IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 27 June 2024, accepted 18 July 2024, date of publication 25 July 2024, date of current version 5 August 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3433497

RESEARCH ARTICLE

Convolutional Neural Network for Detecting Deepfake Palmprint Images

TSAI MIN-JEN[1], (Member, IEEE), AND CHANG CHENG-TAO[1,2]
[1]Institute of Information Management, National Yang Ming Chiao Tung University, Hsinchu 30010, Taiwan
[2]Broadband Networks Laboratory, Chunghwa Telecom Laboratories, Taoyuan City 30010, Taiwan

Corresponding author: Tsai Min-Jen (mjtsai@nycu.edu.tw)

This work was supported in part by the National Science Council of Taiwan, China, under Grant NSTC 112-2410-H-A49-024 and Grant NSTC 113-2410-H-A49-062-MY2.

ABSTRACT With the rapid advancement of computer vision technology, various deepfake tools for generating deceptive images have emerged. Generative Adversarial Networks (GANs) can create various deceptive media streams, including images, audio, and video, leading to numerous societal challenges. Palmprint recognition technology has recently been applied in financial identity verification, particularly in confirming transactions across various banking platforms. Manipulating critical financial transactions or generating malicious images to deceive authentication processes can result in significant disruptions. Convolutional Neural Networks (CNNs) are considered practical tools. We propose the implementation of a Dual Cascade Convolutional Neural Network (DC-CNN) algorithm that utilizes a dual-channel technique. This approach involves two networks that train one subnetwork and then apply the same configuration to the other. The feature vectors are combined, the fake inputs can be identified. This dual-channel technique is particularly effective for detecting forged images. Our approach involves comparing various CNN architectures, such as MesoNet, MesoInceptionNet, and Dense CNN (D-CNN), within the framework of GAN methods, such as Wasserstein GAN (WGAN) and Cycle GAN. In our experiments, DC-CNN demonstrates favorable results in detecting fake palmprints based on WGAN and cycle GAN. Specifically, for WGAN-based fake palmprints, the model achieved the weighted precision of 90.83%, weighted recall of 90.20%, weighted F1 scores of 89.92% and accuracy of 90.20%. In the case of Cycle GAN-based fake palmprints, the model exhibited the weighted precision of 87.86%, weighted recall of 87.91%, weighted F1 scores of 87.85% and accuracy of 87.91%. Therefore, DC-CNN emerges as a promising approach in the fields of deepfake palmprint detection and identity verification.

INDEX TERMS Deepfake detection, generative adversarial networks, Wasserstein GAN, cycle GAN, convolutional neural network.

I. INTRODUCTION

The human palm exhibits numerous patterns, including principal lines and wrinkles. Palm patterns are unrelated to genetics and remain stable despite external environmental factors. Once formed, palm patterns remain consistent, and they do not change due to external influences. The main components of palm patterns are classified into three types: the lifeline, the headline, and the heartline. These three significant lines are collectively referred to as principal lines. Please refer to **Figure 1.** for an illustration of palm patterns.

Many studies discuss methods for selecting Regions of Interest (ROI). Connie et al. [2]. suggested the most effective method of surrounding the palm with an ellipse. Following the separation of the palm from the background, the authors utilized the ellipse's major axis to align the palm. The ellipse's center was the reference point for image segmenting to extract the ROI. A notable characteristic of this approach is its necessity for capturing the entire palm depth, including well-spaced and fully extended fingers. By conducting

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Asif.

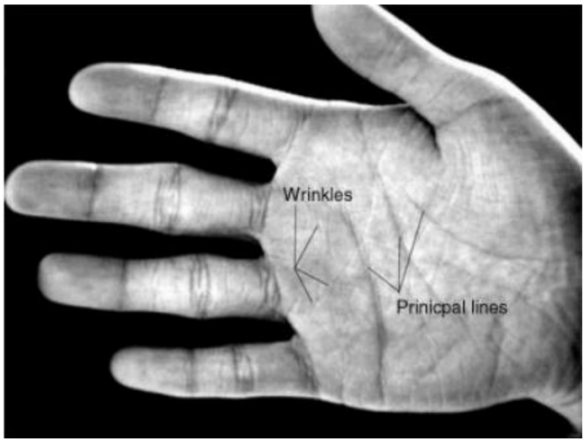

FIGURE 1. The palm pattern diagram [1].

calculations, the orientation of the palm can be automatically calibrated. One standard method for authentic palmprint recognition was introduced by Kadam and Deshmane [3], which presents a reliable palmprint extraction method known as Local Binary Pattern (LBP). LBP is well-suited for non-contact palmprint recognition.

GANs constitute a pivotal deep learning model, primarily composed of a generator and a discriminator. The core objective of this model is the continuous generation of synthetic data, challenging the discriminator to distinguish between real and fake data. Through adversarial training, these components refine their capabilities, optimizing until the generator achieves the production of high-quality synthetic data. Simultaneously, the discriminator evolves to identify fake data with heightened accuracy. This intricate interplay results in a generator proficient in generating realistic synthetic data, blurring the lines for the discriminator's distinction between natural and synthetic. Please refer to **Figure 2.** for an illustration of GANs.

Initially presented by Goodfellow and collaborators in 2014 [4], GANs streamline the generation of convincing synthetic images, audio, and video content, mimicking authenticity. Palm patterns persist unchanged despite external influences, maintaining their consistency once established.

Minaee et al. [5] employed GANs to produce synthetic palmprint data. This study adopts an approach as the foundation for our experiments. We highlight the crucial contributions of WGAN [6] and Cycle GAN [7] in our research.

The use of the Wasserstein distance as a loss function for Generative Adversarial Networks (GANs) is discussed in WGAN [6]. The challenge of directly computing the infimum across all joint distributions by applying the Kantorovich-Rubinstein duality is addressed. This mathematical transformation simplifies the computation to a more manageable formula:

$$W\left(P_r, P_g\right) = \frac{1}{k} SUP_{||f||L \leq K}[E_{x \sim P_r}[f(x)] - E_{x \sim P_g}[f(x)]]$$

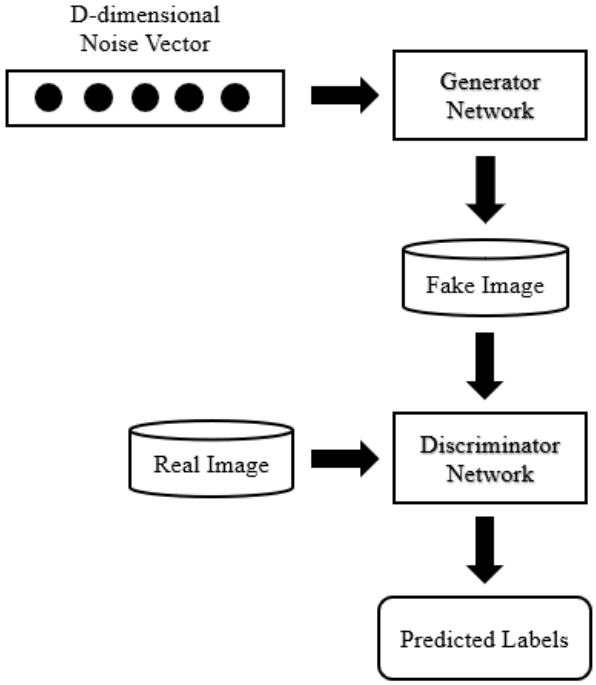

FIGURE 2. An architecture diagram of a GAN.

Here, the supremum (maximum value) is calculated over all functions f that are K-Lipschitz. This approach provides a stable method for measuring the distance between the real data distribution P_r and the generated data distribution P_g, enhancing GAN training and preventing common issues like mode collapse.

The Cycle GAN [7] architecture comprises two generators and two discriminators. This architecture ensures that an image transformed by generator G, then back-transformed by generator F remains close to the original. The cycle consistency loss is formulated as follows:

$$L_{cyc}(G, F) = [E_{x \sim P_{data}(x)}[||F(G(x)) - x||] - E_{y \sim P_{data}(y)}[||G(F(y) - y||]$$

This loss measures the norm between original and reconstructed images, promoting content fidelity and transformation reversibility, which is crucial for unsupervised learning and style transfer tasks without paired data. The generator aims to translate images from one domain to another. One set is responsible for converting images from one style to another, while the other reverses this transformation.

To tackle the challenges presented by DF GANs, there is a pressing need for a resilient and widely applicable DF detection system. An effective DF detection system must accurately discern manipulated and synthetic content from genuine content. Recent literature underscores the development of a robust DF detection framework. However, many existing approaches need more robustness and efficacy in training DF detection models and integrating generalizability and interpretability into the model. Zhang et al. [8] utilized

the Speeded Up Robust Features (SURF) method to pinpoint critical elements for a solution. Yu et al. [9] have revealed that current detection methods still need to improve for application in real-world scenarios, stressing the importance of further research focusing on generalization and robustness. Heo et al. [10] introduce a vision transformer model incorporating a distillation approach tailored for counterfeit video detection.

