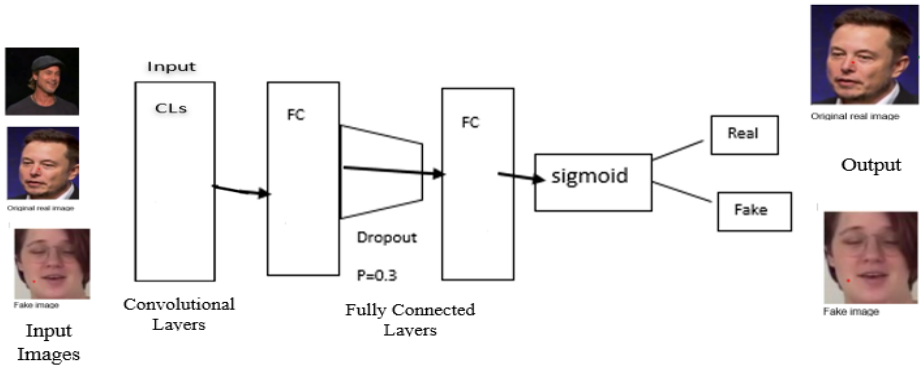
Deepfake Image Detection: Comparison of Different Convolutional Neural Networks for Image Detection

M. C. Weerawardana and T. G. I. Fernando*

Department of Computer Science, Faculty of Applied Sciences, University of Sri Jayewardenepura, Sri Lanka

Date Received: 02-10-2025 Date Accepted: 26-12-2025



Abstract

Deepfake technology relies on advanced theoretical concepts of machine learning and deep learning. Machine-generated fake images can present fictional content as real, often by reenacting or swapping faces using artificial neural networks. The human eye cannot easily distinguish between real and fake faces created by deepfake technology. Therefore, people face numerous difficulties due to the lack of an accurate deepfake detection method. The challenge here is to implement a reliable method for detection of deepfakes due to their rapid spread and the ease of generating deepfakes. In this paper, our objective was to analyze the Convolutional Neural Networks (CNNs) based- existing deepfake detection methods to classify input images as fake (0) or real (1). For our experiment, we employed three different convolutional neural network architectures: 1) 2D CNN, 2) DenseNet-121, and 3) VGGFace-16. We compared their performance and further explored the use of grayscale images and data augmentation techniques with the DenseNet-121 architecture. We used Accuracy, F1-score, Recall, and Average Precision metrics to assess our models and ensure the effectiveness of their performance. We used the two most known publicly available datasets ("140K Real and Fake Faces" dataset and "Real and Fake Face detection" dataset), containing both real and synthetic face images. Our study concluded that the implemented DenseNet-121 architecture showed the best performance with an accuracy of 0.96 and F1-score of 0.96. Although the VGGFace-16 model demonstrated comparable accuracy (0.89), it is computationally very expensive and requires a sophisticated processor in training augmented data. The performance of the custom CNN model (accuracy - 0.90) was comparable to that achieved by the VGGFace-16 model. Our study indicates that existing CNN-based deepfake detection methods may no longer be fully effective due to advancements in current deepfake datasets.

Keywords: *Machine learning, Deepfakes, Convolutional neural networks, Deep learning, Deepfake image detection*

**Correspondence:* tgi@sjp.ac.lk

1. Introduction

With the recent advances of deepfake technologies, computer-generated images, videos, and sounds have become real instead of mere creations. Deepfakes can reenact and manipulate images by swapping one person's face with another using artificial neural networks. To date, numerous deepfake applications have emerged across various societal domains. Notably, during the 2020 United States presidential election, several instances of manipulated or synthetically generated media involving political figures were reported, raising concerns about the potential misuse of deepfake technologies for political misinformation (Parker and Ashley, 2020; Jake and Shayan, 2022). Fake news, malicious information, pornographic content of celebrities, financial fraud, criminals, and suicides are bad sides of the synthetic applications. The movie industry is currently undergoing a major transformation due to deepfake technology. A popular example is the instance where the face of Amy Adams was modified to the face of actor Nicolos Cage to get a fake image (Marek, 2019).

The development of fake creation technologies is the reason for a rise in harmful incidents day by day. FaceForensics (FF) (Rossler et al., 2018) and FaceForensics++ (FF++) (Rossler et al., 2019) have helped the development of face manipulations and reenactments. Forgery manipulation methods and techniques became more powerful and realistic due to the advancement of the software, tools, and datasets. Face2Face (Thies et al., 2016), Deepfake (Marek, 2018), and FaceSwap (Marek, 2019) are the most used, well-known, powerful forgery creation techniques. Recently, deepfakes have played the main role among those technologies. Generative Adversarial Network (GAN) is a well-known tool for generating high-quality synthesized forgeries. Fake content spreads rapidly via social media platforms like Facebook, YouTube, and Twitter.

On the other hand, deepfake technology presents a growing risk to personal identity and digital security. When online interactions become a crucial part of our daily lives, it is important to ensure the security and authenticity of digital content. Detecting a deepfake image, video, or audio is treated as a binary classification (1 or 0), where the system determines whether the content is genuine or artificially generated. CNNs have been shown to be particularly effective in addressing such classification challenges, due to their strong ability to recognize patterns and features. In this study, our primary objective was to implement and evaluate several CNN architectures and compare the performance of several existing CNN-based models for deepfake image detection. We showed that state-of-the-art CNNs can distinguish deepfakes with minimal mis-classification inaccuracies between real and fake images. These minimal inaccuracies remain a critical area of future research. In addition, color images and gray scale images show the difference in pixels and resolution. Our study showed the color impact on classification of fake or real in deepfake images. Our comparative analysis is very important for future research to get an idea about the poor performance of the CNN-based state-of-the-art deepfake detection methods. Our custom CNNs showed good performances, because they are trained and evaluated on current deepfake datasets. The current datasets are rich in qualitative and quantitative parameters. The current datasets ensured the variety in gender, skin tone and age including wide-ranging lighting conditions and head poses of images. Our study showed that existing CNN-based deepfake detection methods are not valid methods furthermore, because technology is improving day by day. However, future researchers attempt to discover a novel method for detecting deepfakes generated by advanced generating technologies.

