INTERNATIONAL JOURNAL ON INFORMATICS VISUALIZATION

journal homepage : www.joiv.org/index.php/joiv

In-Air Hand Gesture Signature Recognition Using Multi-Scale Convolutional Neural Networks

Alvin Lim Fang Chuen [a], Khoh Wee How [a,*], Pang Ying Han [a], Yap Hui Yen [a]

[a] *Faculty of Information Science & Technology (FIST), Multimedia University, Bukit Beruang, 75450, Melaka, Malaysia*

Corresponding author: [*]whkhoh@mmu.edu.my

***Abstract*—The hand signature is a unique handwritten name or symbol that serves as a proof of identity. Due to its practicality and widespread use, hand signature is still used by financial institutions as a means of verifying and validating the identity of their customers. The emergence of the COVID-19 global pandemic has raised hygiene concerns regarding the conventional touch-based hand signature recognition system, which often requires sharing the acquisition devices among the public. This paper presents in-air hand gesture signature recognition using convolutional neural networks to address this concern. We designed a shallow multi-scale convolutional neural network using 3x3 and 5x5 kernel filter sizes to extract features on different scales. The feature maps from these two filters are then concatenated to provide more robust features, which improve the model's performance. The proposed architecture was evaluated on the In-Air Hand Gesture Database (iHGS) and compared its performance with other existing architectures, including GoogleNet, AlexNet, VGG-16, and ResNet-50, under the same experimental setting. The experiment results show that the proposed architecture outperforms other architectures, which obtained the highest accuracy of 93.00%. On the other hand, our architecture consumed significantly fewer computational resources, requiring only an average of 3 minutes and 33 seconds to train. Additionally, the performance of the proposed architecture could be further enhanced by integrating it with recurrent neural networks (RNN). This integrated architecture of convolutional recurrent neural networks (C-RNN) can capture spatio-temporal features simultaneously.**

***Keywords*— Hand gesture signature; gesture recognition; in-air signatures; convolutional neural networks.**

Manuscript received 7 Dec. 2022; revised 11 Jul. 2023; accepted 24 Aug. 2023. Date of publication 30 Nov. 2023.

I. INTRODUCTION

Hand signature recognition is one of the most researched topics in the field of biometrics due to its immense potential for security applications. A signature is a person's unique handwritten name or symbol that proves their identity. Due to its widespread use and practicality, hand signatures have become one of the most widely used forms of biometrics that have been established for centuries as a means to validate a person's identity for both authentication and verification purposes.

The traditional way of acquiring and verifying a person's signature is prone to forgery. Before digital signature acquisition systems were introduced, the verification method still relied on a handwritten approach, such as verifying bank cheques, which is vulnerable to forgery. This allows anyone to learn and mimic a person's signature for unauthorized uses.

Several attempts have been made to address the forgery issues by using digital acquisition devices, such as tablets and styluses [1], [2], [3]. Unlike the traditional approach, which only captures the end result of a signature (the appearance), it does not capture other behavioural traits, such as the time it takes to complete the signature and the pressure applied on the writing surface. The tablet and stylus approach are able to capture and record its dynamic properties, such as the time it takes to complete a signature as well as the pressure applied on the writing surface. Capturing these additional traits is beneficial for both authentication and verification purposes because they are more difficult to imitate, making forgery attempts less likely to succeed. This can further improve the accuracy of identifying a person's identity.

Recently, the demand for the contactless biometric system has drastically increased due to the emergence of the COVID-19 global pandemic [4], which has raised hygienic concerns about the touch-based system. While the tablet and stylus-based hand signature recognition system is reliable and still widely used in institutions such as banks. The touch-based system is prone to germ and virus contamination as it often requires sharing among the public. This has increased

research and development for contactless hand signature recognition systems.

Two main approaches in contactless hand signature recognition are vision-based and signal-based. The vision-based approach uses a camera sensor to capture a static image or video of the signatures, whereas the signal-based approach utilizes the built-in sensors of mobile devices such as the accelerometer and gyroscope to capture the time-series data while signing [5], [6], [7], [8]. On the other hand, various classification methods have been explored for contactless hand signature recognition, such as the handcrafted method as well as the deep learning methods. One of the most widely used deep learning methods is convolutional neural network which is more robust than handcrafted methods as it can automatically extract and learn the features of hand signatures.

Li et al. [9] proposed signal-based hand signature recognition using recurrent neural networks with smartwatch sensors data. The hand signatures were captured in the form of a 9-dimensional accelerometer, gyroscope, and attitude from the Apple Watch Series 6. The authors conducted the experiments on a self-collection of 22 subjects. Each person was given a smartwatch to wear and provided ten genuine and 10 forged signatures by signing in the air while wearing the smartwatch. The proposed method reported an Equal Error Rate (EER) of 0.83%.