CNN has become a widely used technique in detecting deepfakes (DF), as numerous recent research papers have shown. Initially, CNN models undergo pre-training on individual frames, with these methods involving decision-making based on grouped data. However, many CNN approaches function like black boxes, leading to overfitting issues in the models and resulting in varied interpretations of the same data under different conditions. Thus, the approach that Patel et al. [11] proposed with D-CNN offers a promising solution to this challenge. Consequently, given the current application of palmprint recognition in financial identity, if there were deliberate attempts in the future to generate a large volume of fake palmprints to disrupt financial systems, could an effective method be devised to detect them? To address this question, a DC-CNN is introduced to enhance the recognition capabilities of D-CNN across various domains of GANs.

A. NOVELTY

MesoNet and MesoInceptionNet [12] are renowned CNN models for their outstanding performance in DF. Although D-CNN offers a viable solution for enhancing model generalizability, these detection methods are specifically designed to evaluate against commonly used deepfake detection databases, addressing societal issues that have already transpired. However, there needs to be a specific assessment method for the forgery of other biometric features, such as palmprints. It is challenging to anticipate which GANs malicious actors may utilize for forgery, but we can improve our accuracy by leveraging the strengths of various algorithms. Therefore, in this study, we initially partition the images into high-frequency and low-frequency regions, assigning them to separate channels in D-CNN. Subsequently, we merge these two channels to enhance accuracy.

B. RESEARCH CONTRIBUTION

The primary contributions of this paper include:

1. Employing a CNN model to identify palmprint features created by DF offers a viable approach to address potential societal concerns in the future.
2. We are employing diverse methods to produce different types of fake palmprints, with DC-CNN serving as the foundational model for detection and training to improve generalization.
3. It simultaneously compares multiple model architectures to assess performance using accuracy, precision, recall, and F1 score metrics.

C. ARTICLE LAYOUT

The paper's structure is organized as follows: Section II explores the current methodologies of DF detection models. Section III explains the proposed model approach and systematically describes the model processing. Section IV discusses the performance evaluation of the model. Moving to Section V, we provide a discourse on the proposed scheme's future challenges. Finally, Section VI concludes the work, providing insights into future directions.

II. RELATED WORK

The palmprint recognition systems' absolute image recognition methods encompass a range of strategies, beginning with identifying the palmprint's ROI and analyzing its physiological characteristics. Various principles are employed to define the ROI [13]. One prevalent rule for locating the ROI involves establishing a coordinate system based on the gaps between fingers, which is the effective research starting point. These defined ROIs lay the foundation for subsequent signal-level feature extraction, offering a comprehensive basis for analysis.

The Scale-Invariant Feature Transform (SIFT) was initially introduced in [14] for object classification purposes and has been proposed as one of the signal-level feature extraction methods, as suggested by Wu [15], showing remarkable efficacy in contactless palmprint recognition. SIFT-based features exhibit invariance to image scaling, rotation, and partial invariance to changes in projection and illumination. Thus, leveraging SIFT features for detecting contactless palmprint images is highly suitable. Subsequent signal-level research has achieved notable advancements, such as using RANSAC and local palmprint descriptor distance to eliminate mismatched SIFT points. These techniques can significantly enhance the accuracy of the final matched SIFT points, considered the matching score of two samples in decision-making processes.

In the work by Zhao et al. [16], matched SIFT points were utilized to align palmprint images, and competitive codes were extracted based on this alignment. Integrating the matching scores of SIFT descriptors and competitive code planes enhances the accuracy of contactless palmprint verification. Furthermore, SIFT can be integrated with other local orientation descriptors. Notably, in signal-level research [17], these methods have made considerable advancements.

Contactless palmprint images often suffer from severe misalignment and noise, posing challenges for traditional signal-level research methods in achieving comparable performance in palmprint recognition. Moreover, deep learning methods, including AlexNet [18], VGG-16 [19], Inception-V3 [20], and ResNet-50 [21], have emerged as pivotal directions in real-world applications. As a result, palmprint recognition is transitioning towards a data-driven approach. Liu and Sun [22] proposed a palmprint recognition method. Initially, palmprint images undergo preprocessing using an enhanced fuzzy enhancement algorithm.

Subsequently, feature extraction is performed utilizing AlexNet, which boasts a network structure with eight layers.

The increasing focus on data-driven approaches has intensified the research on image recognition, with CNN architectures being mainly used for various applications. Wang et al. [23] enhanced palmprint recognition with a dense hybrid attention network for image super-resolution, while Li et al. [24] developed WaveletKernelNet to improve fault detection in mechanical systems. Santoso and Finn [25] used CNN-based models to differentiate between legitimate and malicious actions in robotic systems, aiming for high accuracy in detecting threats. Additionally, the challenge of obtaining specific data, especially in imbalanced datasets with minimal training data, has led to the adoption of Siamese neural networks [26], [27], [28]. These networks, which consist of two identical sub-networks processing separate inputs, facilitate advanced feature transformation and extraction, mapping inputs to a new feature space and improving the classification performance under challenging conditions.

As discussed in the paper, prior studies have predominantly emphasized the accuracy of signal feature methods or data-driven models using authentic images. However, with the advancement of GAN technology, a substantial amount of synthetic data has been created. Hence, we introduce a DC-CNN model that integrates signal features and data-driven methods to detect and evaluate fake palmprint data.

III. PROBLEM FORMULATION

The model consists of a generator network G and a discriminator network D. Let G represent the generator function that maps a latent space vector z sampled from a noise distribution to an image space. R refers to a three-dimensional real space, where H, W, and C represent an image's height, width, and number of channels as (1).

$$G : R^d \rightarrow R^{H \times W \times C} \tag{1}$$

Let D represent the discriminator function that maps an image to the probability of that image being real as (2).

$$D : R^{H \times W \times C} \rightarrow [0, 1] \tag{2}$$

The combined GAN model can be defined as a composite function where D is composed of G. This can be defined as (3).

$$GAN\ (z) : D(G(z)) \tag{3}$$

Given an input vector z, the generated image I by $GAN(z)$ is defined as (4)

$$I_{generated} = GAN(z) \tag{4}$$

The DC-CNN functions as F_{DC-CNN}, where some series of operations could be a series of convolutional layers, activation functions, pooling layers, etc., depending on the specifics of the network. This involves applying the model to rounding the probability to produce a binary label as (5). L is the probability that the given image is I, where $0 \leq L \leq 1$.

$$L = F_{DC-CNN}(I_{generated}) \tag{5}$$

Palmprint images serve as valuable biometric data, but are susceptible to environmental elements like dirt, sweat, and external factors, which can obscure the intricate patterns and ridges within the palmprint, making accurate identification and classification challenging. We leverage an advanced CNN tailored to extract two distinctive information channels from the palmprint image to defeat this obstacle. This methodology efficiently minimizes interference from the surroundings and improves the visibility of the palmprint's unique features. Algorithm 1 provides a comprehensive overview of the sequence of operations within the proposed system architecture. The process begins by taking input image data and directing it to the build_gan function, which involves the collaboration of the generator and discriminator to generate a synthetic image model.

Algorithm 1 Flow of Proposed Method

```
INPUT: noise_vector
OUTPUT: L (label), P (predicted_label)

FUNCTION build_gan(noise_vector):
    discriminator.trainable = False
    CREATE model using Sequential()
    ADD generator to model
    ADD discriminator to model
    GENERATE II_generated using the model on the noise vector
RETURN II_generated

FUNCTION DC_CNN(input_image, I_generated):
    LL = model.predict(input_image)
    PP = round(LL)
    RETURN LL, PP

I_generated = build_gan(noise_vector)
L, P = DC_CNN (image, I_ generated)
DISPLAY P
```

This synthesized image data, combined with the authentic data, forms a dataset which serves as a parameter for the deepfake function input. The overall outcome contributes to a comprehensive predictive image of the probability of detecting deepfake content.

