Comparative Analysis of Deepfake Image Detection Method Using VGG16, VGG19 and ResNet50

Zahra Nazemi Ashani¹, Iszuanie Syafidza Che Ilias¹,*, Keng Yap Ng², Muhammad Rezal Kamel Ariffin³, Ahmad Dahari Jarno⁴, Nor Zarina Zamri⁴

¹ Institute for Mathematical Research, Universiti Putra Malaysia, Jalan Universiti 1, 43400 Serdang, Selangor, Malaysia
² Department of Software Engineering and Information System, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Jalan Universiti 1, 43400, Serdang, Selangor, Malaysia
³ Department of Mathematics, Faculty of Science, Universiti Putra Malaysia, Jalan Universiti 1, 42400, Serdang, Selangor, Malaysia
⁴ Malaysian Security Evaluation Facility, (MySEF), Department CyberSecurity Proactive Services Division, CyberSecurity Malaysia, Menara Cyber Axis, Jalan Impact, 63000 Cyberjaya, Selangor, Malaysia

ARTICLE INFO

ABSTRACT

In the rapidly evolving digital landscape, the proliferation of online image and video sharing has reached unprecedented levels. This surge has been accompanied by the emergence of deepfake technology, powered by generative adversarial networks and deep learning techniques, enabling the creation of highly realistic fabricated images and videos. Consequently, the widespread dissemination of deepfake content on social media platforms has prompted the global community to seek effective methods for detecting and combating this issue. Recognizing that Artificial Intelligence (AI) has played a significant role in propagating deepfake challenges, researchers and experts believe that AI can also provide viable solutions. In this study, we explore the application of AI for deepfake image detection. Specifically, we focus on three convolutional neural network (CNN) algorithms—VGG16, VGG19, and ResNet—for this purpose. To evaluate the performance of these CNN models, a dataset comprising 1,200 images, including a combination of fake and real images, was utilized. The deepfake images were generated using FaceApp, a prominent tool for creating manipulated visuals. Our findings demonstrate that VGG19 outperforms both VGG16 and ResNet50, achieving an impressive accuracy rate of 98% on the test dataset when small size is small. By harnessing the power of AI, particularly through the application of CNN algorithms, this research makes a significant contribution to the field of deepfake image detection. The results underscore the efficacy of VGG19 in accurately identifying manipulated images, thereby providing valuable insights for the development of robust detection systems to combat the proliferation of deepfakes in today’s digital world.

Keywords:
Deep fake image detection; convolutional neural network; VGG16; VGG19; ResNet

* Corresponding author.
E-mail address: iszuanieupm@gmail.com (Iszuanie Syafidza Che Ilias)

https://doi.org/10.37934/araset.47.1.1628
1. Introduction

In today’s digital landscape, the rise of deepfake technology presents a significant challenge. Deepfake involves overlaying one person’s face onto another person’s face, creating convincingly realistic manipulated videos and images [1]. This emerging subdomain of AI technology has gained widespread prominence, particularly across social media platforms. However, the rapid proliferation of deepfakes has resulted in the erosion of trust in media content. The consequences of deepfake manipulation are far-reaching. They have the potential to cause negative impacts on individuals who are targeted by such deceptive content. This is especially concerning in the current era, where the accessibility of deepfake creation tools has significantly increased, and social media enables the rapid dissemination of these fabricated contents. Deep learning, a subset of AI, serves as a fundamental element for generating deepfake images and videos. There exist various available tools, such as Facewap, Facewap-GAN, Few-Shot Face translation, DFaker, DaapFake_tf, and DiscoFaceGAN, that facilitate the creation of deepfakes using AI. As AI technology has played a role in the proliferation of deepfakes, there is a growing battle between those who generate deepfakes and those who strive to develop AI-based methods for detecting them. Consequently, the quality of deepfakes has been continuously improving, necessitating the advancement of corresponding detection techniques. The research community has recognized the urgency of addressing deepfake detection and has directed significant efforts toward the development of detection algorithms. Numerous studies have been conducted in this domain, focusing on enhancing the understanding of deepfake detection methods and exploring innovative approaches to tackle this pressing issue.