Zhao et al. [10] proposed a multi-modal Siamese neural network for hand signature recognition, utilizing a unique way to capture the signal data. A total of 8 participants were asked to write their signatures on a piece of paper placed 5 centimeters away from a smartphone. The phone microphone captured the sound produced during the writing process. This method obtained an EER of 5.79%. Although the proposed method is novel, its practicality for real-world applications is limited as it requires both a pen and paper and a smartphone's microphone.

Guerra et al. [11] investigated using Leap motion sensors. They present an alternative approach to capturing hand signatures that eliminates the need to wear or hold a device by utilizing the motion tracking capability of the motion sensor. The experiments were conducted using the Least Squares Support Vector Machine (LV-SVM) on their dataset, consisting of 10 genuine and ten forged signatures provided by 100 individuals, and obtained an EER of 0.25%.

De Luisa et al. [12] utilized a haptic device to effectively capture the in-air signatures for identity verification using dynamic time warping and hidden Markov models. The proposed methods were evaluated on a self-collected dataset from 52 different individuals. The experiment results showed an EER of 1.9% for random and 2.7% for skilled forgeries.

Kancharla et al. [13] designed a lightweight convolutional neural network for handwritten signature recognition. It consists of two convolution layers and three fully connected layers to learn and classify the features of hand signatures. The proposed network was trained and tested on three different nationality signatures (Chinese, Dutch, and Persian) from the SigComp 2011 and UTsig Persian datasets. The authors conducted the experiments using two different adaptive learning optimizers, Adam and RMSprop. The highest accuracy obtained was 100% using the Adam optimizer on Dutch signatures, while the lowest accuracy was 82.98 using RMSprop on Chinese signatures.

Xiao and Ding [14] proposed a two-stage Siamese network model for offline handwritten signature recognition. The traditional Siamese network was found to be insufficient in representing the writers' style features and struggled with imbalanced positive and negative signature samples. To address these issues, their model utilized a two-stage approach to verify original and enhanced signatures simultaneously. Additionally, they employed Focal Loss to handle the extreme sample imbalance. The proposed model achieved the highest accuracy of 95.66% on the Cedar dataset. Several other studies have also focused on handwritten signature recognition and obtained promising results [15], [16], [17], [18], [19]. However, despite its widespread usage, the handwritten signature is prone to forgery. Researchers have also explored other alternative approaches to address this issue, such as in-air signed hand signature recognition [20], [21], [22], which aims to mitigate the issues of forgery by leveraging the unique biometric traits with signing gestures performed in mid-air.

Even though deep learning methods can outperform handcrafted methods in terms of accuracy, one of the main challenges of deep learning is that it requires a large dataset for training the model [23], [24], [25]. Training a model with a small dataset can result in overfitting, which leads to high accuracy on training samples and low accuracy on test samples. There is still a lack of an in-air signed hand signature database with large samples due to cost and time constraints on collecting the signature samples. The In-Air Hand Gesture database (iHGS) was chosen for our work as it is currently the only publicly available dataset for hand gesture signatures [26].

This work proposes a multi-scale convolutional neural network for in-air hand gesture signature recognition. The remainder of this paper is structured as follows: In Section 2, the proposed architecture, pre-processing methods, experimental setup, and settings are explained in details. Experimental results and analysis are discussed in Section 3. Finally, Section 4 concludes our findings and provides suggestions for future work.

II. Materials and Method

This section proposes a multi-scale convolutional neural network (MS-CNN). The architecture of the model, the description of the database, pre-processing methods, and experimental setup and settings are explained in detail.

A. MS-CNN architecture

The architecture of the MS-CNN comprises two convolution layers, two parallel convolution layers, and two fully connected layers, as illustrated in Figure 1. The model's input is a single-channel image with a resolution of 224x224x1.