The process begins with an input image aimed to classify it as real or fake and to estimate the model's confidence in its prediction. Initially, a GAN is utilized, configured through the build_gan function. In this setup, the discriminator is made non-trainable, meaning its weights remain fixed during training. The generator, which ideally takes a noise vector, is first added to a new model constructed using the sequential method, followed by the non-trainable discriminator. This model then generates a new image, II_generated. Subsequently, the original input image is used in the DC_CNN function to predict the authenticity. The DC_CNN model outputs a continuous probability (LL), which is then rounded to yield the final label (PP). The entire process involves

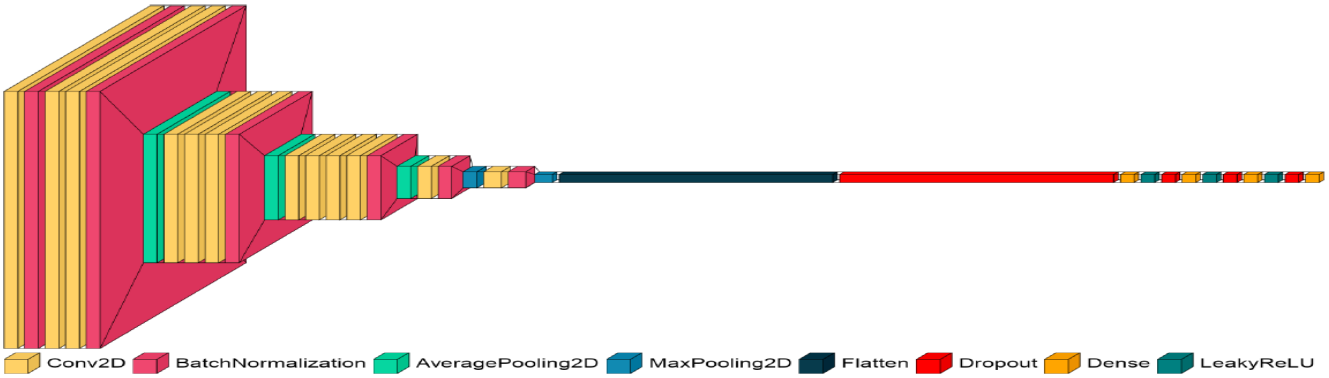

FIGURE 3. Architectural diagram of D-CNN [11].

generating I_generated via build_gan, and then using only the original image to determine the label (L) and the predicted confidence (P) by DC_CNN. Finally, the confidence level (P) is displayed, reflecting the likelihood of the image being real.

IV. PROPOSED METHOD

As mentioned in the third section, the model's architecture follows a binary classification approach, where the image input source is processed through GAN and separated into two channels by CNN. These two channels weaken the input source, which is processed through GAN and separated into two channels by CNN. These two channels undermine the impact of environmental interference. As CNN base is a suitable choice for DF detection, each channel enters its respective D-CNN base pathway, as shown in **Figure 3.**

This architecture is based on CNN and D-CNN. The schematic representation of this proposed architecture is depicted in **Figure 4.** The DC-CNN within the framework includes one CNN and a D-CNN module composed of (a) to (k) blocks. The processes ultimately pass through a fully connected layer (l) in which all outputs for the classification prediction are integrated.

In the first block (a), a 2D convolution is performed on the input using 3×3 filters with LeakyReLU as the activation function. Since this layer initiates high-level feature extraction, a smaller filter size of 3×3 is preferred over larger filters (such as 5×5 or 7×7). Batch normalization is then applied.

The second block (b) continues with two 2D convolutional layers, each using 3×3 filters and LeakyReLU for activation. Batch normalization follows each convolutional layer, standardizing the feature maps' output, stabilizing model training, enhancing adaptability to diverse inputs, and accelerating learning. A pooling layer (max or average pooling) reduces the spatial dimensions of the feature maps, lowering the computational load while preserving crucial feature information and preventing overfitting.

The third block (c) includes a neural network module with three convolutional layers, each followed by LeakyReLU activation. Each layer is configured with "convolutional 2D $32 \times (3 \times 3)$," indicating 32 filters of size 3×3 per layer. LeakyReLU helps to maintain the flow of negative gradients, improving the learning process.

The fourth block (d) comprises four convolutional layers, each with 64 filters of size 3×3. This configuration captures more complex and subtle feature variations, improving the network's recognition of input details. Each layer is followed by LeakyReLU activation and batch normalization to standardize output, reduce internal covariate shift, speed up training, and enhance generalization. A 2×2 window average pooling layer follows, retaining key features.

The fifth block (e) starts with a 2D convolutional layer using 128 filters of size 5×5 and LeakyReLU for activation, followed by batch normalization to stabilize and improve training efficiency. The block ends with a 2×2 max pooling layer, reducing spatial dimensions and focusing on essential features.

The sixth block (f) has a similar structure but with increased complexity. It begins with a 2D convolutional layer containing 256 filters of size 5×5, using LeakyReLU activation. This is followed by batch normalization and a 2×2 max pooling layer. The output dimensions are set to $8 \times 8 \times 256$, further reducing spatial dimensions, while increasing the depth to capture more detailed features.

The seventh block (g) includes a flattened layer, which transforms the multi-dimensional input into a one-dimensional vector for dense processing. The eighth block (h) features a Dense layer with 32 units and LeakyReLU activation, introducing non-linearity and preventing vanishing gradients. A Dropout layer is included to deactivate a fraction of the neurons during training, preventing overfitting and promoting generalization.

The ninth (i) and tenth (j) blocks also contain dense layers with 16 units each, utilizing LeakyReLU activation and including Dropout layers for the same reasons as the previous block. The eleventh block (k) consists of a sigmoid activation layer, converting inputs to values between 0 and 1, ideal for binary classification tasks. The twelfth block (l) features a fully connected layer that gathers outputs from the sigmoid

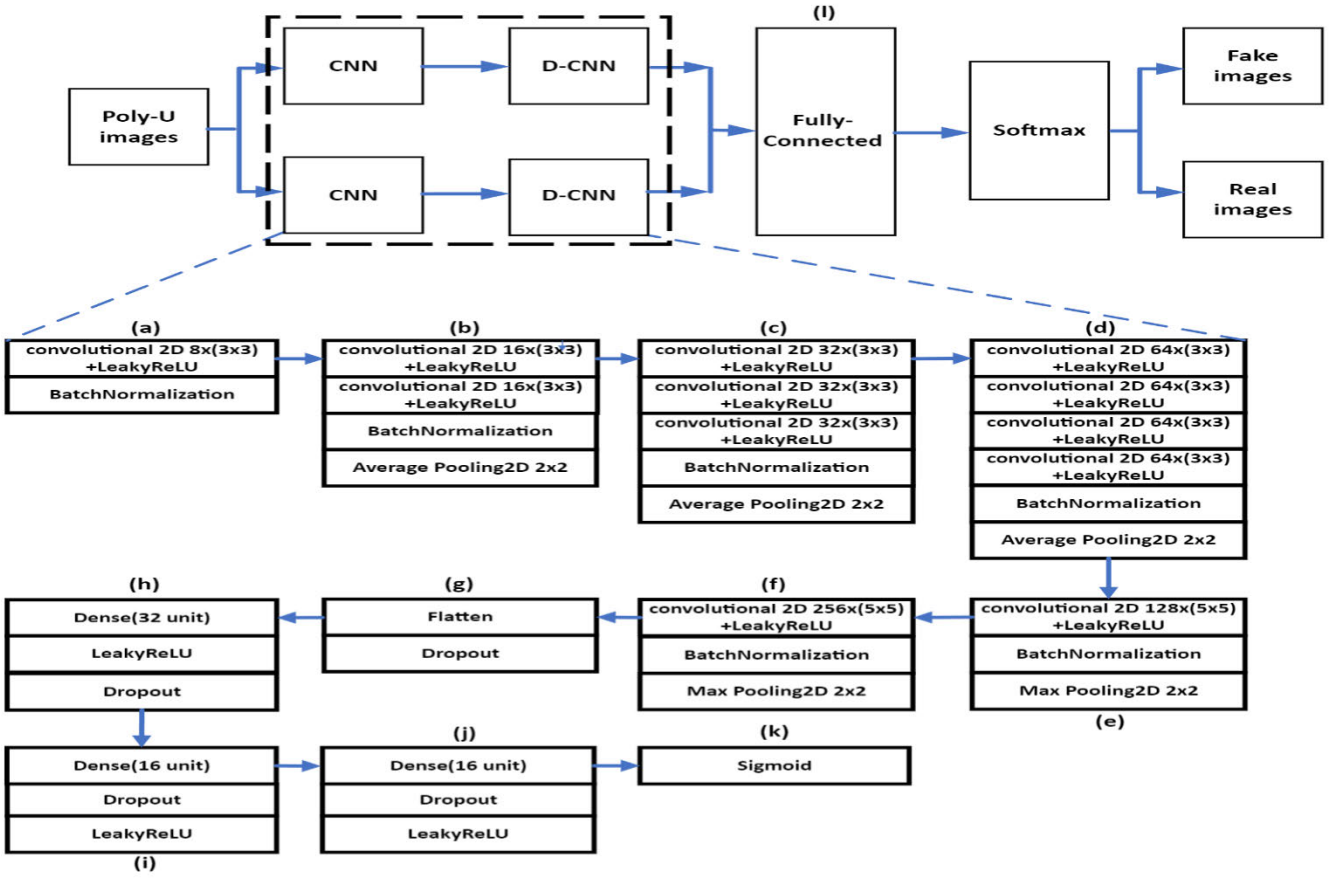

FIGURE 4. An architecture diagram of DC-CNN.

layer to compute the final classification result based on the processed features.

