ICIC Express Letters
Volume **16**, Number **5**, May **2022**

 ISSN 1881-803X
pp. **529–536**

AN END-TO-END THAI FINGERSPELLING RECOGNITION FRAMEWORK WITH DEEP CONVOLUTIONAL NEURAL NETWORKS

Siriwiwat Lata and Olarik Surinta*

Multi-agent Intelligent Simulation Laboratory (MISL)
Department of Information Technology
Faculty of Informatics
Mahasarakham University
Khamriang Sub-District, Kantarawichai District, Mahasarakham 44150, Thailand
61011261004@msu.ac.th; *Corresponding author: olarik.s@msu.ac.th

Received September 2021; accepted November 2021

Abstract. *The WHO reports that approximately 34 million people worldwide experience deafness and hearing loss. In 2050, these will increase to affect 900 million people. It is essential to communicate with the hearing impaired in hand sign language. This paper proposes an end-to-end fingerspelling recognition framework of the Thai sign language based on deep convolutional neural networks (CNNs). We divided our framework into two processes. In the first process, we focus on the detection of hands using the YOLOv3 objection detection framework. In the second process, we propose using five CNN architectures, MobileNetV2, DenseNet121, InceptionResNetV2, NASNetMobile, and EfficientNetB2, to create the most robust model that provides high recognition performance. Hence, we evaluated the proposed framework to detect and recognize three Thai fingerspelling (TFS) datasets: TFS, KKU-TFS, and Unseen-TFS. We found that YOLOv3 showed a high precision value on the TFS dataset. However, the worst performance was found with KKU-TFS and Unseen-TFS datasets. Also, our proposed framework could not detect hands from only one image on the KKU-TFS and Unseen-TFS datasets. Therefore, we also examined the CNN architectures to recognize the 1-stage Thai fingerspelling images. The experimental results showed that DenseNet121 obtained an accuracy of 93.99% on the TFS dataset and 90.40% on the KKU-TFS dataset.*
Keywords: Thai fingerspelling recognition, Convolutional neural network, Hand detection, Hand gesture recognition, YOLO architecture

1. **Introduction.** Humans are considered as a valuable resource. Various factors such as congenital disabilities and accidents that affect human beings can result in imperfections in their physical condition causing disabilities, such as congenital disabilities and accidents. The World Health Organization (WHO) reported that over 1 billion people, or approximately 15% of the world's population are disabled. Globally, more than 1.5 billion people experience some decline in their hearing capacity during their life course, of whom at least 430 million will require care [1]. Therefore, it is assumed that these groups need to communicate with each other using sign language, which is likely to cause communication problems with ordinary people.

Many researchers have developed technological facilities for the disabled especially those with hearing loss in order to improve the efficiency of sign language communication, using, for example, sensor technology [3] and depth cameras [3]. Because of the expensive technology involved in these approaches, researchers have focused on improving image and video recognition using artificial intelligence techniques. Zhao et al. [4] proposed using only a webcam to communicate with disabled people. These techniques have detected

DOI: 10.24507/icicel.16.05.529

hand movement and interpreted hand gestures that disabled people use to express to ordinary people. Due to the complexity of sign language, researchers designed and developed processes for static-sign language recognition, which means one gesture has only one meaning [6]. Pariwat and Seresangtakul [6] and Nakjai and Katanyukul [7] created a static-sign language image dataset. The image collections are created by setting the scene and assigning the background color to contrast with human skin color. However, a black suit is also worn to provide contrast and facilitate detection of the hand area.

In actual situations, sign language interpretation may not be in the setup area. It harmfully impacts the algorithm that is then unable to detect and recognize the hand correctly. However, deep learning algorithms, such as single shot detector (SSD) [8] and YOLO [9], can be proposed to detect the hand area without a black background, making it possible to detect hands in any environment.