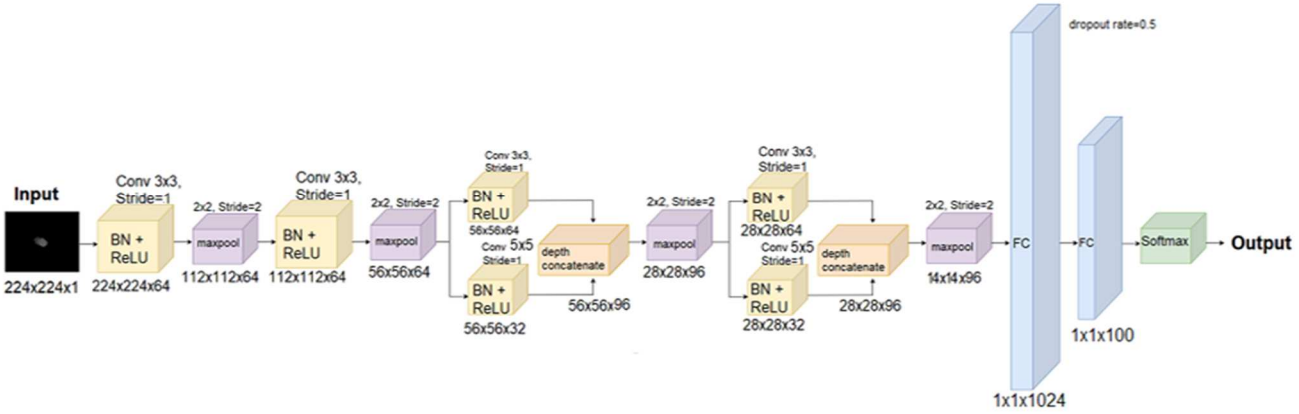

Fig. 1 MSCNN Architecture

The first two layers of the model are convolutional layers with a kernel size of 3x3 and 64 filters. These layers are designed to extract low-level features such as edges, textures, and small patterns from the initial input. The output of the second convolution layer is then passed to the subsequent layer, which is the parallel convolution layer. The parallel convolution layer consists of two separate convolution layers, one with a kernel size of 3x3 and 64 filters and another with a kernel size of 5x5 and 32 filters. The 5x5 filter is used to extract high-level features such as shape. Finally, a depth concatenation operation is performed by stacking the output from both parallel convoluted layers, resulting in a total of 96 filters. Figure 2 illustrates the parallel convolution layer architecture: two separate convolutional layers with different kernel sizes and filter configurations.

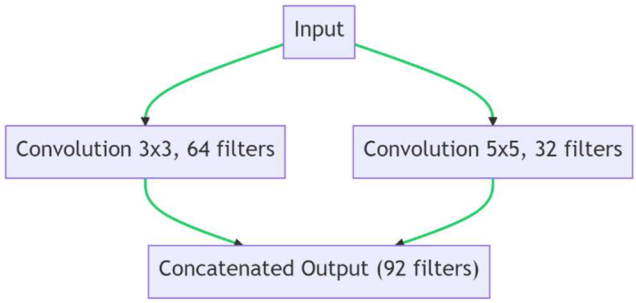

Fig. 2 Parallel Convolution Layer

The output from the first parallel convolution layer is fed as input to the second parallel convolution layer for additional feature extraction. The process in the second parallel layer is similar to the first one, where the 3x3 and 5x5 filters are used to extract more complex features from the first concatenated features. Doing this allows the model to learn more robust feature representations.

The concatenated features from the second parallel convolution layer are then fed to a fully connected layer with 1024 neurons. This layer is used to map the features from the previous layer. In addition, the dropout regularisation technique is applied at a rate of 50% to reduce the chance of overfitting by dropping a random selection of neurons. Another fully connected layer is added after the first one with 100 neurons, corresponding to the total number of classes. The output from this layer is then passed to the final layer, where the SoftMax function is applied for classification purposes. The SoftMax function is defined as:

$$f(x_i) = \frac{e^{x_i}}{\Sigma_{j=1}^{n} e^{x_j}} \quad (1)$$

The i and j denote the index of the input x, where x_i is the input for a given class i, and n is the total number of classes. The inputs are computed into a probability distribution with values ranging from 0 to 1, and the sum of all class probability equals 1. This allows the model to output a probability score for each class at the output layer.

To further improve the model's overall performance, the output of each convolution layer is normalized using the Batch Normalisation (BN) technique before the activation function. This normalization technique can accelerate the training time and improve model accuracy by reducing the changes in the input distribution to each layer,nown as the internal covariate shift during the model training process [27].

The Rectified Linear Unit (ReLU) is chosen as the activation function for our model. The ReLU is defined as:

$$f(x) = max(0, x) \quad (2)$$

where x is the input of the function, and 0 is the threshold value. If the input x is a negative value, the output of the function is 0. Otherwise, the output is equal to the input value. Furthermore, the max pooling technique is applied with a window size of 2x2 pixels and a stride value of 2 to reduce the dimension of the features output from the convolution layers. A max-pooling layer is added after the first and second convolution layers, and another max-pooling layer is added for both parallel convolution layers after the depth concatenation. This could help reduce the computation cost and make the model more efficient by reducing the dimension of the features, resulting in a faster model training time.

B. Dataset

The In-Air Hand Gesture Signature (iHGS) database was the sole dataset used in this work for experiments and performance evaluation of models [12]. It consists of both genuine and forged in-air signatures from 100 individuals. This database's total number of samples is 2980 (2000 genuine & 980 forged). The hand signatures were collected in a controlled environment using the Microsoft Kinect sensor, which captures colour and depth images. In addition, the sensor was pre-set at 640 x 480 resolution with 30 frames per

second. Each person contributed 20 samples of their own signature. As for the forged signatures, each person was instructed to learn and imitate the signatures of other participants. The acquisition of forged signatures only took place once they were ready.

C. Pre-processing

In this work, only the depth samples were utilized. The depth samples are single-channel image sequences that require fewer computational resources and are easier to process compared to the color samples, which are three-channel images. Although convolutional neural networks are able to extract and learn features without any handcrafted input, it is crucial to pre-process the training samples by removing the background noise. This is to prevent the model from incorrectly interpreting such noise as features and consequently learning from it. The entire flow of pre-processing the image sequences is illustrated in Figure 3.