For our research, we used three CNN-based architectures for binary image classification—distinguishing real (labelled as 1) from fake (labelled as 0) images—using large-scale publicly available datasets. For deepfake image detection, we employed two existing CNN frameworks: VGGFace (Parhi et al., 2015) and DenseNet (Huang et al., 2017), along with a custom-designed CNN architecture. To facilitate comparative analysis, we utilized three types of CNN architectures: (1) 2D CNN, (2) DenseNet-121, and (3) VGGFace-16. The experiments were conducted using two datasets: "The 140K Real and Fake Faces Kaggle Dataset" (Kaggle, 2019a) and "The Real and Fake Face Detection Kaggle Dataset" (Kaggle, 2019b). In total, five models were trained for this comparative study. Among them, three models were based on the DenseNet-121 architecture, one employed

*Correspondence: tgi@sjp.ac.lk

VGGFace-16, and one utilized the custom CNN. The performance evaluation and accuracy comparison of those models were employed with some existing state-of-the-art approaches. All implementations are carried out using the Keras library imported into Python.

This paper has been organized as follows: Part 1 presents a brief introduction to deepfakes, along with an overview of the structure. This part also includes a concise literature review focused on deepfake image detection using convolutional neural networks (CNNs). Part 2 describes the experimental method. Part 3 presents experimental findings and discussion. Finally, Part 4 reaches the conclusion with the key contributions of our experiment.

1.1 What is the Deepfake

Deepfake technology has become an increasingly concerning phenomenon in contemporary society. Utilizing artificial neural networks, this technology can manipulate facial features and perform face-swapping with remarkable realism. The advancement of deepfake generation techniques has progressed in parallel with the development of deepfake detection methods. Since 2018, there has been a rapid proliferation of deepfake-related applications, tools, and synthetic media content. As a result of these developments, the need for robust and effective deepfake detection techniques has grown significantly. Correspondingly, the number of academic publications related to deepfakes have shown a steady increase.

Deepfake is a new technology. Deepfake is combined with the theoretical concepts of machine learning and deep learning. A deepfake algorithm captures a source frame and reenacts it with a target frame using artificial neural networks (Weerawardana and Fernando, 2021). Face replacement could be done in three forms. Face swap, face reenactment, and full fake face generation are the above-mentioned three deepfakes manipulation forms. Recently, many deepfake generation tools have been made available commercially. Examples are Face2Face, FaceApp, and FakeApp (Bitouk et al., 2008). FaceApp automatically reenacts the source face and generates a rich reenacted face, changing skin tone, age, gender, hairstyle, and other attributes. For reenactments, FaceApp is used on smartphones as the hardware platform. In contrast, FakeApp is a desktop-only application used specifically for creating deepfakes. The first known deepfake was created by a Reddit user (Reddit_User, 2017), who utilized publicly available video footage, content from social media platforms, and the TensorFlow software library. Deepfake generation is a complex process, and its detection is naturally even more challenging.

1.2 Convolutional Neural Networks

Recently, most deepfake image detection methods have utilized Convolutional Neural Networks (CNNs). CNNs are considered user-friendly and are accessible even to beginners. CNNs are a type of feedforward neural network, applies convolution operations in one or more layers (Du and Swamy, 2019). Notably, CNN models are capable of recognizing faces even when they are modified—such as being rotated, horizontally flipped, or zoomed. CNN models can understand and capture any spatial correlations across the given input image instead of in vector visualization. The following equations present the theoretical basis of convolutional operations, while Figure 1 illustrates their practical implementation. For example, the CNN network reads an attribute $X \in \mathbb{R}^{n \times m}$ as an input image. The CNN operation gets a kernel in two-dimensional form as $K \in \mathbb{R}^{k \times \ell}, k \leq n, \ell \leq m$ (Goodfellow et al., 2016).

The convolutional form is as follows,

$$(X * K)i, j = \sum_k \sum_r X_{k,r} K_{i-k,j-r} \quad (1)$$

In the first iteration, filter F comes to the top right (from top to bottom or from left to right) corner of the image as x^1. The following equations give the resulting pixel in the output image. In Figure 1 denotes the last two iterations of the CNN network on the top row of the input image with 3x3 filters.

*Correspondence: tgi@sjp.ac.lk

$$F(X^1) = \sum_{i=1}^{3} \sum_{j=1}^{3} F_{i,j} X_{i,j} \quad (2)$$

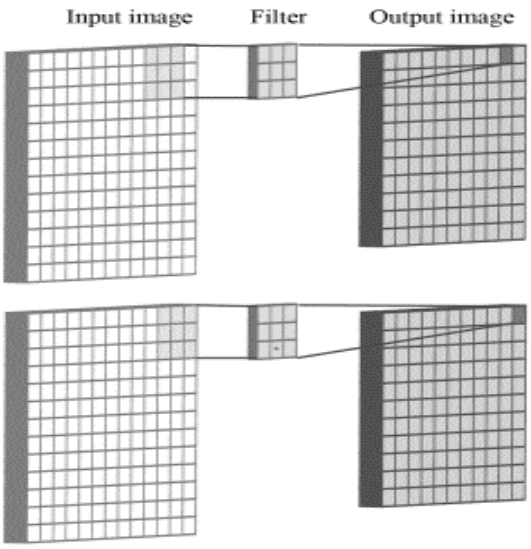


Figure 1. The CNN network uses filters on images.