Hand detection. Le et al. [10] proposed YOLO architecture using the spatial transfer connection (STC) for resolving the hands from a complex environment. With the STC operation, a small convolution operation with the size of 1×1 was applied to the input features in each convolution layer to reduce the output feature. These two YOLO architectures were evaluated based on the intersection over union (IoU) value on the hand dataset. The result showed that the YOLO with STC operation outperformed the original YOLO architecture. Bose and Kumar [11] used Faster R-CNN with the InceptionV2 architecture as a backbone module. The Faster R-CNN method was proposed to find out the region proposals and classify the hands from the whole image. This method was trained using three gradient descent optimization techniques (Momentum, RMSprop, and Adam) and training with 35,000 epochs. The result showed that the Adam optimization algorithm performed better than momentum and RMSprop optimizers.

Hand gesture recognition using convolutional neural network. Yasir et al. [12] proposed a convolutional neural network for recognition of the Bangla sign language. First, the leaf motion device was used to track the hand motion. Second, the continuous frames were then segmented using the hidden Markov model. The time-series data were generated at this state. Finally, the time-series data was sent to train with the CNN method. The experimental result showed that the error rate was decreased from 5% to around 1.5% in only three epochs. Rao et al. [13] presented a multi-stage CNN for Indian sign language gesture recognition. First, the feature extraction module consisted of four convolution layers, four rectified linear unit (ReLU) activation functions, and two stochastic pooling layers. Second, the classification module comprised one dense layer followed by the ReLU layer and a softmax function which was the last layer of the network. The output of the proposed CNN model had 200 units. The multi-stage CNN model achieved a recognition accuracy of 92.88%.

This paper provides an end-to-end Thai fingerspelling (TFS) recognition framework that focuses on 1-stage Thai fingerspelling images using deep learning techniques. The framework is divided into two main parts: hand detection and 1-stage Thai fingerspelling recognition. We examined the hand detection method on the TFS dataset that included 7,200 images. Also, we tested the 1-stage TFS recognition on two datasets, Unseen-TFS and KKU-TFS, which contain 300 and 375 images, respectively. The experimental results revealed that the proposed framework achieved a high mAP value and a successful recognition result.

This paper is organized in the following way. Section 2 presents the framework of the end-to-end Thai fingerspelling recognition. Section 3 describes the fingerspelling datasets used in the experiments. The experimental results are explained in Section 4. The conclusion and future work are presented in Section 5.

2. End-to-End Thai Fingerspelling Recognition Framework.

This section introduces the end-to-end Thai fingerspelling recognition framework that mainly focuses on

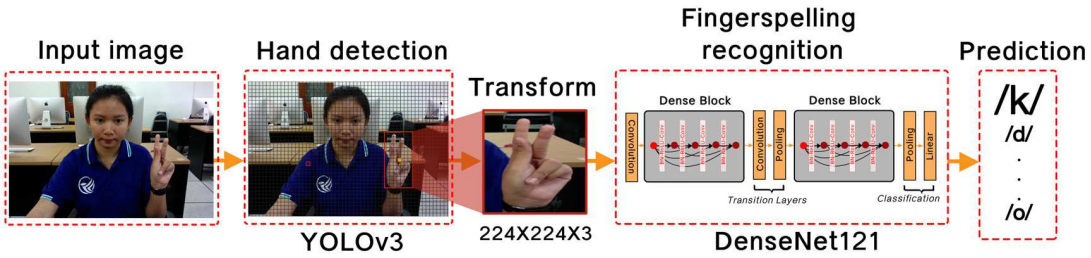


FIGURE 1. The proposed framework of the 1-stage Thai fingerspelling recognition

1-stage Thai fingerspelling images, as shown in Figure 1. The details of the proposed framework are described as follows.