For instance, in 2018, Güera and Delp [2] proposed the temporal-aware pipeline method that uses CNN and long short-term memory (LSTM) to detect deepfake videos. An accuracy of 97% was obtained using a dataset of 600 videos. In 2019, Sabir et al., [3] leveraged the use of spatio-temporal features of video streams to detect deepfakes. They proposed recurrent convolutional model to exploit temporal discrepancies across frames. The proposed method was tested on 1000 videos and presented the promising results. Kim et al., [4] applied three CNN architecture which were ShallowNet, VGG-16 and Xception in order to detection deepfake images. The dataset which was used in this research was Disguised Faces in the Wild (DFW) 2018. The provided dataset was imbalanced. In this regard, data augmentation was applied to balance the dataset. All the models were trained based on the provided dataset. The results demonstrated that the Xception presented 62% accuracy which is better than other models. Indeed, this research suggested that deeper off-the-shelf (OTS) classifier with additional optimization is able to improve the performance for deepfake image detection. In 2020 and 2021 the number of studies which evaluate the application of AI for deepfake image detection increased significantly. Shad et al., [1] provided a comparative analysis of deepfake image detection method using CNN. In this research, eight different CNN models were used in order to compare the accuracy of the proposed models for deepfake detection. Five evaluation metrics were considered. Between all models, VGGFace performed the best with 99% accuracy. In 2021, Ismail et al., [5] proposed a new method for deepfake video detection using You Only Look Once (YOLO) and convolutional recurrent network (RNN). Indeed, they applied YOLO algorithm for objection detection, EfficientNet-B5 which is a fine-tuned CNN was used for extracting spatial features and bidirectional Long short-term memory to extract the temporal features. As a result, the experimental analysis approves the superiority of the proposed method compared to the state-of-the-art methods. It can be seen that most of the research which has been carried out for features extraction, used an extension of CNN. For example, in 2021, Bang and Lee proposed a combined CNN-GRU in order to obtain spatio-temporal feature. Indeed, they were interested in classification of electroencephalogram (EEG) signal. The EEG signal has a high dimension of feature
space, thereby, appropriate feature extraction methods are needed to improve classification performance. In this study, CNN is responsible for spatial feature extraction and GRU is responsible for temporal feature extraction. The proposed method, outperformed all other methods [6]. There are more researches which have been fulfilled in this area [7-13]. It can be clearly seen that in the most articles different CNN architectures were used for deepfake image detection. Moreover, various CNN architectures were used for extracting features at frame level followed by RNN when people were interested in deepfake video detection. In most cases, public datasets were used for training.

Despite these research endeavours, the comprehensive detection of deepfakes remains a complex challenge, requiring further investigation and advancements in the field. To contribute to this ongoing research, the present study aims to investigate deepfake image detection using AI when small sample size is used for training. Specifically, we focus on three prominent convolutional neural network (CNN) algorithms, namely VGG16, VGG19, and ResNet, for the purpose of deepfake detection. By analyzing the performance of these models on a dataset comprising 1,200 images, including both real and fake samples created using FaceApp, we aim to provide valuable insights into the effectiveness of these algorithms in accurately identifying deepfake content. Through our research, we aim to advance the current understanding of deepfake detection techniques and contribute to the development of robust and reliable methods to combat the proliferation of deepfakes. The structure of this article is as follows: Section 2 explains about the research significant. In section 3, material and methods for the proposed ML algorithms have been discussed. In section 4, the results have been presented and finally in section 4 there is a discussion which compares the result of this article with previous ones.

2. Methodology
2.1 Research Significant

The significance of this research lies in its unique approach to deepfake image detection, specifically focusing on the application of three CNN architectures: VGG16, VGG19, and ResNet50 when sample size for training is small. While previous literature predominantly utilized public datasets sets in Kaggle such as DFDC dataset, FFHQ, DFFD and Flickr for training and evaluation, this study aims to fill a gap by employing a dataset that has been specifically collected for this research [1,14,15]. One notable aspect of this study is the exploration of how these CNN architectures perform when trained on a comparatively smaller dataset, in contrast to the larger public datasets commonly used in prior research. This investigation allows us to understand the implications of dataset size on the accuracy and effectiveness of the deepfake detection algorithms. By analyzing the performance of VGG16, VGG19, and ResNet50 on this smaller collected dataset, we gain insights into the generalization capabilities and robustness of these models when faced with novel manipulation techniques. As mentioned earlier, previous studies have demonstrated that deepfake detection models often excel when tested on the same kind of manipulations they were trained on. However, their performance significantly declines when encountering novel manipulation techniques [16]. By focusing on similar manipulations within the collected dataset, we aim to analyze how the selected CNN architectures respond to these specific types of deepfake images. This analysis provides valuable insights into the models' adaptability and detection capabilities in real-world scenarios. In summary, this research stands out by employing the VGG16, VGG19, and ResNet50 CNN architectures on a dataset that is specifically collected for this study. By exploring the performance of these models on a smaller dataset and focusing on similar manipulations, we contribute to the understanding of the generalization capabilities of these algorithms. The findings of this research will shed light on the
implications of dataset size and the models' effectiveness, providing valuable insights for the development of deepfake detection systems in various contexts.