1.3 Deepfake Image Detection

Several deepfake image detection methods have been developed to successfully detect synthesized content; however, they still exhibit certain limitations. There is an important requirement to find deepfake detection methods due to rapid improvement of fake creation methods. According to the literature, some CNN-based forgery image analysis methods have been proposed. Copy-Move image detection method (Wu et al., 2018[a], 2018[b]; Pandey et al., 2014) and Splicing image detection method (Wu et al., 2019) are applicable only for detecting some swapped face images. These two methods are not valid for the deepfake images. Input image and its privacy preservation have been identified in 2008 (Bitouk et al., 2008). The method (Bitouk et al., 2008) searches for a target face in the given dataset that has a similar appearance to the input face. This method has also considered blending manipulations of a similar face into the source face. However, this method failed to cover a seamless swapping of any two faces. Later, a method was implemented to identify general misrepresentations of any image (Bayer and Stamm, 2016). Furthermore, an image detection method has been implemented to detect the double JPEG compression on an input image (Barni et al., 2017). In the meantime, detection methods have increasingly shifted toward computer graphics and photographic images in order to establish clear distinctions (Rahmouni et al., 2017). Rahmouni's findings are beneficial for some current researchers. Later, feature-based image detection methods have shown some progress. In 2019, eyes, mouth, and facial outline were considered as features to detect computer-generated faces (Matern et al., 2019). However, this attempt is not applicable further.

Recently, Korshunova et al. introduced a model that was trained using pre-selected two source faces (Korshunova et al., 2017). These source faces have been generated using the GANs tool. The benefit of this model goes toward research that continues to detect only GAN-generated forgeries. In 2019, Zhang et al. tried to implement a new model called AutoGAN to classify deepfake images (Zhang et al., 2019). They used the spectra of the frequency of an image. AutoGAN is a GAN simulation framework. In addition, the author used the color frequency of the image to identify the reenactments of a face image. However, some researchers turned to color inconsistency instead of color frequency. Later, some researchers moved to differentiate color inconsistency between the real image and the GAN-generated image (Li et al., 2018; McCloskey and Albright, 2018). The authors have managed to classify the selected image in non-RGB color spaces. However, the method was unable to identify the local regions of deepfake images. In the meantime, Zhou et al. have made a good

*Correspondence: tgi@sjp.ac.lk

effort to detect the altered faces only in images using a GoogleNet-based CNN model (Zhou et al., 2017). Instead of faces, Cozzolino detected fake fingerprints using a CNN model called NoisePrint (Cozzolino and Verdoliva, 2018). MesoNet (Afchar et al., 2018) is another CNN architecture, which is used to classify the real and fake images generated by Face2Face and DeepFakes.

According to the literature, most of the CNN-based face swapping, bending, and face reenactment techniques were based on similarity of face or face patches between target and source image. Advancement of the deep learning techniques, existing CNN methos are not applicable furthermore for deepfakes. We do not have a clear notification to state whether the methods mentioned would be effective in detecting deepfake faces because they are captured in detecting facial expressions, lips movements or eye blinking only. Thus, existing CNN-based detection methods need to be changed and improved on deepfakes. However, the continued advancements and developments of face swapping techniques will result in generation of more complex deepfakes. It will be a big challenge for deepfake detection research due to being harder to detect by existing methods. For successful achievement, more generic algorithms and databases need to be implemented in future. Our comparative study clearly showed that traditional CNN-based methods should be improved and customized on advancements of current deepfakes.

2. Methodology

Our main objective was to evaluate the effectiveness of CNN-based deepfake image detection models. We employed five CNN models overall for Deepfake image detection and used two large-scale publicly available datasets: "140K Real and Fake Faces Kaggle dataset" and "Real and Fake Face Detection Kaggle dataset". We used pre-trained VGGFace-16 and DensNet-121 for the feature extraction task.

Among the five CNN models, three were implemented by modifying the DensNet CNN architecture, one was based on the VGGFace architecture, and the other was a custom-designed CNN model. We provide a comprehensive explanation of these five models in the section below.

2.1 Our Trained CNN models

For our experiment, we used pre-trained (on ImageNet dataset) VGGFace-16 and DenseNet-121 architectures along with a custom CNN framework. DensNet is an extension of the Residual CNN architecture (ResNet). ResNets are widely used for facial recognition in the field of computer vision. DenseNet is different from ResNet and other CNN networks. Because all layers of the DenseNet network are interconnected directly. For the strongly connected interconnection between each layer, the DenseNet network has been used with dense blocks (Huang et al., 2016). In our work, DenseNet has been utilized in different layers, like the transition layer, average pooling layer, and normalization. For the purpose of comparative performance, we used the VGGFace-16 architecture besides the DenseNet architecture.

VGGFace architecture has been developed by the Visual Geometry Group at Oxford University (Parkhi, 2015). VGGFace is also an image recognition model and has been pre-trained on the standard face recognition dataset. The VGGFace framework has been built upon the theoretical concept of the deep convolutional neural network architecture, including several convolutional blocks and small kernels. The maxpooling layers followed the ReLU activation functions.