2.1. **Hand detection using YOLOv3.** Redmon et al. [9] proposed YOLO (you only look once) for real-time object detection and could process video in real time at 45 frames per second, which was implemented and resulted in improving the object detection and tracking faster [14]. YOLOv3 consists of two parts: feature extraction and bounding box prediction. In the feature extraction part, 53 convolutional layers are followed by two fully connected layers for extracting the feature from the grid cell. In the bounding box prediction part, the YOLOv3 algorithm starts by dividing the image into a small grid cell. Then, the YOLOv3 algorithm will predict the bounding boxes at three different scales from each grid cell and return the class probabilities for those boxes. Finally, these probabilities are used to predict if the grid cell contains target objects.

2.2. **1-stage Thai fingerspelling recognition with deep convolutional neural networks.**

DenseNet121. Huang et al. [15] proposed the DenseNet architecture designed to connect a current layer to other layers in a feed-forward layer. In DenseNet, the current layer obtained additional feature maps from all earlier layers. Thus, the feature maps from previous layers were then concatenated with the current feature maps. Further, they connected until the last layer. The DenseNet121 architecture consists of five layers: convolution, max-pooling, dense block, transition, and classification. The advantage of DenseNet architecture was that it collected all the knowledge of earlier layers.

MobileNetV2. Sandler et al. [16] invented the MobileNetV2 architecture that was an extended version of the MobileNet. MobileNetV2 was designed based on inverted residual and linear bottlenecks. Depthwise convolution operations were proposed to create a lightweight architecture because it allowed factorizing the convolution layer into two separate layers: depthwise and pointwise convolution layers. As a result, the model was relatively small and reduced the chance of over-fitting.

InceptionResNetV2. Szegedy et al. [17] proposed InceptionResNetV2, which was improved from the architecture of Inceptionv3. The InceptionResNetV2 architecture is better than Inceptionv3 in its accuracy rate and computation time because it reduced the size of the feature map by factorizing the convolutional layer. It contains three main blocks: stem block, Inception-ResNet block, and reduction block.

NASNetMobile. Zoph et al. [18] proposed the neural architecture search, called NASNet architecture. The main architecture included two cells: normal cell and reduction cell. To create the normal cell, a recurrent neural network (RNN) was proposed to train with reinforcement learning and generate the normal cell by selecting from a set of convolutional operations, such as depthwise-separable, max pooling, average pooling, and dilated convolution. For the reduction cell, the stride operation was used to decrease the size of the feature maps on both height and width.

EfficientNetB2. Tan and Le [19] invented the EfficientNet architecture that included eight models (B0, B1, ..., B7). The main idea of the EfficientNet is to increase the scale

of the CNN models, including width, depth, resolution, and compound. We propose to use EfficientNetB2 that scaled up using a different compound coefficient.

3. Fingerspelling Datasets.

3.1. **TFS dataset.** This research focused only on the one-stage fingerspelling of Thai sign language that contained 15 signs (/k/, /d/, /t/, /n/, /b/, /p^{h}/, /f/, /m/, /y/, /r/, /l/, /w/, /s/, /h/, and /o/).

The TFS dataset used in the experiments included 7,200 images, 480 images in each class, and $1280 * 720$ pixels resolution. We collected the Thai fingerspelling consonants with the support of 14 undergraduate students in the Special Education Department who have experience using Thai sign language and 50 undergraduate students with no experience in using Thai sign language. We recorded the images with both complex and non-complex backgrounds. Examples of the TFS dataset are shown in Figure 2.

(a) (b)

FIGURE 2. Examples of the 1-stage Thai fingerspelling consonants that recorded in (a) non-complex and (b) complex backgrounds

3.2. **Unseen Thai fingerspelling dataset (Unseen-TFS).** We also propose the unseen Thai fingerspelling dataset, called Unseen-TFS, to evaluate Thai fingerspelling detection and recognition algorithms. Remarkably, four volunteer undergraduate students with no experience using Thai sign language supported us in completing this dataset. The Unseen-TFS dataset consisted of 300 images of 15 consonants, as shown in Figure 3(a).