Images were sourced from the POLY-U database, followed by applying GAN methods. The proposed architecture, illustrated in **Figure 5**, consists of several layers. The first layer generates fake image layers using selected approaches such as WGAN GAN and Cycle GAN, which transform authentic images into fake ones. The second layer stores both real and fake photos in their respective databases. These synthetic palmprints are integrated into a database containing fake palmprints. Subsequently, the third layer utilizes various prediction methods to assess the model's capability to discern between genuine and forged images.

V. RESULTS AND DISCUSSION

This section explores the implications of generating synthetic palmprint images using various GAN techniques, utilizing samples extracted from the Poly-U palmprint database. Specifically, it investigates the outcomes of applying WGAN and Cycle-GAN to create these synthetic images.

A. SETUP

This study used Anaconda for development. This is an environment tailored for data science and machine learning tasks. the NVIDIA Geforce RTX 3070 GPU is applied for deep learning tasks throughout this study, assisting both model training and inference processes. Anaconda also provides a rich set of tools and libraries, such as TensorFlow, which enables developers to efficiently perform data analyses, modeling, and experimentation.

B. DATASET DESCRIPTION

A subset of the Poly-U palmprint database, mainly the left-hand palmprint images, is analyzed in this study. This subset is comprised of 1301 images, serving as our primary dataset for synthesis. Using this dataset, 2210 deepfake images are generated by WGAN and 2053 deepfake images by Cycle GAN, as detailed in TABLE 1.

We divided the image dataset into three subsets: training, validation, and testing sets. Specifically, 60% of the images were allocated for training, 20% for validation, and 20% for testing, making a total of 1301 real images. Therefore, 780 real images are used for training, approximately 261 images for validation, and 260 images for testing.

All 2210 images in the WGAN test dataset are fake. We utilized 60% of the images from the WGAN images (1323 images), for training. We selected 20% of the fake images (443 images) for the validation set, and allocated the remaining 20% (444 images) for the test set.

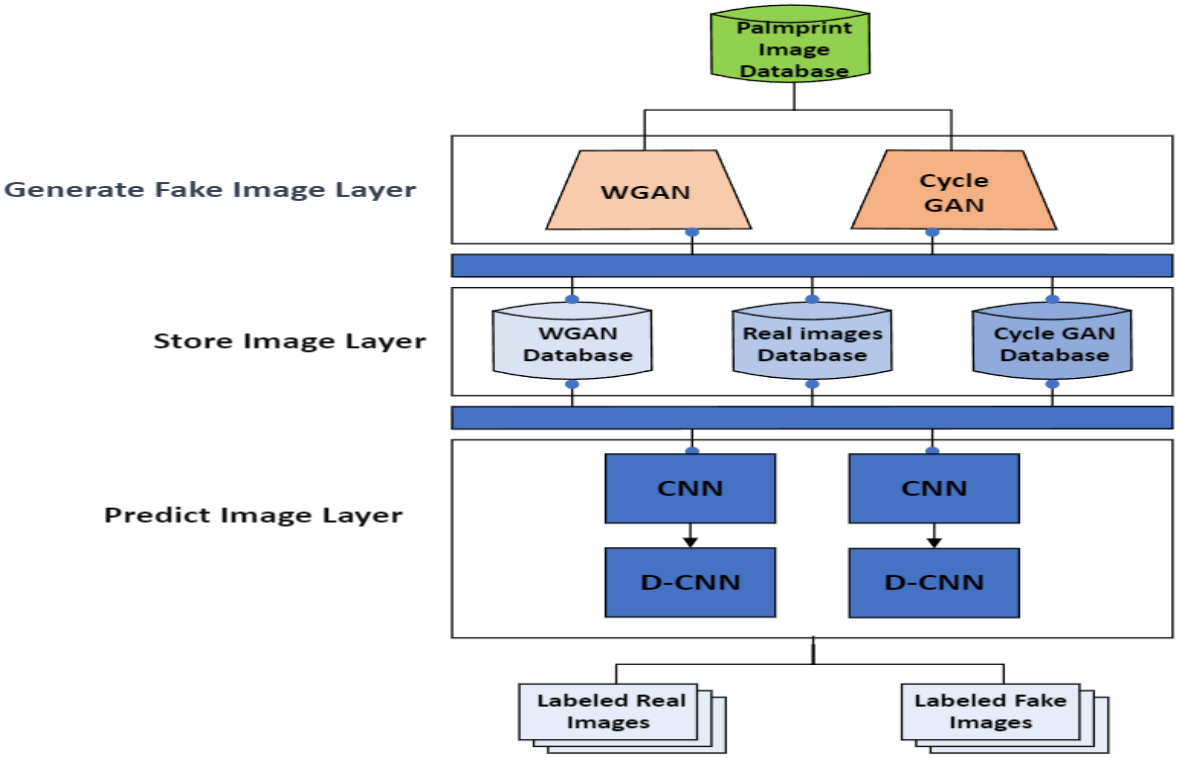

FIGURE 5. The architecture of the proposed method.

TABLE 1. Image sources.

Sources	Total
WGAN	2210
Cycle GAN	2053
POLY-U Data Set Left	1301

TABLE 2. Training setups.

Parameters	Values
Image size	256×256
Adam optimizer learning rate	0.001
Epochs	50
Batch size	64
Beta	(0.9, 0.999)

Similarly, all 2053 images in the Cycle GAN test dataset are fake. We used approximately 60% of the images from the fake dataset (approximately 1232 images) for training. From this training set, we selected around 20% of the images for validation (approximately 411 images) and assigned the remaining 20% (approximately 410 images) for the test set,

The selected image size is 150×150. We scaled the image width and weight to 256×256. Throughout the training process, we employed the Adam optimizer with a learning rate of 0.001 and set beta to (0.9, 0.999). The training spanned a total of 50 epochs. The batch size is configured to 64, as shown in TABLE 2.

We adopt three evaluation metrics for our experiment: precision, recall, and F1 score. Precision evaluates the model's accuracy in identifying positively predicted instances. It indicates the proportion of images classified as deepfakes by the model that are genuinely authentic. True Positives (TP) are correctly classified deepfakes, while False Positives (FP) are instances wrongly identified as deepfakes. The formula for precision is as follows: Precision = TP / (TP + FP).

Recall, also called sensitivity or true positive rate, assesses the model's capability to identify positive instances accurately. It measures the proportion of genuine deepfake images correctly classified by the model out of all the actual deepfake images provided. True Positives (TP) are instances correctly identified as deepfakes, while False Negatives (FN) are actual deepfake images wrongly classified as real. The formula for the recall is Recall = TP / (TP + FN).

The F1 Score represents a balance between precision and recall, calculated as the harmonic mean of both values. Its formula is: F1 = 2 × (precision × recall) / (precision + recall).

C. FAKE PALMPRINT SAMPLES

In WGAN, we implemented a technique known as weight clipping using a weight clipping limit of 0.01. This parameter setting is crucial to maintain the Lipschitz constraint on the discriminator during training. As an essential

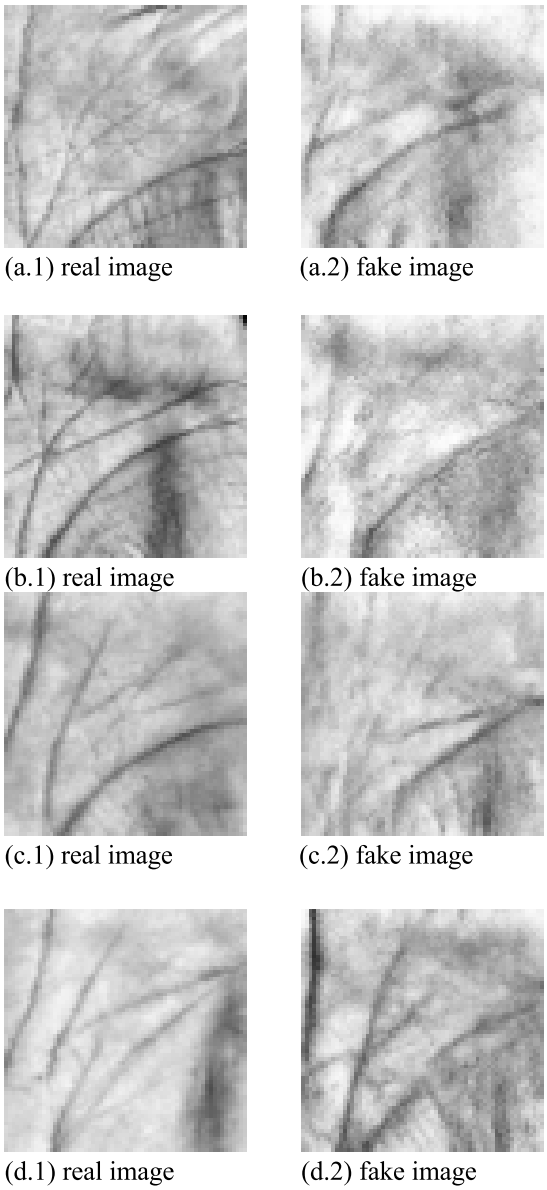

FIGURE 6. Generated images produced by WGAN. Pictures (a) to (d) showcase the real and generated images in pairs.

characteristic of WGANs, and compared to traditional GAN models, this helps to improve the model's training stability and convergence. Bias is not set because, in particular network architectures, it may be desirable to reduce the number of parameters to simplify the model, especially in large-scale networks, to minimize the risk of overfitting. The generators and discriminators are configured with Adam optimizers, with a learning rate of 0.03 and beta parameters of (0.9 and 0.999).