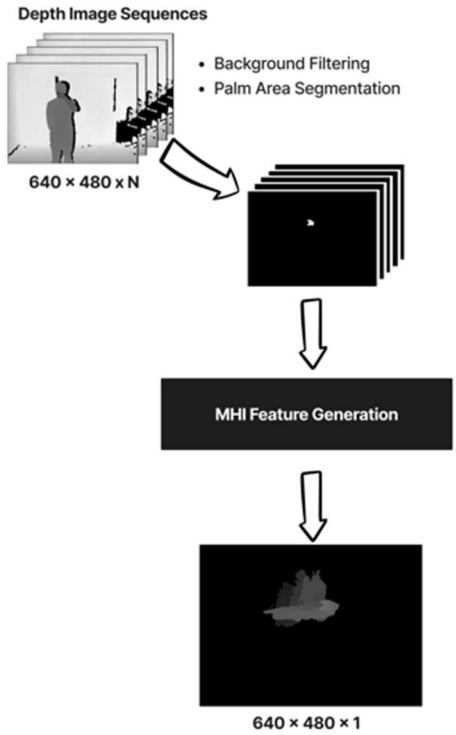

Fig. 3 Image Sequence Pre-processing

The region of interest (ROI) in the depth image sequences is the entire palm region, and it is the closest object to the sensor. By applying a threshold value of 180, any pixel values in the image less than this were set to 0, which is the pixel value for the black color, resulting in only the ROI remaining in the image. Besides, the palm may not be the only object closest to the sensor while signing in the air; the face region could also be the closest object. This could lead to the face region being incorrectly captured together with the palm region. To alleviate this issue, the predictive palm segmentation algorithm was applied to accurately segment the palm region, as proposed by Khoh et al. [28].

Next, the Motion History Image (MHI) [29], [30] were generated from the segmented image sequences. An in-air signed hand signature can consist of various frames, ranging from 50 to 150. The MHI algorithm is able to condense any number of frame sequences into a single static image while preserving the spatio-temporal information by capturing the silhouettes of hand-signing motion. Figure 4 illustrates the generated MHI, where the brighter region represents the most recent motion of the palm while the darker region is the motions from the earlier frames.

Fig. 4 Motion History Image

D. Experimental Setup

To evaluate the performance of our proposed model, we created three architectural variants and conducted an initial comparison with architectures A, B, and C. Based on this comparison, we selected the best architecture to make a performance analysis and comparison with several popular models that participated in the annual ImageNet Large Scale Visual Recognition Challenge (ILSVRC), including GoogleNet, AlexNet, VGG-16, and ResNet-50. The experiments were conducted using Matlab R2021a on a desktop computer running Windows 11 64-bit operating system equipped with an Intel i5-12400f CPU with a base clock speed of 2.5 GHz, 16GB of RAM, and an Nvidia RTX 3060 GPU with 12GB of VRAM.

TABLE I
EXPERIMENTAL PARAMETERS

Parameter	Setting
Input Size	224x224
No. of training trials	5
No. of classes	100
Validation frequency	50
Train/Test ratio	70:30
Learning rate	0.001
Learning algorithm	SGDM
Epoch	32
Batch size	16

All the models were trained from scratch with a learning rate of 0.001, 32 epochs with a mini-batch size of 16, and a validation frequency of 50 using the Stochastic Gradient Descent with momentum (SGDM) optimizer. In addition, the dataset was split into a ratio of 70:30, with 70% of the samples used for training and the remaining 30% for testing purposes. All the pre-trained model architectures have an input of RGB (Red, Green, Blue) channel with a resolution size of 224x224, whereas AlexNet's input size is 227x277. Therefore, the input image was resized to the appropriate resolution according to the model's input requirement by converting the single-channel image into a three-channel image before feeding it into the model.

Besides, we also made some modifications at the output layers of pre-trained models to classify only 100 classes, as the architecture of these models was originally designed for

1000-class classification. To ensure the fairness and reliability of the experiment results, each model was trained five times with randomly initialized parameters, and the parameters of the models were not saved after each trial's training and testing process.

E. Experimental Setting

This section outlines the experimental settings. The first experiment demonstrates how incorporating batch normalization and parallel convolution layers that extract and learn features at different scales can improve the model's performance. Three variants of architecture were created, namely Architecture A, B, and C. Architecture A is the proposed architecture, while Architecture B removes the parallel convolution layer. Architecture C is identical to Architecture B, but batch normalization is further removed. Table II shows three of the architectural details.