3.3. **KKU Thai fingerspelling dataset (KKU-TFS).** The KKU-TFS was proposed by Pariwat and Seresangtakul [6] in 2017. It contains 15 signs of 1-stage Thai fingerspelling and has 375 images, five images in each sign. Each image has a size of $250 * 288$ pixels resolution recorded from five volunteers. Samples of the KKU-TFS dataset are shown in Figure 3(b).

(a) (b)

FIGURE 3. Examples of the TFS datasets: (a) Unseen-TFS and (b) KKU-TFS datasets

4. **Experimental Results.** We evaluated experiments on the TFS datasets, including Thai fingerspelling (TFS), Unseen-TFS, and KKU-TFS. All experiments were performed using TensorFlow deep learning framework that was running on Intel(R) Core-i5 9400F CPU @ 2.90GHz, 32GB RAM, and GPU NVIDIA GeForce GTX 1080. The experimental results are explained as follows.

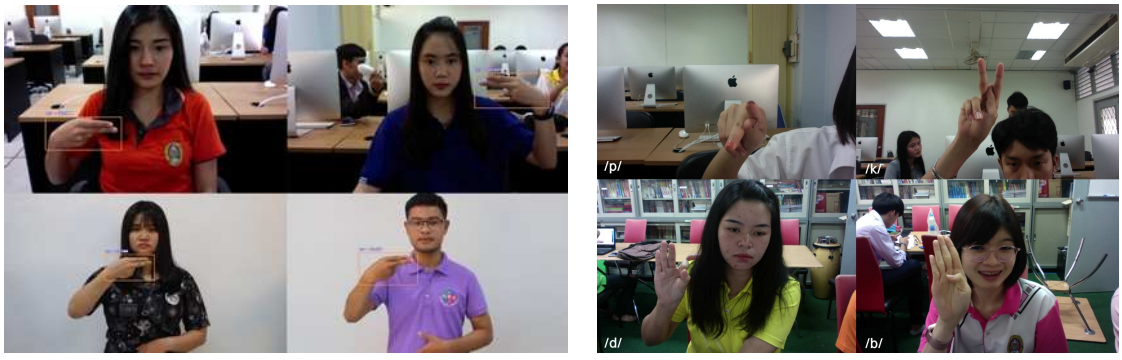
4.1. **Experiments on hand detection using YOLOv3.** This experiment mainly focused on using the YOLOv3 real-time object detection framework for hand detection. We concentrated on training the network to identify the location of an object which is a hand in an image. We used the mean average precision (mAP) that compares the output of the network and the ground-truth bounding box to evaluate a model. A higher mAP value indicates better detection performance.

We trained the YOLOv3 model on the TFS dataset that included 4,320 training images and was evaluated on 2,880 test images. The sizes of the input images were $1280*720$ and $1080*720$ pixels resolution. The parameter settings used to train the YOLOv3 model were as follows: downsample and the batch size were 32 and 4, intersection over union (IoU) and non-maximum suppression were 0.5 and 0.5, and the number of training iterations was 100 epochs.

Table 1 showed the high average precision value achieved from the YOLOv3 model with mAP = 0.9787. Hence, the highest mAP value appeared in class /h/ with the mAP value of one and also the lowest value appeared in class /t/ with the mAP value of 0.9312. However, most of the results could not detect the hand from the complex background, as shown in Figure 4(b).

Table 1. Performance evaluation of the YOLOv3 on the TFS dataset

Classes	mAP	Classes	mAP
/k/	0.9808	/y/	0.9819
/d/	0.9688	/r/	0.9635
/t/	0.9312	/l/	0.9845
/n/	0.9583	/w/	0.9844
/b/	0.9952	/s/	0.9948
/p^h/	0.9844	/h/	1.0000
/f/	0.9792	/o/	0.9887
/m/	0.9851	**Overall**	**0.9787**



(a) (b)

Figure 4. Illustration of the output that was detected using the YOLOv3: (a) The correct hand detection and (b) not detecting hand

Moreover, the longest (and therefore most expensive) computation time spent while training the YOLOv3 was 77 hours because we trained the YOLOv3 on 4,320 training images and used a GPU device that did not have high computational capacity. However, we suggest training the YOLOv3 on the high computational GPU device and CPU to decrease the computation time. Therefore, if the mAP value is needed, there is still an opportunity to improve by training more example images, increasing training iteration, and proposing novel data augmentation techniques.