2.2 Material and Methods

The methodology employed in this study is outlined in Figure 1, which illustrates the framework guiding each step. The dataset utilized consists of a combination of real and fake images, totalling 1200 samples. This dataset comprises 600 fake images, generated using FaceApp, and 600 real images. Prior to training, all sample images underwent image pre-processing techniques to ensure optimal data quality. Three specific CNN architectures were selected for training and subsequently fine-tuned through a validation process, enabling the development of optimized models Figure 1.

Fig. 1. CNN architectures implementation framework
2.3 Data Source

FaceApp is a popular app and is in fact one of the first few apps to really popularise and democratize deepfakes and AI-generated face editing on smartphones. With FaceApp you can simply upload your picture to the app and then see what you’ll look like when you’re old, make yourself smile, and more. As mentioned above, the app uses AI to edit the photos, so they look quite realistic [17,18]. The sample size which has been considered in this research contains 1200 images, 600 images were real while another 600 images were fake and were generated by FaceApp.

2.4 Resize and Normalize

Image resizing and normalization are crucial pre-processing steps before feeding the image data into the CNN architecture. In this study, the dataset consists of 1200 image samples, each originally sized at 256x256 pixels. To ensure compatibility with the chosen CNN model, a standard resizing function was employed to reduce all images to 224x224 pixels, following the recommended approach outlined in previous researches [19-23]. Resizing the images offers several advantages. Firstly, it reduces the computational complexity during both the training and prediction phases, leading to faster model training and potentially improved performance. Furthermore, resizing helps eliminate noise and irrelevant details from the images, enhancing the model’s ability to focus on essential features [24]. Following image resizing, it is essential to normalize the pixel values to ensure they fall within a suitable range for the CNN models. One commonly adopted normalization method involves dividing the pixel values by 255, which corresponds to the maximum value representable by an 8-bit color channel [25]. This normalization procedure transforms the pixel values to a standardized range between 0 and 1, which is commonly used for normalizing image data. By resizing the images to a consistent size and normalizing the pixel values, we establish a uniform and standardized representation of the dataset. These pre-processing steps enable the CNN model to effectively learn and extract meaningful features from the images during the training process, contributing to accurate and reliable deepfake image detection.

2.5 Training and Testing Dataset

The process of splitting the dataset into distinct training and test sets is a crucial step in developing and evaluating machine learning models. By allocating a subset of the data for training, we can effectively fit the model parameters, while reserving the remaining data for assessing the model’s performance on unseen samples [26]. It is common practice to perform a random split to ensure unbiased representation in both sets [27]. In this study, the dataset consisting of a total of 1200 sample images were divided into a training set and a test set. The training set comprises 1000 images, accounting for approximately 83% of the dataset. Within the training set, 500 images were classified as fake, and the remaining 500 were categorized as real. This balanced distribution ensures that the model receives an equal representation of both fake and real images during the training phase. The remaining 200 images, accounting for approximately 17% of the dataset, were allocated to the test set. This independent set serves as an unseen dataset to evaluate the model’s performance on detecting deepfake images. It includes 100 fake images and 100 real images, maintaining the balance observed in the training set. By utilizing this split strategy, we ensure that the model is trained on a sufficiently large dataset, enabling it to learn and generalize effectively. The evaluation on the separate test set provides insights into the model’s ability to accurately classify and
differentiate between fake and real images, contributing to a comprehensive assessment of the deepfake detection algorithm [28].