In our comprehensive experiments, the WGAN has demonstrated remarkable capabilities in generating high-quality images, showcasing a notable improvement in stability and a significantly reduced risk of mode collapse when juxtaposed with conventional GANs. The outcomes derived from the implementation of WGAN, as gleaned from our extensive experiments, are visually depicted in **Figure 6.** (a) to (d).

The Cycle GAN architecture consists of two generators and two discriminators, all of which are configured with Adam optimizers, with a learning rate of 0.0002 and a beta parameter of 0.5. During the training process, the bias and weights of the generator are adjusted based on the features of the training data and feedback from the loss function, enabling the generator to produce realistic images. The discriminator's objective is to distinguish the images generated by the generator from the real images. Like the generator, the bias and weights of the discriminator are adjusted during the training process based on the training data and the loss function. Experimental findings, illustrated in **Figure 7**(a) to (d), reveal the left side as real images and the right side displaying fake images. Manipulated images exhibit intensified patterns and lines, which are not characteristic of authentic human palmprints. Thus, it can be inferred that these images are fabricated.

D. DC-CNN PREDICTION RESULTS

The DC-CNN, constructed upon the CNN architecture, upholds the interpretive capabilities of D-CNN, ensuring robustness and adaptability in addressing cross-domain challenges in deepfake detection. It leverages the strengths of D-CNN to enhance MesoNet and MesoinceptionNet, resulting in significant improvements in deepfake content detection. Notably, it effectively mitigates the issue of false negatives being predicted as positives compared to WGAN. Furthermore, its performance in identifying deepfake content in Cycle GAN surpasses that of the other three methods. Detailed confusion matrices can be found in **Figure 8.** and **Figure 9.**

E. TRAINING

We used the Adam optimizer with a learning rate of 0.001 during the training process. The total number of iterations for training was 50. The batch size was set to 64, mainly stabilizing the training process. We recorded that, during the training process, the validation accuracy convergently approached the training accuracy from 10 to 20 epochs, as shown in **Figure 10**. Overall, the iterations showed a similar rising trend, which can also be observed in the loss, as shown in **Figure 11**. Fluctuations in validation accuracy and loss values are most likely due to batch normalization and dropout layers, which intensified this situation. However, aside from these fluctuations, it can be seen that the trends in validation accuracy and loss values are similar to those of the training accuracy and loss values. Although the validation accuracy and loss values are slightly fluctuating, they closely follow the training loss values, indicating that the model is not overfitting.

we calculated the time spent on each epoch during the training and validation phases to measure the time consumed. The experimental process shows that the training time for DC-CNN was 8688.54 seconds, almost twice as long as the training time for D-CNN, which was 4012.0400 seconds.

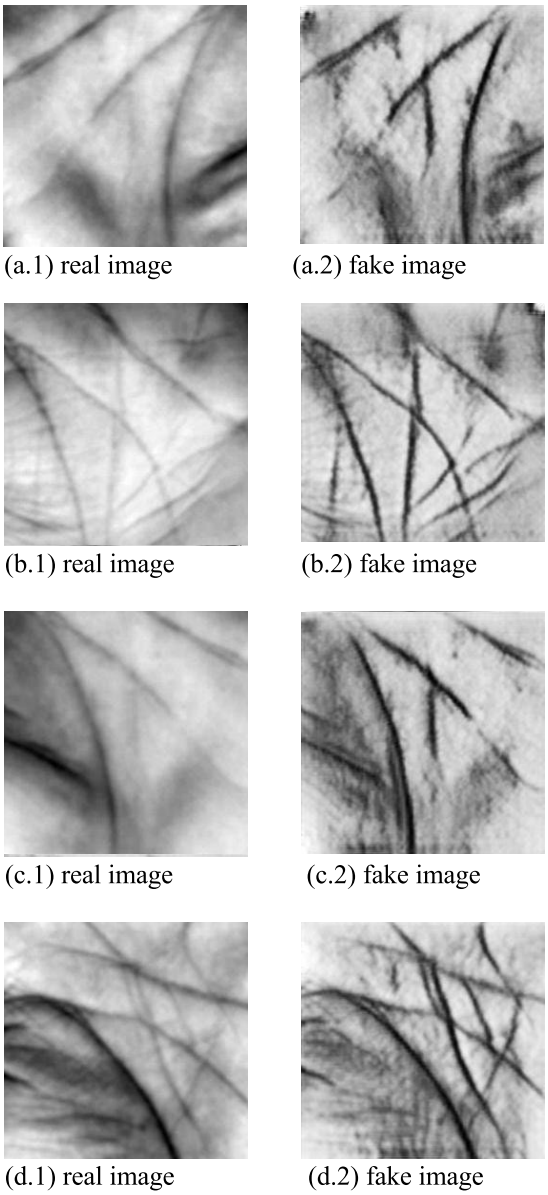

FIGURE 7. Generated images produced by Cycle GAN. Pictures (a) to (d) show the real and generated images in pairs.

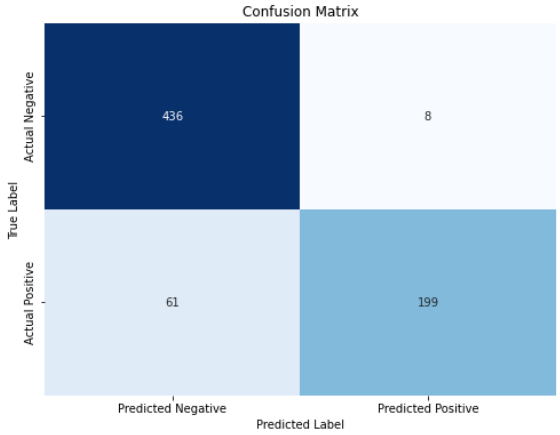

FIGURE 8. The performance of DC-CNN with confusion matrix in WGAN.

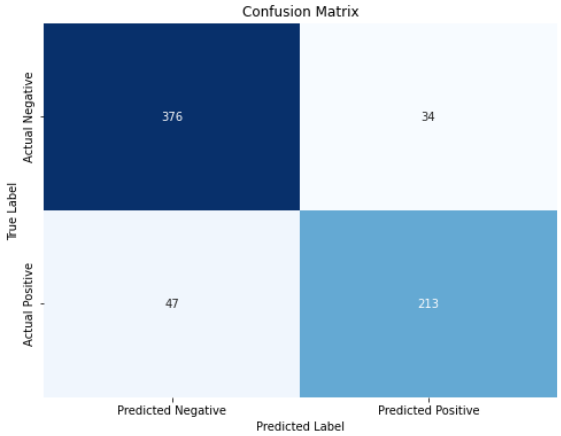

FIGURE 9. The performance of DC-CNN with confusion matrix in Cycle GAN.

This was mainly due to our adoption of dual-channel Siamese generation techniques.

The training time for D-CNN was longer than for MesonInceptionNet, which took 3820.1700 seconds, and MesoNet, which took 3775.2500 seconds. This is because D-CNN uses more parameters, which means it requires more computational time. Finally, the training times for MesonInceptionNet and MesoNet were very close, as they belong to the same type of model, with MesonInceptionNet introducing computational efficiency improvements based on MesoNet.

During the validation process, DC-CNN took 3997.5500 seconds, D-CNN 3504.2000 seconds, MesonInceptionNet 3545.9400 seconds, and MesoNet 3547.5200 seconds. DC-CNN still required more computational time for validation, as shown in **Figure 12.** Hence, it can be seen that, overall, DC-CNN incurs a higher computational cost, but in return, it provides superior model prediction accuracy.

F. DISCUSSION

To understand the model's performance and cross-domain execution, it is necessary to apply the models of deepfake detection to these images for a comparison to gain insight into the generalization capabilities of the proposed model. We expanded our analysis by evaluating the images from each data source separately. This allowed us to understand the model's generalization abilities better. Therefore, we augmented more synthetic images from the original data, including 2210 deepfake images from WGAN and 2053 deepfake images from Cycle GAN.