2.1.1 Architectural Design and Hyperparameter Selection

We utilized a custom Convolutional Neural Network (CNN) model to perform a comparative analysis alongside the DenseNet-121 and VGGFace-16 architectures. The custom CNN architecture (Figure 2) was designed to balance classification performance and computational efficiency. Notably, the custom model has combined the dropout and padding techniques, which were not explicitly integrated in the original VGGFace and DenseNet architectures. Dropout layers were incorporated to

*Correspondence: tgi@sjp.ac.lk

improve generalization by mitigating overfitting, while batch normalization was used to stabilize training and accelerate convergence. Hyperparameters such as learning rate, batch size, and number of epochs were selected based on commonly adopted values in prior deepfake detection studies and empirical observations during preliminary experiments. The Adam optimizer was employed due to its adaptive learning rate mechanism, which is effective for training deep neural networks. Binary cross-entropy loss was used as the objective function, as the task involves binary classification between real and fake images. We used Sigmoid activation function in the output layer of the model. Figure 2 illustrates the structural design of the base CNN architecture. We implemented five different models described in the following section by changing the structural configuration of each model.

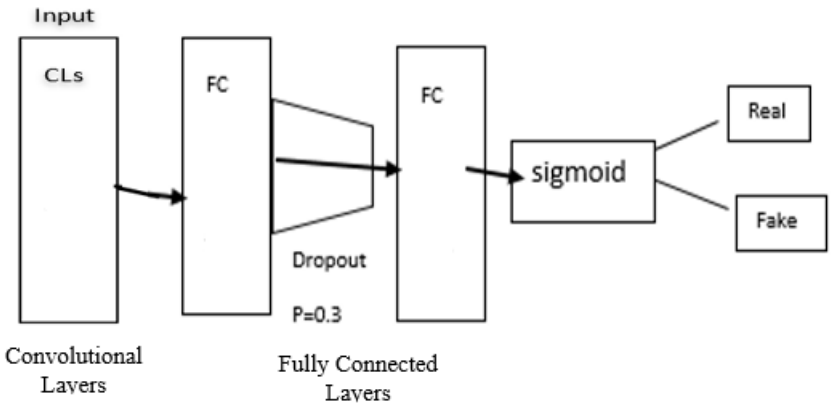


Figure 2: The CNN -based model architecture

Model 1 - Custom CNN Model: We used 6 convolutional layers for the custom CNN model between the batch normalization and the maxpooling layers. Rectified Linear Unit (ReLU) was the activation function for all convolutional layers of the model. Overfitting minimization, we applied dropout for each convolutional layer. In addition, we integrated data augmentation into the inputs to ensure the accuracy changes. Also, we applied a horizontal flip to 0.2 as a zoom range to images to be zoomed in. We wanted to ensure that all numbers in the RGB matrix were in the range of [0,1], to satisfy the rescaling factor.

Model 2 - VGGFace 16: As model 2, we used the pre-trained VGGFace-16 architecture. Model 2 contained five-layer blocks, and each block contained convolutional and maxpooling layers. Two 3x3 convolutional layers and one maxpooling layer consisted of the first layer block and the second layer block. Three 3x3 convolutional layers and one maxpooling layer were included on the third layer block, fourth layer block, and fifth layer block of the model. We used ReLU as an activation function. We fine-tuned the VGGFace architecture due to pre-trained weights. Additionally, a dense layer was appended after the fifth convolutional layer block. The additional dense layer contributed to capturing the facial features of the input faces. Finally, we applied the Sigmoid activation function to the output dense layer.

Model 3 - DenseNet 121: In model 3, we added the dense layer after the existing convolutional layer at the end of the CNN architecture. We integrated DenseNet-121 architecture in Keras. Also, we added a few modifications to observe the accuracy of the model. We included four dense blocks to the model. These dense blocks are connected closely with layers, batch normalization, 3x3 convolutional layers, ReLU activation and transition layer. The transition layer contained a 2x2 average pooling layer and a 1x1 convolutional layer. Further, we added a custom dense layer feeding the Sigmoid function following the final dense block. Our prediction was binary classification labeled: real-1, fake-0. Also, we used the Sigmoid function as the activation function.

*Correspondence: tgi@sjp.ac.lk

Model 4 - DenseNet with Data Augmentation: Model 4 used the same layers as in model 3. Additionally, we added image rotation and set horizontal flip to 20 as the required image rotation range. We ensured that the RGB matrix values are normalized to the range [0,1] to satisfy the rescaling factor.

Model 5 - DenseNet with Gary Scale: Model 5 used the same layers as in model 3. The specialty of the model 5 was adding gray scale images. The purpose of using gray scale ensured the impacts of the color on classification of real versus fake images.

Gray Scale: The grayscale-based experiment was designed to investigate the role of color information in distinguishing real and fake facial images. Previous studies have suggested that many deepfake artifacts arise from inconsistencies in texture, illumination, and facial structure rather than chromatic information. To examine whether color cues significantly contribute to classification performance, Model 5 was constructed using the same DenseNet architecture as Model 3, with only modification being the conversion of all input images to grayscale. A pre-trained DenseNet-121 model was employed, and all color images were converted to grayscale using the color mode parameter during preprocessing. This experimental design allowed for a controlled comparison between color and grayscale inputs, thereby isolating the impact of color information on deepfake detection performance.

Data Augmentation: Data augmentation is a powerful technique to improve model generalization by artificially increasing the size of the training dataset with transformed versions of existing data. We applied transformations like horizontal flipping, random rotations, zoom, etc. incorporating data augmentation using TensorFlow's ImageDataGenerator.
Data Augmentation with ImageDataGenerator:
horizontal_flip=True: Random horizontal flipping.
rotation_range=20: Random rotations from -20° to +20°.
zoom_range=0.2: Random zoom in/out by up to 20%.
width_shift_range=0.2 and height_shift_range=0.2: Random shifts in width and height by 20%.
shear_range=0.2: Random shearing transformations.