TABLE II
PROPOSED ARCHITECTURES

Layer	Architecture A	Architecture B	Architecture C
1	Input 224x224x1	Input 224x224x1	Input 224x224x1
2	Conv, 32 kernel 3x3	Conv, 32 kernel 3x3	Conv, 32 kernel 3x3
	BatchNorm + ReLU	BatchNorm + ReLU	ReLU
	Max-Pool 2x2	Max-Pool 2x2	Max-Pool 2x2
3	Conv, 64 kernel 3x3	Conv, 64 kernel 3x3	Conv, 64 kernel 3x3
	BatchNorm + ReLU	BatchNorm + ReLU	ReLu
	Max-Pool 2x2	Max-Pool 2x2	Max-Pool 2x2
4	**Parallel Convolution layer**	Convolution, 64 kernel 3x3	Convolution, 64 kernel
	Conv 3x3(64 filters) & 5x5(32 filters), BatchNorm + ReLU depth	BatchNorm + ReLU	3x3
	concatenation	Max-Pool 2x2	ReLU
	Max-Pool 2x2		Max-Pool 2x2
5	**Parallel Convolution layer**	Convolution, 64 kernel 3x3	Convolution, 64 kernel
	Conv 3x3(64 filters) & 5x5(32 filters), BatchNorm + ReLU depth	BatchNorm + ReLU	3x3
	concatenation	Max-Pool 2x2	ReLu
	Max-Pool 2x2		Max-Pool 2x2
6	FC Layer 1: 1024	FC Layer 1: 1024	FC Layer 1: 1024
	Dropout: 0.5	Dropout: 0.5	Dropout: 0.5
	FC Layer 2:100	FC Layer 2:100	FC Layer 2:100
	SoftMax	SoftMax	SoftMax

The architecture that obtains the highest accuracy from the first experiment will be selected in the second experiment, and its performance will be compared with the existing pre-trained model architecture. Although all of the proposed architectures differ in design, they have an identical number of parameters, within 19.0 million, due to the presence of two fully connected layers with many neurons. Table III shows the information for both the proposed architectures and the pre-trained models.

TABLE III
MODEL ARCHITECTURE INFORMATION

Model Architecture	Input Size	Depth	Parameter (million))
GoogleNet [31]	224x224x3	22	7.0
AlexNet [32]	227x227x3	8	60.0
VGG-16 [33]	224x224x3	16	138.0
ResNet-50 [34]	224x224x3	50	25.6
Proposed (A, B, C)	**224x224x1**	**6**	**19.0**

III. RESULTS AND DISCUSSION

In this section, the experimental results of averaged accuracy and training duration are elaborated upon and discussed in detail.

A. Results of Experiment-1

Batch normalization is known to accelerate the training time of a model. However, the experiment result shows that architectures with batch normalization require more training time than Architecture C, which does not have batch normalization. Despite the longer training time, architectures with batch normalization exhibit higher recognition accuracy, whereas Architecture C, without batch normalization, has a lower accuracy rate.

TABLE IV
ARCHITECTURE A PERFORMANCE

Trial	Accuracy	Precision	Specificity	Recall	F1-Score	Train Time
1	91.50	93.38	99.91	91.50	91.13	3m53s
2	93.50	94.82	99.93	93.50	93.34	3m29s
3	92.50	93.96	99.92	92.50	92.16	3m49s
4	93.33	94.57	99.93	93.33	93.10	3m39s
5	92.50	93.66	99.92	92.50	92.18	3m33s
AVG	93.00	94.25	99.93	93.00	92.74	3m33s
STD	0.47	0.43	0.01	0.47	0.54	

TABLE V
ARCHITECTURE B PERFORMANCE

Trial	Accuracy	Precision	Specificity	Recall	F1-Score	Train Time
1	91.50	93.38	99.91	91.50	91.13	3m53s
2	92.83	93.85	99.93	92.83	92.47	3m23s
3	91.83	93.13	99.92	91.83	91.58	3m14s
4	92.00	93.38	99.92	92.00	91.64	3m15s
5	90.67	92.33	99.91	90.67	90.41	3m02s
AVG	91.77	93.21	99.92	91.77	91.45	3m25s
STD	0.78	0.56	0.01	0.78	0.75	

TABLE VI
ARCHITECTURE C PERFORMANCE

Trial	Accuracy	Precision	Specificity	Recall	F1-Score	Train Time
1	91.50	92.71	99.91	91.50	91.24	3m01s
2	89.67	91.62	99.90	89.67	89.06	2m52s
3	91.33	92.26	99.91	91.33	90.89	2m56s
4	89.67	92.23	99.90	89.67	89.04	2m55s
5	89.83	92.16	99.90	89.83	89.38	2m51s
AVG	90.40	92.20	99.90	90.40	89.92	2m55s
STD	0.93	0.39	0.01	0.93	1.06	

On the other hand, the parallel convolution layer requires more computation resources due to multiple convolution layers that extract and learn features at different scales simultaneously. Although Architecture A has parallel layers, the training time is identical to Architecture B and has also outperformed its accuracy by 1.23%. Overall, Architecture A obtained the highest average accuracy of 93.00%, while Architecture C required the least computation resources and had the lowest average accuracy of 90.40%.

B. Results of Experiment 2

Based on the experiment results shown in Table XI and Table XII, the highest average accuracy achieved is 93.00% by our proposed model, while the lowest average accuracy is 85.57% from GoogleNet. Regarding model training duration, both the AlexNet and VGG-16 have the longest training times, with an average training duration of 20 minutes.