We then combined these images from different data sources with real images separately. We began by comparing the model evaluations, specifically for WGAN, with the test set of 704 images and Cycle GAN with the test set of 670 images. In TABLE 3, we utilized MesoNet + WGAN with an accuracy of 36.93%. In MesoNet, there are

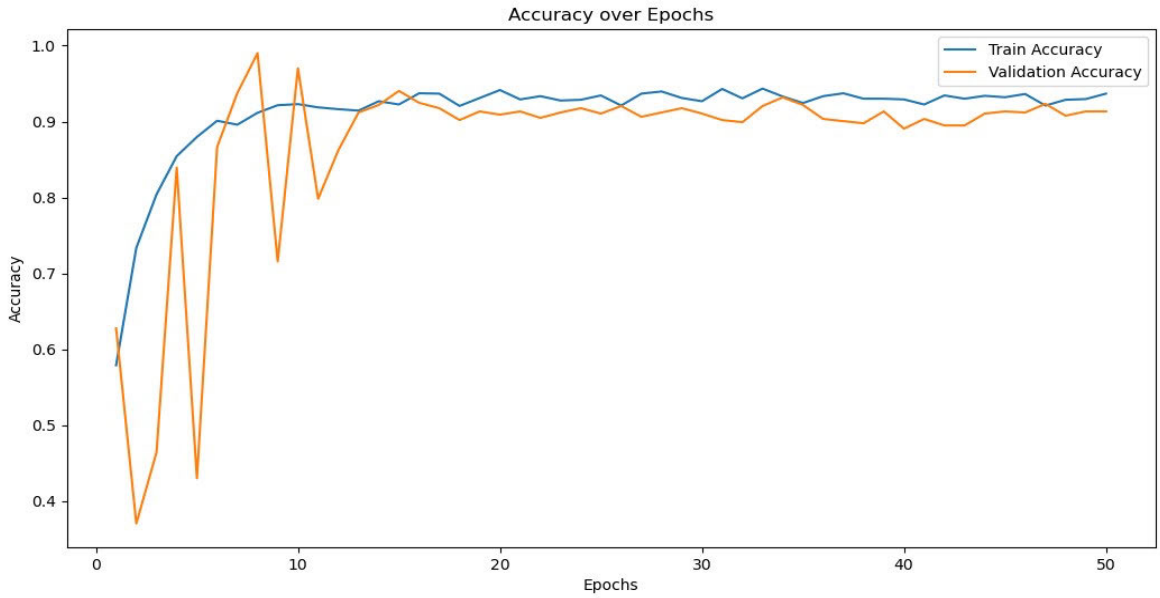

FIGURE 10. Training accuracy over epochs.

TABLE 3. WGAN accuracy reports.

Model	Accuracy
MesoNet	36.93%
MesoInceptionNet	37.78%
D-CNN	64.49%
DC-CNN	90.20%

TABLE 4. Cycle GAN accuracy reports.

Model	Accuracy
MesoNet	38.81%
MesoInceptionNet	65.67%
D-CNN	63.88%
DC-CNN	87.91%

260 in TP, indicating a relatively good performance, while FP is 444, which is very high, resulting in a low overall accuracy of approximately 36.93%. This suggests that the model's ability to identify the negative class correctly could be better, adversely affecting its overall performance mesoInceptionNet + WGAN with an accuracy of 37.78%. MesoInceptionNet with TN at 7, FP at 437, FN at 1, and TP at 259, the precision, calculated as the ratio of TP + FP, results in a precision of approximately 0.3778.

This indicates that the model has limited ability to identify true positive cases relative to all accurately predicted positive cases, which is potentially due to the high rate of false positives. D-CNN + WGAN with an accuracy of 64.49%; and + DC-CNN with an accuracy of 90.20%. We observed that MesoNet performed poorly with WGAN, possibly due to the fewer parameters in the MesoNet architecture and the lower resolution of the generated images, leading to misjudgments in MesoNet's recognition. With the improvement of the architecture, MesoInceptionNet and D-CNN also improved.

This is because these models' predictions of FP and FN were not ideal. Particularly MesoNet's predictions leaned heavily towards FP, resulting in notably low accuracy. Since D-CNN is an optimization of CNN-based, it showed a better performance. As for DC-CNN, it integrates an initial CNN equivalent layer with a single-channel filter, followed by a dual-channel pathway and the subsequent merging of feature vectors, effectively enhancing its accuracy, especially in reducing FP and FN.

In our evaluation using Cycle GAN, as shown in TABLE 4, we observed the performance of MesoNet + Cycle GAN with an accuracy of 38.81%. In MesoNet evaluation, the TP at 260 performs relatively well, while the FP at 410 leads to lower overall accuracy. MesoInceptionNet + Cycle GAN with an accuracy of 65.67%, D-CNN + Cycle GAN with an accuracy of 63.88%, and DC-CNN + Cycle GAN with an accuracy of 87.91%. Significantly, MesoInceptionNet and D-CNN outperformed MesoNet in the prediction report. Furthermore, the accuracy of MesoInceptionNet is like that of DC-CNN, with DC-CNN being the best among these models. This situation is similar to that of WGAN and Cycle GAN, where the performance of Cycle GAN is abysmal in FP.

Based on the comprehensive experimental results of comparing MesoNet, MesoInceptionNet, D-CNN, and DC-CNN under both WGAN and Cycle GAN frameworks, DC-CNN exhibits greater versatility and higher accuracy in predictions. These four models are all based on CNN networks. The

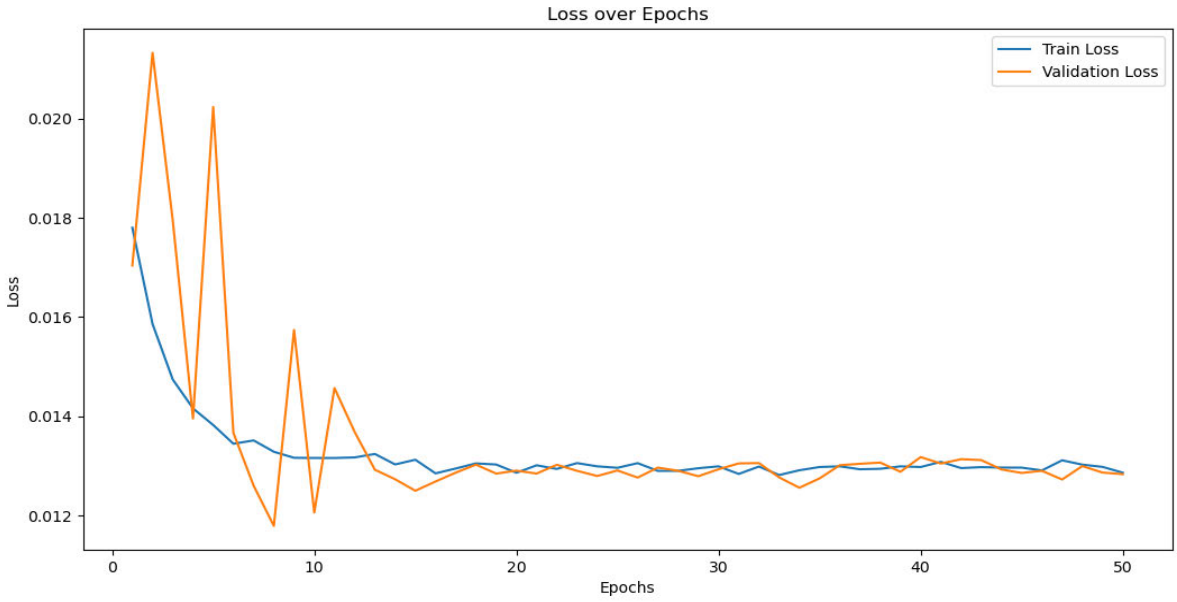

FIGURE 11. Training loss over epochs.

TABLE 5. WGAN weighted average of prediction report.

Model	Precision	Recall	F1 support
MesoNet	0.1364	0.3693	0.1992
MesoInceptionNet	0.6893	0.3778	0.2196
D-CNN	0.7728	0.6449	0.5195
DC-CNN	0.9083	0.9020	0.8992

TABLE 6. Cycle GAN weighted average of prediction report.

