VFDNET	1.0000	0.00001	0.9913	0.0068
ResNet50	0.9870	0.0318	0.9278	0.3117
MobileNetV3	0.9970	0.0805	0.9814	0.1113
DFCNET	0.9882	0.0325	0.9446	0.1842

### B. Results Performance Analysis of Employed Techniques

Table II presents the comparative performance of the proposed and baseline models. The VFDNET model achieved the highest accuracy 99.13% and demonstrated balanced precision, recall, and F1-score 99.00%, indicating superior generalization. MobileNetV3 also performed robustly with 98.00% accuracy. In contrast, DFCNET achieved moderate accuracy 95.76% but maintained high precision and recall with 92% and 91%. ResNet50 recorded the lowest performance across all metrics 84.28%, underscoring the effectiveness of transformer-based and lightweight architectures over traditional CNNs.

TABLE II: Performance Metrics of the Proposed and Baseline Models

Proposed	Acc.(%)	Precision(%)	Recall(%)	F1-Score(%)
MobileNetV3	98.00	98.11	98.09	98.09
VFDNET	99.13	99.00	99.00	99.00
ResNet50	84.28	84.00	84.00	84.00
DFCNET	95.76	92.00	91.00	89.00

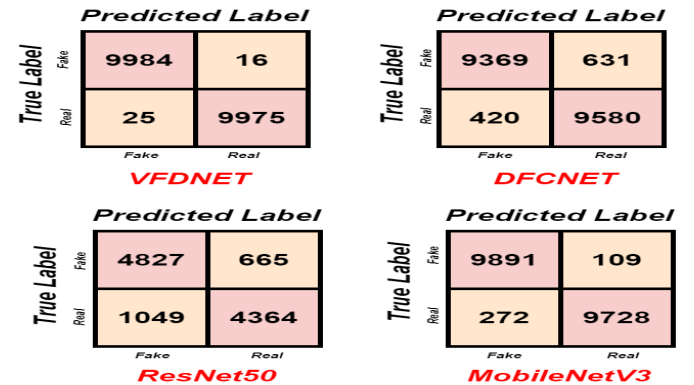


Fig. 5: Class wise Prediction Performance via Confusion Matrices of Proposed Models

Figure 5 represents the confusion matrices of each model tested: VFDNET, DFCNET, ResNet50, and MobileNetV3.

VFDNET achieves the highest accuracy levels, correctly labeling 9984 fake and 9975 real images, with only 41 misclassifications in total. DFCNET identified 9369 fake and 9580 real images correctly but exhibited a higher degree of confusion and misclassification of approximately 1051 samples. ResNet50 ranked lowest in weakness was its performance: only 4827 fake and 4364 real images might rightfully be called classification, with a total number of 1714 misclassifications, all indicating poor generalization potential. MobileNetV3 performed well, correctly labeling 9891 fake and 9728 real images, with 381 misclassifications, ranking second after VFDNET. These results therefore indicate VFDNET to be the most potent deepfake detector, followed by MobileNetV3, which is also highly trusted. Figure 6 illustrates the predictive performance of our best proposed model, VFDNET, on the test dataset images. It displays the predicted class (fake or real) and the corresponding likelihood percentages for both real and counterfeit classifications.



Fig. 6: Accurate Identification of employed VFDNET model

### C. Discussion and Comparative Study

As shown in Table III, the proposed VFDNET model achieved the highest accuracy 99.13%, surpassing existing methods such as VGG16 99% and Ensemble Learning 97.04%. MobileNetV3 also performed strongly with 98% accuracy, demonstrating high efficiency. Although DFCNET recorded a balanced accuracy of 95.76%, it remains comparable to traditional models. Overall, the proposed methods exhibit superior performance, validating their effectiveness for deepfake detection.

## VI. CONCLUSION

The research examined CNN and Vision Transformer systems to detect deepfakes and determined that VFDNET achieved the highest overall performance. Model robustness and generalization improved through the use of preprocessing techniques and augmentation strategies. The research proposes

TABLE III: Comparative Analysis of Existing Works and Proposed Methods

Work	Method	Dataset	Acc. (%)
[7]	Ensemble Learning	CelebA	97.04
[15]	ResNet18	Real and Fake Face	89.50
[6]	VGG16	140k Real and Fake Faces	99.00
	DenseNet-121	140k Real and Fake Faces	92.00
[5]	ViT	Deepfake and Real Images	89.91
Proposed	DFCNET	140k Real and Fake Faces	<b>95.76</b>
	VFDNET	140k Real and Fake Faces	<b>99.13</b>
	MobileNetV3	140k Real and Fake Faces	<b>98.00</b>
	ResNet50	140k Real and Fake Faces	<b>84.28</b>

a dependable method for protecting against deepfake threats in actual situations.

## REFERENCES

- [1] D. S. W. Nguyen and D. T. Ademiluyi, "Deepfake video detection employing human facial features," *Journal of Computer and Communications*, vol. 11, no. 12, pp. 1–13, 2023.
- [2] A. Sensity, "The state of deepfakes 2020: Updates on statistics and trends," 2021.
- [3] B. Chesney and D. Citron, "Deep fakes: A looming challenge for privacy, democracy, and national security," *Calif. L. Rev.*, vol. 107, p. 1753, 2019.
- [4] Y. Patel, S. Tanwar, P. Bhattacharya, R. Gupta, T. Alsuwian, I. E. Davidson, and T. F. Mazibuko, "An improved dense cnn architecture for deepfake image detection," *IEEE Access*, vol. 11, pp. 22081–22095, 2023.
- [5] B. Ghita, I. Kuzminykh, A. Usama, T. Bakhshi, and J. Marchang, "Deepfake image detection using vision transformer models," in *2024 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, pp. 332–335, 2024.
- [6] W. Alkishri, J. Yousif, and A.-B. Mahmood, "Deepfake image detection methods using discrete fourier transform analysis and convolutional neural network," *Jilin Daxue Xuebao (Gongxueban)/Journal of Jilin University (Engineering and Technology Edition)*, vol. 42, p. Online Open Access, 05 2023.
- [7] J. Zhang, K. Cheng, G. Sovernigo, and X. Lin, "A heterogeneous feature ensemble learning based deepfake detection method," in *ICC 2022-IEEE International Conference on Communications*, pp. 2084–2089, IEEE, 2022.
- [8] L. Liu, C. Shen, and A. Van den Hengel, "The treasure beneath convolutional layers: Cross-convolutional-layer pooling for image classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4749–4757, 2015.
- [9] A. Alsaleh and C. Perkgoz, "A space and time efficient convolutional neural network for age group estimation from facial images," *PeerJ Computer Science*, vol. 9, p. e1395, 2023.
- [10] W. Ma and J. Lu, "An equivalence of fully connected layer and convolutional layer," *arXiv preprint arXiv:1712.01252*, 2017.
- [11] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network. arxiv e-prints, art," *arXiv preprint arXiv:1505.00853*, 2015.
- [12] D. M. Pelt and J. A. Sethian, "A mixed-scale dense convolutional neural network for image analysis," *Proceedings of the National Academy of Sciences*, vol. 115, no. 2, pp. 254–259, 2018.
- [13] C.-F. R. Chen, Q. Fan, and R. Panda, "Crossvit: Cross-attention multi-scale vision transformer for image classification," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 357–366, 2021.
- [14] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1314–1324, 2019.
- [15] R. Rafique, R. Gantassi, R. Amin, J. Frnda, A. Mustapha, and A. H. Alshehri, "Deep fake detection and classification using error-level analysis and deep learning," *Scientific reports*, vol. 13, no. 1, p. 7422, 2023.