2.6 CNN Algorithms

In this study, three pretrained CNN architectures using transfer learning were examined: VGG16, VGG19, and ResNet50. Transfer learning involves utilizing the weights of the pre-trained models' convolutional layers and training only the last layers with data from the newer classes [29]. There are literatures in various fields that have applied them for image classification [22,30-34]. Moreover, there are some studies that compared the accuracy of the mentioned models [31,34]. It has been shown that ResNet50 achieves higher accuracy than VGG19 and VGG16. In this section, a brief description of the mentioned CNN architectures has been provided.

2.6.1 VGG16

VGG16 is a CNN architecture introduced in 2014 through a competition. It consists of 16 layers, with 13 convolutional layers, five max pooling layers, and three dense layers. The network utilizes 3x3 filters with a stride of 1 in the convolutional layers and 2x2 max pooling filters with a stride of 2. VGG16 is known for its simplicity, as it does not have an excessive number of hyperparameters. It has approximately 138 million parameters and achieves an accuracy of 92.7% when classifying 1000 different categories. VGG16 is particularly suitable for transfer learning and accepts input tensors of size 224x224 with 3 RGB channels.

2.6.2 VGG19

VGG19 follows a similar concept to VGG16 but includes 19 weighted layers. It possesses additional convolutional layers, enabling the model to capture more complex features. The architecture of VGG19 can be summarized as follows: 2Conv - 1Maxpool - 2Conv - 1Maxpool - 4Conv - 1Maxpool - 4Conv - 1Maxpool - 4Conv - 1Maxpool - 1FC - 1FC - 1FC.

2.6.3 ResNet

ResNet, introduced in 2015, has gained significant success in various competitions and tasks. It achieved first place in the ILSVRC 2015 classification competition and the COCO 2015 competitions in ImageNet detection, localization, and segmentation. ResNet addresses the challenge of vanishing gradients in deep CNNs through the use of residual connections. These connections allow for the training of very deep networks with hundreds or even thousands of layers. Notably, ResNet has demonstrated superior performance compared to VGG16 and VGG19 [35,36]. It offers efficient training of networks with 100 or 1000 layers.

2.7 Model Evaluation

The evaluation of classification models in this study involved analyzing their performance based on recall, precision, accuracy, and F1 score. These metrics provide insights into different aspects of the model's predictive capabilities. In order to calculate the mentioned metrics, it is required to calculate confusion matrix. A confusion matrix is an N×N matrix used for evaluating the performance of a classification model. N is presenting the number of target classes. The matrix presents the
number of True and False prediction by machine learning model. Indeed, it renders a holistic view of how well the classification model is performing and what kind of errors is making. For a binary classification, the confusion matrix is as follows.

In the Table 1 below:
(a) TP represents the number of positive samples correctly predicted.
(b) TN represents the number of negative samples correctly predicted.
(c) FP represents the number of negative samples wrongly predicted as positive.
(d) FN represents the number of positive samples wrongly predicted as negative.

<table>
<thead>
<tr>
<th>Predicted Values</th>
<th>Actual Values</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Positive (1)</td>
</tr>
<tr>
<td>Positive (1)</td>
<td>TP</td>
</tr>
<tr>
<td>Negative (0)</td>
<td>FN</td>
</tr>
</tbody>
</table>

Table 1
Confusion Matrix for binary classification

Therefore, the definitions for mentioned evaluation metrics are as follows.
Recall, also known as sensitivity, measures the proportion of correctly predicted positive observations out of all the actual positive observations (Eq. (1)). It indicates how well the model identifies positive instances.

Recall = \( \frac{TP}{TP+FN} \)  

(1)

Precision, on the other hand, quantifies the ratio of correctly predicted positive observations to all predicted positive observations (Eq. (2)). It assesses the accuracy of the model's positive predictions, indicating the extent to which positive predictions are reliable.

Precision = \( \frac{TP}{TP+FP} \)  

(2)

Accuracy represents the overall correctness of the model's predictions, calculated as the ratio of correctly predicted observations to the total number of observations (Eq. (3)). It provides a general measure of the model's performance in classifying both positive and negative instances.

Accuracy = \( \frac{(TP+TN)}{(TP+FP+TN+FN)} \)  

(3)

The F1 score is a combined metric that considers both precision and recall, providing a single value that reflects the model's performance (Eq. (4)). It is the weighted average of precision and recall, taking into account both the model’s ability to make accurate positive predictions and its ability to capture positive instances effectively.