Model	Precision	Recall	F1 support
MesoNet	0.1506	0.3881	0.2170
MesoInceptionNet	0.7487	0.6567	0.6532
D-CNN	0.6257	0.6388	0.6255
DC-CNN	0.8786	0.8791	0.8785

experimental results regarding image of prediction report are shown in TABLE 5 and TABLE 6 compared to existing models. A decrease in accuracy can be observed in Mesonet, with WGAN showing precision = 0.1364, recall = 0.3693, F1 support = 0.1992, and Cycle GAN showing precision = 0.1506, Recall = 0.3881, F1 support = 0.2170. The accuracy of MesoInceptionNet is also observed in WGAN, showing precision = 0.6893, recall = 0.3778, F1 support = 0.2196, and in Cycle GAN, showing precision = 0.7487, recall = 0.6567, F1 support = 0.6532. It is found from the statistics that D-CNN demonstrates better generalization in WGAN, showing precision = 0.7728, recall = 0.6449, F1 support = 0.5195, and in Cycle GAN, precision = 0.6257, recall = 0.6388, F1 support = 0.6255. DC-CNN inherits this characteristic and incorporates a dual-channel feature in the fully connected part to enhance the feature vectors, thereby achieving better results with precision = 0.9083, recall = 0.9020, F1 support = 0.8992 in WGAN, and precision = 0.8786, recall = 0.8791, F1 support = 0.8785 in Cycle GAN. This reflects the challenging task of achieving generalization and underscores the importance of real-world applications. As shown in **Figure 13**, a further investigating error analysis of the WGAN results indicates the results in terms of confidence scores, which essentially reflect the probability that the image is a deepfake or not.

1. Top left (a) as ‘correct real‘ model confidence is 0.9266: The model is highly confident that this image is real and correctly classifies it.
2. Top right (b) as ‘correct deepfake‘ is 0.1587: Despite the correct classification as a deepfake, the model shows shallow confidence, indicating uncertainty.
3. Bottom left (c) as ‘misclassified real‘ is 0.2340: This image is real, but was misclassified. The low confidence score here also indicates that the model was unsure of its decision.
4. Bottom right (d) as ‘misclassified deepfake‘ is 0.9812: The model incorrectly classified a deepfake as real, but was highly confident, suggesting that the model was quite sure of its incorrect decision.

The provided data could be analyzed using the confidence scores from the Cycle GAN results shown in **Figure 14**:

1. Top left (a) as ‘correct real‘ is 0.9998: The model is almost entirely certain that this image is real, showing a very high confidence score of 0.9998. It accurately identifies the image as real, reflecting a strong alignment with features typical of genuine images.
2. Top Right (b) as ‘correct deepfake‘ is 0.0395:

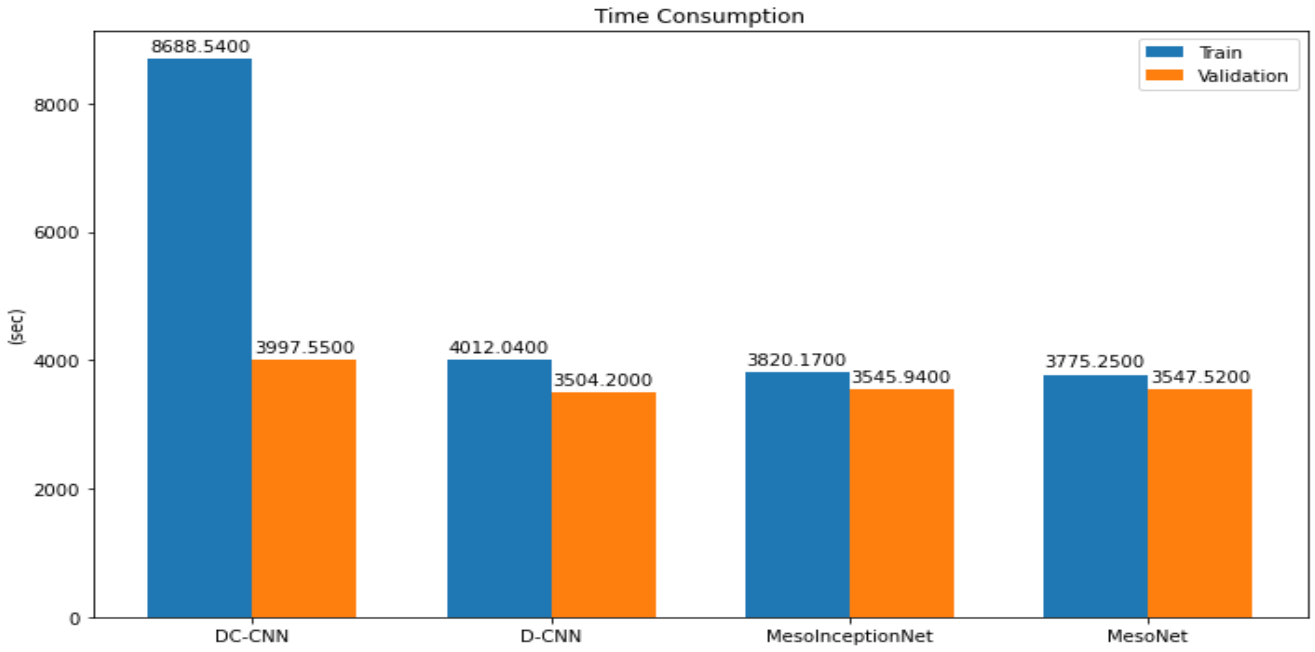

FIGURE 12. Time consumption in different models.

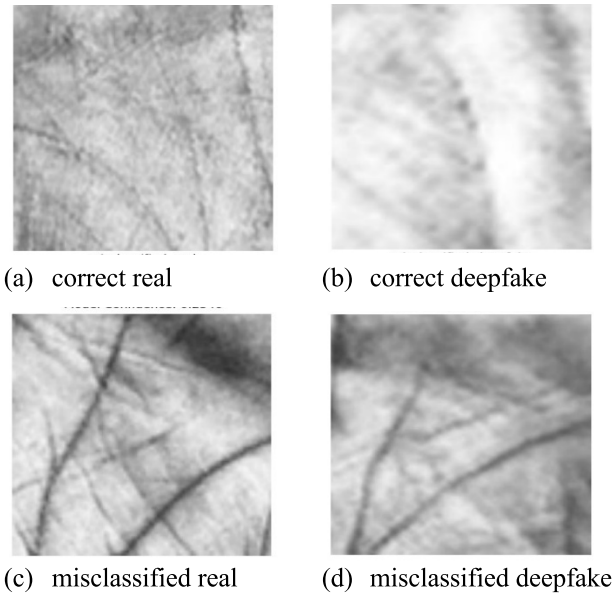

FIGURE 13. From top left (a) to bottom right (d), displaying the model confidence performance in WGAN.

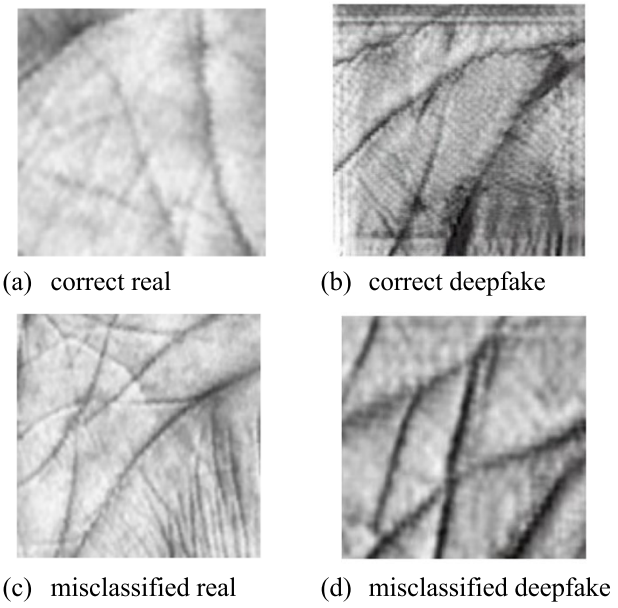

FIGURE 14. From top left (a) to bottom right (d), displaying the model confidence performance in Cycle GAN.

Although the model correctly identifies the image as a deepfake, its confidence is shallow at 0.0395. This low confidence score indicates a high level of uncertainty, suggesting that the image closely mimics the characteristics of a real one, making it hard for the model to confidently classify it as a deepfake.

3. Bottom Left (c) as 'misclassified real' is 0.3670:
   The image is real, yet the model misclassified it with a moderate confidence of 0.3670. This indicates some ambiguity in the image's features, making the model unsure; hence, the misclassification as not real.
4. Bottom Right (d) as 'misclassified deepfake' is 0.7081:
   The model incorrectly identifies this deepfake as real with a relatively high confidence of 0.7081. This suggests that the deepfake image has been well-crafted to closely resemble a real image, misleading the model and making it confident in its incorrect decision.

VI. FUTURE SCOPE

It's essential to adjust specific parameters and hyperparameters of the deepfake detection method to optimize the model's performance. This helps find a better balance, improve accuracy, and reduce latency. Exploring various optimization techniques and configurations is crucial for further enhancement. Staying updated with the latest deepfake advancements and continuously learning new features are vital for long-term effectiveness. Additionally, we will continue to focus on detecting deepfakes and adopting novel models to continuously optimize the model architecture.