2.2 Model Training

Both VGGFace-16 and DenseNet-121 models are existing trained models, and they were trained on different datasets. To compare performance, all models were retrained and evaluated under a common experimental setup. We have used Python scripts and Jupyter notebooks for the coding task. Google Colaboratory was utilized for model training and code execution, which has given its support for GPU acceleration. Because of the substantial size of the dataset, we fed 100,000 inputs to train the model. 20,000 images are set for testing and validation. The source was the "Real and Fake Face Detection" dataset on Kaggle. The models have been trained for 10 epochs using Adam optimizer with a batch size of 64. The CNN models were trained with a learning rate of $5x10^{-4}$and a learning rate decay factor of 0.92. According to the above-mentioned resources and conditions, we retrained the above models on a single dataset (Real and Fake Face Detection dataset) and saved them as trained models in the Google Drive.

2.3 DataSets

We utilized two datasets from Kaggle, which contributed to gaining performance and effectiveness of the models. One of the primary datasets was the "140K Real and Fake Faces" dataset and the other one was the "Real and Fake Face detection" dataset. We combined both datasets together for the testing of the models.

2.3.1 140K Real and Fake Faces Dataset

*Correspondence: tgi@sjp.ac.lk

The "140K Real and Fake Faces Dataset" includes totally 140,000 human faces. 70,000 are real faces, while the other 70,000 are synthetic face images. The real faces have been collected from the Flickr, while the fake faces have been generated using StyleGAN. The original dataset was partitioned into three subsets such as training, validation, and testing, consisting of 100,000, 20,000, and 20,000 face images, respectively. Each subset maintains an equal quantity of fake faces and real faces e.g., testing subset consists of 10,000 real images and 10,000 synthetic images. The dataset encompasses diversity in age, gender, skin tone, and ethnicity, enhancing model generalizability. For binary classification, we assigned the label 1 to real faces and label 0 to fake faces.

2.3.2 Real and Fake Face Detection Dataset

This dataset has been divided into training, validation, and testing sets, each containing both real faces and fake faces. The fake face images (960 in total) are high-quality, expert-generated manipulations created using Photoshop. These manipulations may involve the entire face or specific facial feature: eye, nose, or mouth. However, this dataset includes 1,081 real, unaltered face images. This dataset provides valuable examples of localized facial forgery, making it a useful complement to the synthetic fake faces in the "140K Real and Fake Faces" dataset.

3. Results and Discussion

The effective performance of our five models utilized in this study is depicted in Table 1. The most commonly used evaluation metrics are Accuracy, F1-Score, Recall, and Average Precision (AP). Higher values of Accuracy or AP correspond to superior model performance.

Table 1: Performance of our Deepfake detection models

CNN Models	Metrics			
	Accuracy	**F1-score**	**Recall**	**Avg. Precision**
Model 1-Custom CNN	0.90	0.89	0.89	0.89
Model 2-VGGFace-16	0.89	0.89	**0.95**	**0.95**
Model 3-DenseNet-121	**0.96**	**0.96**	0.93	0.94
Model 4-DenseNet with Data Augmentation	0.87	0.83	0.78	0.78
Model 5-DenseNet with Gray Scale	0.95	0.95	0.94	0.94

According to the results, the custom CNN model shows better performance (Accuracy is 90% and F1-score is 89%). Among the architectures tested, DenseNet-121 achieved the highest accuracy (0.96) and F1-score (0.96) on the test dataset. The VGGFace-16 model exhibited performance comparable to both the custom CNN and DenseNet models; however, it requires significantly more advanced hardware resources, such as a sophisticated processor, to train effectively on augmented data. Consequently, the VGGFace-16 model is computationally more expensive compared to DenseNet-121. As shown in Table 1, DenseNet-121 performed well, delivering results closely aligned with those of the VGGFace-16 and custom CNN models. Our Model 4, DenseNet with data augmentation, yielded a lower accuracy of approximately 87%, likely due to the increased computational demands of training on augmented images without sufficiently powerful hardware. Conversely, the DenseNet model (Model 5) trained on grayscale images achieved an accuracy of 95%, indicating the same results as the DenseNet-121 model (Model 3-96%). The comparative results indicate that grayscale inputs achieved performance comparable to color images due to color images and grayscale images show the difference in pixels and resolution. According to our results, DenseNet models (Model 3 and 5) demonstrate higher accuracies compared to custom CNN and VGGFace-16 models. Although the proposed models demonstrate strong performance across multiple evaluation

*Correspondence: tgi@sjp.ac.lk

metrics, statistical significance testing and confidence interval analysis were not conducted. This limitation will be addressed in future work through repeated experiments and cross-dataset evaluation.

Figure 3(a), (c), and (e) illustrate the training and validation loss of the VGG-16 model, Custom CNN model, and DenseNet-121 model. Figure 3(b), (d) and (f) illustrate the corresponding training and validation accuracy of the VGG-16 model, Custom CNN model, and DenseNet-121 model respectively. In all figures, along the x-axis and the y-axis, the number of epochs and loss and accuracy for each epoch are displayed, respectively.