F1 score = \( \frac{2}{1/Precision + 1/Sensitivity} \)  

(4)

The results for confusion matrix for three CNN models are presented in Table 2, Table 3 and Table 4.
For VGG-16:

Table 2
Confusion Matrix for VGG-16

<table>
<thead>
<tr>
<th>Predicted Values</th>
<th>Actual Values</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Positive (1)</td>
<td>Negative (0)</td>
<td></td>
</tr>
<tr>
<td>Positive (1)</td>
<td>97</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Negative (0)</td>
<td>9</td>
<td>91</td>
<td></td>
</tr>
</tbody>
</table>

and for VGG-19:

Table 3
Confusion Matrix for VGG-19

<table>
<thead>
<tr>
<th>Predicted Values</th>
<th>Actual Values</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Positive (1)</td>
<td>Negative (0)</td>
<td></td>
</tr>
<tr>
<td>Positive (1)</td>
<td>100</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Negative (0)</td>
<td>3</td>
<td>97</td>
<td></td>
</tr>
</tbody>
</table>

and for ResNet50:

Table 4
Confusion Matrix for ResNet50

<table>
<thead>
<tr>
<th>Predicted Values</th>
<th>Actual Values</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Positive (1)</td>
<td>Negative (0)</td>
<td></td>
</tr>
<tr>
<td>Positive (1)</td>
<td>84</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>Negative (0)</td>
<td>0</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

The classification models' performance for three various models in terms of these metrics is summarized in Table 5. Table 5 presents the recall, precision, accuracy, and F1 score values for each model.

Table 5
Evaluation Metrics for VGG-16, VGG-19 and ResNet50

<table>
<thead>
<tr>
<th></th>
<th>Recall</th>
<th>Precision</th>
<th>Accuracy</th>
<th>F1-Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG-16</td>
<td>0.9151</td>
<td>0.97</td>
<td>0.94</td>
<td>0.9547</td>
</tr>
<tr>
<td>VGG-19</td>
<td>0.9708</td>
<td>1</td>
<td>0.985</td>
<td>0.9924</td>
</tr>
<tr>
<td>ResNet50</td>
<td>1</td>
<td>0.84</td>
<td>0.92</td>
<td>0.8782</td>
</tr>
</tbody>
</table>

These performance metrics allow us to assess the effectiveness and reliability of the classification models in accurately identifying and classifying instances in the dataset.

3. Result

The study aimed to detect deepfake images using three CNN architectures: VGG16, VGG19, and ResNet50. The dataset consisted of 1200 images, including 600 fake images generated using FaceApp and 600 real images. Among the dataset, 1000 images (500 real and 500 fake) were allocated for training, while 200 images (100 real and 100 fake) were used for testing. After training the VGG16, VGG19, and ResNet50 architectures on the training dataset, their performance was evaluated using the test dataset. The accuracy results showed that VGG16 achieved 94% accuracy, VGG19 achieved 98.5% accuracy, and ResNet50 achieved 92% accuracy. Notably, VGG19 demonstrated higher
accuracy compared to both VGG16 and ResNet50. These findings slightly differ from previous study by Mascaren and Agarwal [35] in 2021, where they compared the accuracy of VGG16, VGG19, and ResNet50 (They explored that the accuracy of ResNet50 was higher than VGG19 and VGG16). Their research utilized a real dataset obtained from a regional retailer.

The number of parameters in ResNet50 architecture is 445,951,874. Therefore, training ResNet50 required more computations, resulting in increased training time and energy consumption. VGG19, approximately has 139,578,434 parameters, and VGG16 has with 134,268,738 parameters. The key difference between VGG16 and VGG19 is the inclusion of three additional convolutional layers in VGG19, enabling it to capture more complex features. In this study, ResNet50 exhibited lower accuracy than both VGG19 and VGG16. This can be attributed to the increased complexity of ResNet50, which requires a larger dataset to achieve higher accuracy. The limited sample size used in this study might have affected the performance of ResNet50 compared to the other two models. Overall, the results indicate that VGG19 outperformed both VGG16 and ResNet50 in terms of accuracy for deepfake image detection. Further investigations with larger datasets could provide additional insights into the performance of these CNN architectures for deepfake detection.