TABLE VII
GOOGLENET PERFORMANCE

Trial	Accuracy	Precision	Specificity	Recall	F1-Score	Train Time
1	89.67	91.67	99.90	89.67	89.06	7m10s
2	85.67	90.46	99.76	85.67	84.70	7m32s
3	85.33	89.42	99.85	85.33	84.43	8m08s
4	86.17	89.92	99.86	86.17	85.45	7m40s
5	81.00	86.06	99.81	81.00	80.10	7m17s
AVG	85.57	89.51	99.84	85.57	84.75	8m07s
STD	3.09	2.10	0.05	3.09	3.19	

TABLE VIII
ALEXNET PERFORMANCE

Trial	Accuracy	Precision	Specificity	Recall	F1-Score	Train Time
1	90.50	92.15	99.90	90.50	90.02	20m59s
2	91.50	92.81	99.91	91.50	91.02	19m42s
3	90.33	91.74	99.90	90.33	89.43	20m44s
4	89.33	92.61	99.89	89.33	89.18	20m34s
5	90.33	92.04	99.90	90.33	89.81	20m13s
AVG	90.40	92.27	99.90	90.40	89.89	20m38s
STD	0.77	0.43	0.01	0.77	0.71	

TABLE IX
VGG-16 PERFORMANCE

Trial	Accuracy	Precision	Specificity	Recall	F1-Score	Train Time
1	90.83	92.43	99.91	90.83	90.19	19m23s
2	92.83	94.03	99.93	92.83	92.39	20m46s
3	89.67	91.67	99.90	89.67	88.61	20m29s
4	91.50	92.80	99.91	91.50	91.04	21m15s
5	92.50	93.46	99.92	92.50	91.92	19m48s
AVG	91.47	92.88	99.91	91.47	90.83	20m18s
STD	1.28	0.91	0.01	1.28	1.50	

TABLE X
RESNET-50 PERFORMANCE

Trial	Accuracy	Precision	Specificity	Recall	F1-Score	Train Time
1	92.00	92.98	99.92	92.00	91.44	14m53s
2	93.00	94.21	99.93	93.00	92.58	16m40s
3	92.67	93.39	99.93	92.67	92.16	16m14s
4	92.33	93.42	99.92	92.33	92.00	16m52s
5	92.83	93.95	99.93	92.83	92.37	14m38s
AVG	92.57	93.59	99.93	92.57	92.11	16m03s
STD	0.40	0.49	0.01	0.40	0.71	

TABLE XI
CLASSIFICATION ACCURACY COMPARISON

Model	Accuracy (%)					
	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	AVG
GoogleNet	89.67	85.67	85.33	86.17	81.00	85.57
AlexNet	90.50	91.50	90.33	89.33	90.33	90.40
VGG-16	90.83	92.83	89.67	91.50	92.50	91.47
ResNet-50	92.00	93.00	92.67	92.33	92.83	92.57
Proposed	**93.17**	**93.50**	**92.50**	**93.33**	**92.50**	**93.00**

TABLE XII
TRAINING DURATION COMPARISON

Model	Training Duration (minutes - seconds)					
	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	AVG
GoogleNet	07m38s	08m02s	08m08s	08m01s	08m48s	08m07s
AlexNet	20m59s	20m36s	20m29s	21m15s	19m48s	20m18s
VGG-16	19m23s	20m36s	20m29s	21m15s	19m48s	20m18s
ResNet-50	14m53s	16m40s	16m14s	16m52s	14m48s	16m03s
Proposed	03m18s	03m29s	03m49s	03m39s	03m33s	03m33s

Our proposed model is the fastest model, with an average training duration of 3 minutes and 33 seconds. The ResNet-50 is 50 layers deep with 25.6 million parameters, which is significantly fewer than the 60 million parameters in AlexNet and 138 million in VGG-16. Although it has more layers, ResNet-50 is faster than AlexNet and VGG-16 due to the use of skip connections, which helps reduce the training duration by making the model's parameters easier to optimize during the training process. Undoubtedly, increasing the depth of a model can lead to an improvement in classification accuracy, but it also increases the computation cost and complexity of the model, which requires larger training samples that result in longer training times in order to generalize well on test samples. ResNet-50 obtained an average accuracy of 92.57% but required more computation resources than our proposed architecture.

The VGG-16 model has 77 million more parameters than AlexNet, but both models have similar average training durations of about 20 minutes. However, it only outperforms AlexNet by 1.07%. These results indicate that the higher number of parameters and deeper depth in the VGG-16 model with small training samples may have contributed to its inability to obtain a higher accuracy rate. GoogleNet is faster than AlexNet, VGG-16, and ResNet-50, with an average training duration of 7 minutes and 32 seconds. Yet, it has the lowest average accuracy of 83.7%.