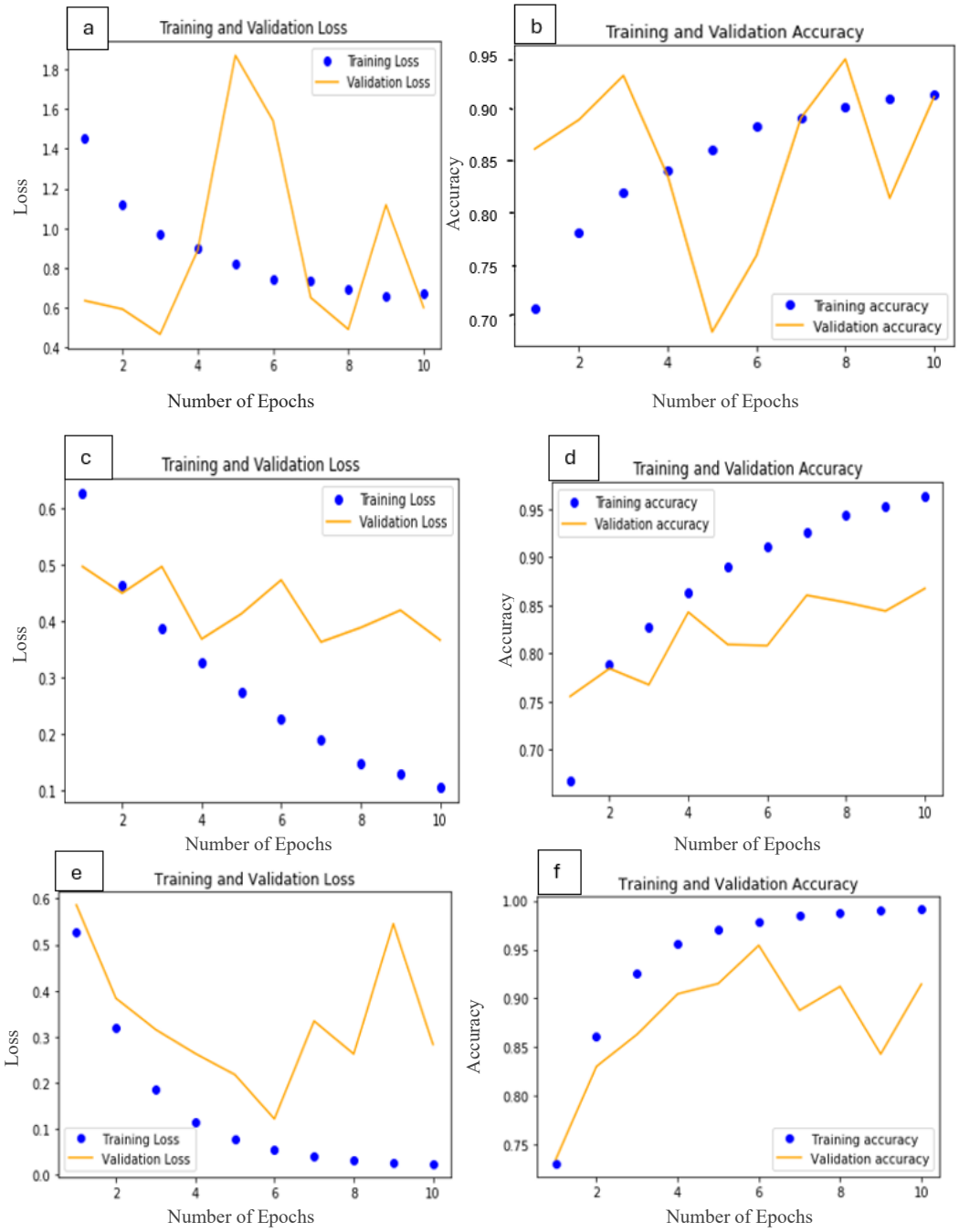


Figure 3: (a) The graph of training and validation loss of VGGFace-16 model **(b)** The graph of training and validation accuracy of VGGFace-16 model **(c)** The graph of training and validation loss of custom CNN model **(d)** The graph of training and validation accuracy of custom CNN model **(e)** The graph of training and validation loss of DensNet-121 model **(f)** The graph of training and validation Accuracy of DensNet-121 model

*Correspondence: tgi@sjp.ac.lk

In Figure 3(a) training accuracy increases gradually and validation accuracy fluctuates in the VGGFace-16 model. But line comes to near 0.90 accuracy in 10 epochs. In Figure 3(d) training accuracy and validation accuracy increase gradually of the custom CNN model. But training accuracy is nearly higher than validation accuracy. In Figure 3(f) training accuracy and validation accuracy increase gradually in the DensNet-121 model. But training accuracy is nearly higher than validation accuracy. In our study, training accuracy line increases gradually while validation accuracy line fluctuates due to training dataset is higher than validation dataset. Figure 3(a), (c) and (e) depict the corresponding loss lines of each model. In a model, loss decreases while accuracy increases.

The overfitting observed in the custom CNN and DenseNet-121 models arises from multiple factors. The relatively high number of trainable parameters compared to the available training data encourages memorization rather than generalization. We used 100,000 data for training and 20,000 data for validation on the "Real and Fake Face Detection" dataset on Kaggle. The size of the dataset effect for the accuracy. Moreover, while data augmentation was employed (for custom CNN model and DensNet-121 model), it may not sufficiently reflect the complexity of real-world deepfake variations. In addition, inherent biases in deepfake datasets, such as recurring manipulation artifacts, can be unintentionally learned by the models. The depth of the DenseNet-121 architecture further amplifies this issue, particularly when regularization strategies are not optimally configured.