The accuracy of the provided models was 98.5%, 94% and 92% for VGG19, VGG16 and ResNet50 respectively. During the last few years, the number of publications for deepfake image detection using AI has dramatically increased. They have introduced many different AI method for deepfake image detection. However, plenty of research demonstrated that detectors trained on image produced by one deepfake model perform poorly when tested on others. In this regard, in 2020, Akhtar et al., [16] fulfilled interesting research related to this issue. They were interested in deepfake detection using CNN. They employed smartphone FaceApp with 11 different filters for creating fake images. They considered three strategies which are as follows:

(i) The first strategy involved training the models on a single face manipulation type and evaluating their performance on the same manipulation type. The accuracy of deepfake detection for all methods in this case ranged from 81.56% to 99.31%. This demonstrates the models' capability to detect deepfakes accurately within the specific manipulation type.

(ii) The second strategy encompassed training the models on multiple face manipulation types and testing them on all manipulation types. The accuracy achieved in this scenario ranged from 92.17% to 99.42%, indicating that the models exhibited robustness when faced with various types of manipulations.

(iii) In third strategy, the models were trained on one manipulation type and tested on images generated using a different manipulation type. The results showed a lower accuracy range of 56.61% to 64.91%, emphasizing the difficulty of detecting deepfakes when confronted with novel manipulation types.

In order to evaluate our model using the result from Akhtar et al., [16], generated fake images using FaceApp. FaceApp is able to generate face according to various features such as age, hairstyles, face sizes, and others. In this study, the fake images were generated by FaceApp by changing the age, hairstyles, smiles, and/or, impression. Out of 100, 20 images will remain as real images and mixed in with the fake images. Three mentioned CNN models were used. The result presented that VGG16 for deepfake image detection exhibited the highest accuracy which is 40%, followed by VGG19, 34% and lastly ResNet50 30% (Figure 2). All the accuracy are below that the previous threshold. The decrease in accuracy observed in third strategy indicates the challenges posed by unseen manipulation types and highlights the need for ongoing research and model refinement.
In recent years, the utilization of images has extended across diverse fields, as evidenced by previous studies [37-39]. The imperative significance of distinguishing between fake and real images has concurrently escalated. In this study, the performance of three CNN architectures, VGG16, VGG19, and ResNet50, was evaluated using small sample size for the detection of deepfake images. The results indicated that VGG19 achieved the highest accuracy of 98.5%, followed by VGG16 with 94% accuracy, and ResNet50 with 92% accuracy. These findings were slightly different from previous study by Mascaren and Agarwal [35], which also compared the accuracy of these models. ResNet50, despite having the highest number of parameters (445,951,874), exhibited lower accuracy compared to VGG16 and VGG19. This can be attributed to the increased complexity of ResNet50, which may require a larger dataset to achieve higher accuracy. The limited sample size used in this study might have affected the performance of ResNet50 compared to the other two models. The key difference between VGG16 and VGG19 lies in the inclusion of three additional convolutional layers in VGG19, allowing it to capture more complex features. As a result, VGG19 outperformed both VGG16 and ResNet50 in terms of accuracy for deepfake image detection.

It is important to note that detecting deepfake images across different manipulation types remains a significant challenge. The evaluation of the models using the results from Akhtar et al.’s [16] study demonstrated that the accuracy of the models decreased when faced with novel manipulation types. This emphasizes the need for ongoing research and model refinement to address the challenges posed by unseen manipulation types and improve detection accuracy. In conclusion, while VGG19 demonstrated the highest accuracy among the tested models, there is still room for improvement in deepfake image detection. Further investigations using larger datasets, encompassing diverse manipulation types, and exploring advanced AI techniques are necessary to enhance the robustness and reliability of CNN architectures for deepfake detection in real-world scenarios.

**Acknowledgement**

This work has been supported by Cyber Security Malaysia (CSM) through collaboration project of CSM-INSPEM, UPM under Test Methods for Deepfakes Facial Image Detection Using Artificial Intelligence (AI) project parked under Tabung Amanah JIMN INSPEM (vot.no.: 6233205-10065).
References


on Computer Vision and Pattern Recognition, pp. 2658-2667. 2016. [20]