IV. CONCLUSION

This paper proposes a multi-scale convolutional neural network for in-air hand gesture signature recognition. The proposed architecture's performance is compared with existing pre-trained models such as GoogleNet, AlexNet, VGG-16, and ResNet-50 in the same experimental setting. Our proposed architecture demonstrated superior performance among the models in the experiments. Despite having the least number of depths, it had more parameters than GoogleNet due to the use of fully connected layers with a large number of neurons. The experiment showed that all the models had relatively similar average accuracy, ranging from 90% to 92%, except GoogleNet. Our proposed model not only achieved the highest average accuracy of 93.00% but also had the fastest training time with an average training duration of 3 minutes and 33 seconds compared to other models, making it more practical and efficient to be used for real-world applications.

Overall, the experiment results show that a shallow model with fewer layers and a large number of parameters can perform well on smaller training samples, while deep models with more layers tend to require larger training samples and more computation resources to obtain optimal performance due to their increased complexity. In addition, there is still room for improvement by integrating our proposed CNN architecture with recurrent neural networks to build a

convolutional recurrent neural network (C-RNN). This integrated architecture can learn and classify the entire hand gesture image sequence. Leveraging both architectures can potentially achieve even higher accuracy for in-air hand gesture signature recognition.

ACKNOWLEDGMENT

We acknowledge the Ministry of Higher Education (MOHE) for funding under the Fundamental Research Grant Scheme (FRGS) (FRGS/1/2021/ICT02/MMU/03/3).

REFERENCES

[1] A. Kholmatov and B. Yanikoglu, "Identity authentication using improved online signature verification method," *Pattern Recognit Lett*, vol. 26, no. 15, pp. 2400–2408, 2005, doi: 10.1016/j.patrec.2005.04.017.

[2] A. McCabe, J. Trevathan, and W. Read, "Neural Network-based Handwritten Signature Verification," *J Comput (Taipei)*, vol. 3, no. 8, 2008, doi: 10.4304/jcp.3.8.9-22.

[3] R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, "Dynamic signature recognition for automatic student authentication," IATED, 2015.

[4] S. Carlaw, "Impact on biometrics of Covid-19," *Biometric Technology Today*, vol. 2020, no. 4, pp. 8–9, 2020, doi: 10.1016/S0969-4765(20)30050-3.

[5] A. Buriro, R. Van Acker, B. Crispo, and A. Mahboob, "AirSign: A Gesture-Based Smartwatch User Authentication," *Proceedings - International Carnahan Conference on Security Technology*, vol. 2018-Octob, pp. 20–21, 2018, doi: 10.1109/CCST.2018.8585571.

[6] M. A. A. Haseeb and R. Parasuraman, "Wisture: RNN-based Learning of Wireless Signals for Gesture Recognition in Unmodified Smartphones," pp. 1–10, 2017, [Online]. Available: http://arxiv.org/abs/1707.08569

[7] L. G. Hafemann, R. Sabourin, and L. S. Oliveira, "Offline handwritten signature verification — Literature review," *2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA)*, vol. (pp. 1-8). 2017. doi: 10.1109/ipta.2017.8310112.

[8] H.-C. Moon, S. Jang, K. Oh, and K.-A. Toh, "An In-Air Signature Verification System Using Wi-Fi Signals," *Proceedings of the 4th International Conference on Biomedical and Bioinformatics Engineering*. pp. 133–138, 2017. doi: 10.1145/3168776.3168799.

[9] G. Li, L. Zhang, and H. Sato, "In-air Signature Authentication Using Smartwatch Motion Sensors," *IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*. pp. 386–395, 2021. doi: 10.1109/COMPSAC51774.2021.00061.

[10] R. Zhao, D. Wang, Q. Zhang, X. Jin, and K. Liu, "Smartphone-based Handwritten Signature Verification Using Acoustic Signals," *Proceedings of the ACM on Human-Computer Interaction*, vol. 5, no. ISS. pp. 1–26, 2021. doi: 10.1145/3488544.

[11] E. Guerra-Segura, A. Ortega-Pérez, and C. M. Travieso, "In-air Signature Verification System Using Leap Motion," *Expert Syst Appl*, vol. 165, no. 113797, 2021, doi: 10.1016/j.eswa.2020.113797.

[12] L. De Luisa, G. E. Hine, E. Maiorana, and P. Campisi, "In-Air 3D Dynamic Signature Recognition Using Haptic Devices," *International Workshop on Biometrics and Forensics (IWBF)*. IEEE, pp. 1–6, 2021. doi: 10.1109/iwbf50991.2021.9465089.

[13] K. Kancharla, V. Kamble, and M. Kapoor, "Handwritten Signature Recognition: a Convolutional Neural Network Approach," *International Conference on Advanced Computation and Telecommunication (ICACAT)*. IEEE, p. (1-5), 2018. doi: 10.1109/icacat.2018.8933575.

[14] W. Xiao and Y. Ding, "A Two-Stage Siamese Network Model for Offline Handwritten Signature Verification," *Symmetry (Basel)*, vol. 14, no. 6, p. 1216, 2022, doi: 10.3390/sym14061216.