Table 2 displays the performance comparison of the pretrained VGGFace-16 and DenseNet-121 architectures with our implemented models. The pretrained VGGFace-16 and DenseNet-121 architectures has been trained on ImageNet dataset. Our models were trained on the "Real and Fake Face Detection" dataset and tested on both datasets used. Our models achieve higher accuracy than previously reported results, with DenseNet-121 outperforming VGGFace-16 across all evaluated metrics (accuracy-96%, precision-94%, recall-93%, and F1-score-96%). Direct comparison with prior work is limited because most studies report only accuracy and use different datasets or preprocessing methods. By providing multiple performance metrics, our study offers a more comprehensive and transparent assessment of deepfake detection performance.

Table 2. Comparative results between existing deepfake detection models and our models

CNN architecture	Reported Accuracy (%)	Our Accuracy (%)	F1-score	Recall	Avg. Precision
VGGFace-16	81.6 (Tolosana et al., 2020)	89	0.89	0.95	0.95
DenseNet-121	92.3 (Huang et al., 2017)	96	0.96	0.93	0.94

In our study, each dense block of DenseNet architecture was employed with transition layers, an average pooling layer, and batch normalization. These additional layers facilitate the feature map concatenation. This concatenation enables knowledge sharing across all layers, enhances the flow of learned features, and allows efficient gradient propagation from the initial input to the loss function. DenseNet requires a relatively small number of feature maps and parameters, making it less computationally expensive compared to other convolutional neural networks. Consequently, DenseNet is both a powerful and efficient architecture.

VGGFace architecture was chosen for its ability to leverage large-scale training data with limited computational resources, having been trained on a dataset comprising over two million facial images.

Training constitutes a critical component of the model learning process. Both supervised and unsupervised learning approaches can be utilized for model development; in our experiments, we employed supervised learning to train the models. Model training constitutes a critical phase of the learning process. Training can be time-consuming, and its duration varies depending on hardware resources, dataset size, quality of the dataset, number of epochs, and which machine learning algorithm is employed. Achieving high accuracy and reliable performance typically requires quality and large

*Correspondence: tgi@sjp.ac.lk

size of data. Additionally, sufficient hardware resources are needed for efficient training; ideally, the computing system should include an NVIDIA GPU. We had to face a few difficulties during the model training. We only have a low-cost laptop and its local hardware resources are not enough for model training. Especially, model training takes too much time for the complete model training. For higher accuracy, more than 100,000 data for the training requires more. It takes too much time for successful training. Many of the Deepfake detection methods are based on modern machine Learning techniques and large quantities of robust training data. An adequate dataset contains around hundreds of thousands of training data to evaluate models with high accuracy. Performance of the implemented model is highly dependent on the training dataset. However, since we do not have enough local resources for model training, it was decided to join Google CoLaboratory. Thus, Google CoLaboratory was utilized to train and to execute all our models. Power cutting and unstable internet connection jump a big trouble during the model training. We had to train one model a few times for one task. It was more time consuming and resource wasting.

Our experimental results indicate that DenseNet-121 consistently outperforms the custom CNN and VGGFace-16 models in deepfake detection. However, several factors may limit the broader applicability of these findings. Dataset biases and limited diversity could reduce performance when models are applied to unseen data or different datasets. Overfitting remains a concern, particularly for deep networks trained on relatively small or augmented datasets. Computational requirements also present practical challenges for real-time or large-scale deployment. Despite these constraints, the results emphasize that architectures with deep layers and extensive feature reuse are promising for improving detection accuracy, highlighting directions for future work such as cross-dataset validation and optimization for efficient deployment.

4. Conclusions

Many deepfake image detection models rely on advanced modern machine learning theories. Approaches such as computer vision-based, multimodal models, and scalable architecture are well-suited for detecting deepfakes.

This paper was able to give a brief introduction about deepfakes and their recent applications. We then provided a concise literature survey within the introduction, covering deepfakes, convolutional neural networks, and related work in Deepfake image detection methods. In the Methodology section, we described the process of model training and testing. The Results and Discussion section presented the analyzed outcomes of each model. Finally, we summarized the key findings in the Conclusion section. Based on comparative analysis, we concluded the following points:

- The training process became more complex when augmented image samples were included, indicating that such data introduces additional variability.
- Models trained over a greater number of epochs generally achieved improved performance.
- Based on the analysis of Model 5, color and grayscale images exhibited differences in pixel distribution and resolution; however, color did not have a significant impact on the classification of real versus fake images.
- Overall, the results show that state-of-the-art CNNs can distinguish with minimal misclassification inaccuracies between real and fake images. These minimal inaccuracies remain a critical area for future research.

References

Afchar, D., Nozick, V., Yamagishi, J., Echizen, I., 2018. Mesonet: a compact facial video forgery detection network, In IEEE International Workshop on Information Forensics and Security (WIFS), 1–7.

Barni, M., Bondi, L., Bonettini, N., Bestagini, P., Costanzo, A., Maggini, M., Tondi, B., Tubaro, S., 2017. Aligned and nonaligned double jpeg detection using convolutional neural networks. Journal of Visual Communication and Image Representation. 49, 153–163.

*Correspondence: tgi@sjp.ac.lk

Bayar, B., Stamm, M.C., 2016. A deep learning approach to universal image manipulation detection using a new convolutional layer, In Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security ACM, 5–10.

Bitouk, D., Kumar, N., Dhillon, S., Belhumeur, P., Nayar, S.K., 2008. Face swapping: Automatically replacing faces in photographs. ACM Trans. Graph. 27, 1–8.

Chollet, F., 2017. XceptionNet: Deep Learning with Depthwise Separable Convolutions. In CVPR, 1251–1258.

Cozzolino, D., Verdoliva, L., 2018. Noiseprint: a cnn-based camera model fingerprint. arXiv preprint arXiv: 1808.08396.