VII. CONCLUSION

Detecting deepfake content is challenging, as the methods for generating these forged images or videos involve various techniques and steps, each of which may simulate or modify the real image data differently. This complexity makes detecting these forgeries a challenging problem, as each generation's technique may require different detection strategies or tools. The task of detecting deepfake content is considered to be a binary classification problem. Inspired by the firm foundation of CNN in detecting fake images, we proposed a dual-channel technique in this paper based on the use of CNN architecture to enhance the accuracy and generalizability of detecting forged images, compared to various network architectures.

DC-CNN features a deeper CNN network structure and an increased number of parameters. Compared to MesoNet and MesoInceptionNet that have shallower network architecture, DC-CNN's deeper structure enhances recognition accuracy. DC-CNN maintains the original D-CNN's generalization advantage, as well as further strengthening its extensibility.

Despite significant variations in the recognizability of these images across these architectures, the DC-CNN provides a good, balanced performance across all data sources. Analyzing WGAN-generated fake palmprints, the model demonstrated impressive results with a weighted precision of 90.83%, a weighted recall rate of 90.20%, a weighted F1 score of 89.92%, and an overall accuracy of 90.20%. In contrast, for fake palmprints using Cycle GAN, the model showed a weighted precision of 87.86%, a weighted recall of 87.91%, a weighted F1 score of 87.85%, and an accuracy of 87.91%.

Most research has been focused on facial detection, with little emphasis on detecting different biometric features, such as palmprints. Therefore, we extended the architecture proposed by D-CNN to detect palmprint textures and emphasized the generality of this approach. The advantage of doing so is that there are two networks with identical configurations, parameters, and weights. Typically, we train only one of the subnetworks and use the same configuration for the others. This is used to find the differences between inputs by comparing their feature vectors. A dual-channel technique can be used to easily handle these forged images. Therefore, the proposed model performs well on the given dataset, demonstrating a good generalization performance.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable comments to improve the quality of this manuscript. They also thank the National Centre for High-Performance Computing (NCHC), National Applied Research Laboratories (NARLabs), Taiwan, for providing computational and storage resources. They also express their appreciation to the Broadband Networks Laboratory, Chunghwa Telecom Laboratories, for their support.

REFERENCES

[1] D. Zhang, W.-K. Kong, J. You, and M. Wong, "Online palmprint identification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 9, pp. 1041–1050, Sep. 2003.

[2] T. Connie, A. T. B. Jin, M. G. K. Ong, and D. N. C. Ling, "An automated palmprint recognition system," *Image Vis. Comput.*, vol. 23, no. 5, pp. 501–515, May 2005.

[3] B. S. Kadam and P. Deshmane, "Palm print recognition based on local binary pattern," *Int. J. Innov. Technol. Adapt. Manag.*, vol. 1, no. 6, Mar. 2014.

[4] I. Perov, D. Gao, N. Chervoniy, K. Liu, S. Marangonda, C. Umé, M. Dpfks, C. S. Facenheim, R. P. Luis, J. Jiang, S. Zhang, P. Wu, B. Zhou, and W. Zhang, "DeepFaceLab: Integrated, flexible and extensible face-swapping framework," 2020, *arXiv:2005.05535*.

[5] S. Minaee, M. Minaei, and A. Abdolrashidi, "Palm-GAN: Generating realistic palmprint images using total-variation regularized GAN," 2020, *arXiv:2003.10834*.

[6] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of Wasserstein GANs," 2017, *arXiv:1704.00028*.

[7] Y. Lu, Y.-W. Tai, and C.-K. Tang, "Attribute-guided face generation using conditional CycleGAN," 2017, *arXiv:1705.09966*.

[8] Y. Zhang, L. Zheng, and V. L. L. Thing, "Automated face swapping and its detection," in *Proc. IEEE 2nd Int. Conf. Signal Image Process. (ICSIP)*, Aug. 2017, pp. 15–19.

[9] P. Yu, Z. Xia, J. Fei, and Y. Lu, "A survey on deepfake video detection," *IET Biometrics*, vol. 10, no. 6, pp. 607–624, Nov. 2021.

[10] Y.-J. Heo, Y.-J. Choi, Y.-W. Lee, and B.-G. Kim, "Deepfake detection scheme based on vision transformer and distillation," 2021, *arXiv:2104.01353*.

[11] 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.

[12] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "MesoNet: A compact facial video forgery detection network," in *Proc. IEEE Int. Workshop Inf. Forensics Secur. (WIFS)*, Dec. 2018, pp. 1–7.

[13] M. Aykut and M. Ekinci, "Developing a contactless palmprint authentication system by introducing a novel ROI extraction method," *Image Vis. Comput.*, vol. 40, pp. 65–74, Aug. 2015.

[14] D. G. Lowe, "Distinctive image features from scale-invariant key points," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[15] X. Wu, Q. Zhao, and W. Bu, "A SIFT-based contactless palmprint verification approach using iterative RANSAC and local palmprint descriptors," *Pattern Recognit.*, vol. 47, no. 10, pp. 3314–3326, Oct. 2014.

[16] Q. Zhao, W. Bu, and X. Wu, "Sift-based image alignment for contactless palmprint verification," in *Proc. Int. Conf. Biometrics (ICB)*, Jun. 2013, pp. 1–6.

[17] Y.-T. Luo, L.-Y. Zhao, B. Zhang, W. Jia, F. Xue, J.-T. Lu, Y.-H. Zhu, and B.-Q. Xu, "Local line directional pattern for palmprint recognition," *Pattern Recognit.*, vol. 50, pp. 26–44, Feb. 2016.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Neural Inf. Process. Syst. Conf. (NIPS)*, 2012, p. 1097.

[19] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[21] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," 2015, *arXiv:1512.00567*.
[22] L. Dian and S. Dongmei, "Contactless palmprint recognition based on convolutional neural network," in *Proc. IEEE 13th Int. Conf. Signal Process. (ICSP)*, Nov. 2016, pp. 1363–1367.
[23] Y. Wang, L. Fei, S. Zhao, Q. Zhu, J. Wen, W. Jia, and I. Rida, "Dense hybrid attention network for palmprint image super-resolution," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 54, no. 4, pp. 2590–2602, Apr. 2024.
[24] T. Li, Z. Zhao, C. Sun, L. Cheng, X. Chen, R. Yan, and R. X. Gao, "WaveletKernelNet: An interpretable deep neural network for industrial intelligent diagnosis," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 52, no. 4, pp. 2302–2312, Apr. 2022.
[25] F. Santoso and A. Finn, "A data-driven cyber–physical system using deep-learning convolutional neural networks: Study on false-data injection attacks in an unmanned ground vehicle under fault-tolerant conditions," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 53, no. 1, pp. 346–356, Jan. 2023.
[26] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a 'Siamese' time delay neural network," in *Proc. 6th Int. Conf. Neural Inf. Process. Syst. (NIPS)*. San Francisco, CA, USA: Morgan Kaufmann, 1993, pp. 737–744.
[27] S. Dey, A. Dutta, J. I. Toledo, S. K. Ghosh, J. Llados, and U. Pal, "SigNet: Convolutional Siamese network for writer independent offline signature verification," 2017, *arXiv:1707.02131*.
[28] D. Chicco, "Siamese neural networks: An overview," in *Artificial Neural Networks*. New York, NY, USA: Humana, 2021, pp. 73–94, doi: 10.1007/978-1-0716-0826-5.

TSAI MIN-JEN (Member, IEEE) received the B.S. degree in electrical engineering from National Taiwan University, in 1987, the M.S. degree in industrial engineering and operations research from the University of California at Berkeley, in 1991, and the engineering and Ph.D. degrees in electrical engineering from the University of California at Los Angeles, in 1993 and 1996, respectively. From 1987 to 1989, he was a second Lieutenant in Taiwan Army. From 1996 to 1997, he was a Senior Researcher with America Online Inc. In 1997, he joined the Institute of Information Management, National Chiao Tung University, Taiwan. He is currently a Full Professor with the Institute of Information Management, National Yang Ming Chiao Tung University. His research interests include multimedia systems and applications, digital rights management, digital watermarking and authentication, digital forensics, and enterprise computing for electronic commerce applications. He is a member of ACM and Eta Kappa Nu.

CHANG CHENG-TAO received the M.S. degree from the Institute of Information Management, National Yang Ming Chiao Tung University, in 2023, where he is currently pursuing the Ph.D. degree. He is a Researcher with the Institute of Broadband Networks Laboratory, Chunghwa Telecom Laboratories, Yangmei District, Taoyuan City. His research interests include digital image process, generative adversarial networks, and deepfake detection.