[15] P. D. Hung, P. S. Bach, B. T. Vinh, H. T. Nguyen, and V. T. Diep, "Offline Handwritten Signature Forgery Verification Using Deep Learning Methods," *Smart Trends in Computing and Communications: Proceedings of SmartCom*. Springer Nature Singapore., p. (75-84), 2022. doi: 10.1007/978-981-16-9967-2_8.

[16] A. L. Hagstrom, R. Stanikzai, J. Bigün, and F. Alonso-Fernandez, "Writer Recognition Using Offline Handwritten Single Block Characters," *International Workshop on Biometrics and Forensics (IWBF)*. IEEE, p. (1-6), 2022. doi: 10.1109/iwbf55382.2022.9794466.

[17] S. S. Harakannanavar, J. H, A. C. N, K. Prashanth, and P. Hudedavar, "Biometric Trait: Offline Signature Identification and Verification Based on Multi-modal Fusion Techniques," *Journal of Positive School Psychology*, vol. 6, no. 4, pp. 2180–2191, 2022, [Online]. Available: https://journalppw.com/index.php/jpsp/article/view/3592

[18] A. Jain, S. K. Singh, and K. P. Singh, "Handwritten Signature Verification Using Shallow Convolutional Neural Network," *Multimed Tools Appl*, vol. 79, no. 27–28, pp. 19993–20018, 2020, doi: 10.1007/s11042-020-08728-6.

[19] Y. Zhou, J. Zheng, H. Hu, and Y. Wang, "Handwritten Signature Verification Method Based on Improved Combined Features," *Applied Sciences*, vol. 11, no. 13, p. 5867, 2021, doi: 10.3390/app11135867.

[20] G. Li and H. Sato, "Sensing In-Air Signature Motions Using Smartwatch: A High-Precision Approach of Behavioral Authentication," *IEEE Access*, vol. 10, pp. 57865–57879, 2022, doi: 10.1109/access.2022.3177905.

[21] Y. Guo and H. Sato, "Smartwatch In-Air Signature Time Sequence Three-Dimensional Static Restoration Classification Based on Multiple Convolutional Neural Networks," *Applied Sciences*, vol. 13, no. 6, p. 3958, 2023, doi: 10.3390/app13063958.

[22] S. Franceschini, M. Ambrosanio, V. Pascazio, and F. Baselice, "Hand Gesture Signatures Acquisition and Processing by Means of a Novel Ultrasound System," *Bioengineering*, vol. 10, no. 1, p. 36, 2023, doi: 10.3390/bioengineering10010036.

[23] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting Unreasonable Effectiveness of Data in Deep Learning Era," *Proceedings of the IEEE International Conference on Computer Vision*. pp. 843–852, 2017.

[24] Z. Y. Poo, C. Y. Ting, Y. P. Loh, and K. I. Ghauth, "Multi-Label Classification with Deep Learning for Retail Recommendation," *Journal of Informatics and Web Engineering*, vol. 2, no. 2, pp. 218–232, 2023, doi: 10.33093/jiwe.2023.2.2.16.

[25] L. Jia-Rou, K.W. Ng, and Y.-J. Yoong, "Face and Facial Expressions Recognition System for Blind People Using ResNet50 Architecture and CNN," *Journal of Informatics and Web Engineering*, vol. 2, no. 2, pp. 284–298, 2023, doi: 10.33093/jiwe.2023.2.2.20.

[26] W. H. Khoh, Y. H. Pang, and H. Y. Yap, "In-air Hand Gesture Signature Recognition: an iHGS Database Acquisition Protocol," *F1000Res*, p. 283, 2023, doi: 10.12688/f1000research.74134.2.

[27] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *International Conference on Machine Learning*. pmlr, pp. 448–456, 2015.

[28] W. H. Khoh, Y. H. Pang, and A. B. J. Teoh, "In-air hand gesture signature recognition system based on 3-dimensional imagery," *Multimed Tools Appl*, vol. 78, no. 6, pp. 6913–6937, 2018, doi: 10.1007/s11042-018-6458-7.

[29] J. C. Davis and A. F. Bobick, "The Representation and Recognition of Human Movement Using Temporal Templates," *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, p. (928-934), 2002. doi: 10.1109/cvpr.1997.609439.

[30] Md. A. R. Ahad, J. K. Tan, H. Kim, and S. Ishikawa, "Motion History image: Its Variants and Applications," *Mach Vis Appl*, vol. 23, no. 2, pp. 255–281, 2010, doi: 10.1007/s00138-010-0298-4.

[31] C. Szegedy *et al.*, "Going Deeper with Convolutions," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 1–9, 2015. [Online]. Available: https://www.cv-foundation.org/openaccess/content_cvpr_2015/html/Szegedy_Going_Deeper_With_2015_CVPR_paper.html.

[32] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Commun ACM*, vol. 60, no. 6, pp. 84–90, 2012, doi: 10.1145/3065386.

[33] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv.org*. 2015. [Online]. Available: https://arxiv.org/abs/1409.1556

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 770–778, 2016.