Du, K.L., Swamy, M.N.S., 2019. Neural Networks and Statistical Learning. Springer London, London.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep learning. MIT press, Cambridge (MA).

Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q., 2017. Densely connected convolutional networks, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (CVPR), Honolulu, HI, USA, pp 4700-4708. https://doi.org/10.1109/CVPR.2017.243

Jake, H., Shayan, S., 2022. False claims of deepfake President Biden go viral, BBC Reality Check & BBC Monitoring. https://www.bbc.com/news/62338593.

Kaggle., 2019a. 140k-real-and-fake-faces. https://www.kaggle.com/datasets/xhlulu/140k-real-and-fake-faces. Accessed November 2024.

Kaggle., 2019b. real-and-fake-face-detection. https://www.kaggle.com/datasets/ciplab/real-and-fake-face-detection. Accessed November 2024.

Korshunova, I., Shi, W., Dambre, J., Theis, L., 2017. Fast face swap using convolutional neural networks, In IEEE International Conference on Computer Vision (ICCV), 3697– 3705.

Lai, H., Xiao, S., Pan, Y., Cui, Z., Feng, J., Xu, C., Yin, J., Yan, S., 2018. Deep recurrent regression for facial landmark detection. IEEE Transactions on Circuits and Systems for Video Technology, 1144–1157.

Li, H., Li, B., Tan, S., Huang, J., 2018. Detection of deep network generated images using disparities in color components. arXiv preprint arXiv: 1808.07276.

Marek. K., 2018. Deepfake project - nonofficial project based on original deepfakes thread. http://www. github.com/deepfakes/faceswap. Accessed 20 July 2025.

Marek Kowalski., 2019. FaceSwap development by creating an account. GitHub.

Matern, F., Riess, C., Stamminger, M., 2019. Exploiting visual artifacts to expose deepfakes and face manipulations, IEEE Winter Applications of Computer Vision Workshops (WACVW), 83–92.

McCloskey, S., Albright, M., 2018. Detecting gan generated imagery using color cues. arXiv preprint arXiv: 1812.08247.

Nhu, T.D., Na, I., Kim, S.H., 2018. Forensics face detection from gans using convolutional neural network.

Pandey, R.C., Singh, S.K., Shukla, K.K., 2014. Passive copymove forgery detection in videos, In 2014 International Conference on Computer and Communication Technology (ICCCT), pp 301–306.

Parker, Ashley, 2020. Trump and allies ramp up efforts to spread disinformation and fake news. The Independent. Accessed 9 April 2022.

Parkhi, O. M., Vedaldi, A., Zisserman, A., 2015. Deep Face Recognition, In Proceedings of the British Machine Vision Conference (BMVC). BMVA Press. https://doi.org/10.5244/C.29.41

Rahmouni, N., Nozick, V., Yamagishi, J., Echizen, I., 2017. Distinguishing computer graphics from natural images using convolution neural networks, In IEEE Workshop on Information Forensics and Security, WIFS 2017, Rennes, France.

*Correspondence: tgi@sjp.ac.lk

Reddit_User, 2017. A Reddit User Starts 'Deepfake', Eyerys. https://www.eyerys.com/articles/timeline/reddit-user-starts-deepfake#event-a-href-articles-timeline-eu-agrees-usb-c-should-be-the-standard-for-chargersthe-european-union-agrees-that-usb-c-should-be-the-standard-for-chargers-a

Rossler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J. & Niener, M., 2018. Face Forensics: A Large-scale Video Dataset for Forgery Detection in Human Faces.

Rossler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J. & Niener, M., 2019. FaceForensics++: Learning to Detect Manipulated Facial Images.

Sabir, E., Cheng, J., Jaiswal, A., AbdAlmageed, W., Masi, I., Natarajan, P. Recurrent Convolutional Strategies for Face Manipulation Detection in Videos. USC Information Sciences Institute, Marina del Rey, CA, USA.

Thies, J., Zollhofer, M., Stamminger, M., Theobalt, C. & Niessner, M., 2016. Face2face: Real-Time Face Capture and Reenactment of RGB Videos. pp 2387–2395.

Tolosana, R., Vera-Rodriguez, R., Fierrez, J. Morales, A., Ortega-Garcia, J., 2020. Deepfakes and beyond: a Survey of face manipulation and fake detection. Information Fusion, 64, 131–148.

Weerawardana, M.C., Fernando, T.G.I., 2021. Deepfakes Detection Methods: A Literature Survey, 10th International Conference on Information and Automation for Sustainability (ICIAfS), Sri Lanka, pp 76-81. doi: 10.1109/ICIAfS52090.2021.9606067.

Wu, Y., AbdAlmageed, W., Natarajan, P., 2018[a]. BusterNet: Detecting Copy-Move Image Forgery with Source/Target Localization, In ECCV, 168–184.

Wu, Y., AbdAlmageed, W., Natarajan, P., 2018[b]. Image copymove forgery detection via an end-to-end deep neural network, *In WACV*, 1907–1915.

Wu, Y., AbdAlmageed, W., Natarajan, P., 2019. ManTra-Net: Manipulation tracing network for detection and localization of image forgeries with anomalous features, *In CVPR*.

Zhang, X., Karaman, S., Chang, S., 2019. Detecting and simulating artifacts in GAN fake images. preprint arXiv: 1907.06515.

Zhou, P., Han, X., Morariu, V.I., Davis, L.S., 2017. Two-stream neural networks for tampered face detection, IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). IEEE, pp 1831–1839.

*Correspondence: tgi@sjp.ac.lk