4.2. **Experiments of the CNN architectures on 1-stage fingerspelling.** We performed the transfer learning technique for the CNN experiments to train on the 1-stage fingerspelling images using five CNN models, including MobileNetV2, DenseNet121, InceptionResNetV2, NASNetMobile, and EfficientNetB2. We also concentrated on operating with a small training set. Hence, we selected the size of the training set with 60%, 50%, 40%, and 30%, which was the 4,320, 3,600, 2,880, and 2,160 images, respectively. For the validation, we tested the YOLOv3 five times and reported the accuracy and standard deviation. The validation and test sets used in the experiments were 720 and 2,880 images, respectively.

In Table 2, we present the experimental results with five CNN models. The results showed that all CNN architectures obtained accuracy above 90%, except the NASNetMobile that gave 89% when training with 30% of the training set. At 60% of the training set, MobileNetV2 and DenseNet121 performed at the highest accuracy which was above 99% on the validation set.

TABLE 2. Performance evaluation of the CNN architectures on the validation and test sets when training with the different numbers of training set

CNNs	**Validation and test accuracy (%) with different numbers of training set**							
	60%		**50%**		**40%**		**30%**	
	Valid	**Test**	**Valid**	**Test**	**Valid**	**Test**	**Valid**	**Test**
MobileNetV2	**99.47 ±0.003**	98.02	97.77 ±0.002	96.91	97.38 ±0.004	95.71	96.17 ±0.006	**93.86**
DenseNet121	99.27 ±0.009	**98.04**	**98.55 ±0.004**	**97.96**	**98.16 ±0.008**	**96.57**	**96.90 ±0.007**	92.35
InceptionResNetV2	97.29 ±0.004	96.62	96.51 ±0.005	96.19	97.00 ±0.006	95.21	93.46 ±0.002	92.80
NASNetMobile	97.09 ±0.007	95.23	94.72 ±0.009	94.69	94.82 ±0.001	92.81	89.78 ±0.010	89.00
EfficientNetB2	93.38 ±0.005	93.09	92.30 ±0.004	92.11	91.68 ±0.001	91.96	90.15 ±0.004	90.06

We found that both MobileNetV2 and DenseNet121 architectures presented significant results when training with a small amount training data. These two models obtained an accuracy of 93.86% and 92.35%, respectively.

4.3. **Experiments of the end-to-end Thai fingerspelling framework.** We considered the results as shown in Table 2, which reveals that the DenseNet121 and MobileNetV2 outperformed other CNN architectures. The detection and recognition results are presented in Table 3.

TABLE 3. The YOLOv3 detection value (mAP) and the CNNs recognition accuracies (%) of 1-stage Thai fingerspelling experiments on the TFS datasets

Processes	**Methods**	**TFS datasets**		
		TFS	**KKU-TFS**	**Unseen-TFS**
Detection	YOLOv3	0.9787	0.6553	0.7944
Recognition	DenseNet121	93.99%	90.40%	82.00%
	MobileNetV2	92.81%	65.33%	74.67%

For hand detection, we trained the YOLOv3 model with 1-stage Thai fingerspelling images and achieved the mAP values of 0.9787, 0.6553, and 0.7944 on the TFS, KKU-TFS, and Unseen-TFS, respectively. However, the mAP value of the KKU-TFS data was relatively low, but it did detect hands in all 375 images. Consequently, the YOLOv3 was not detected in only one image in the Unseen-TFS dataset from 300 images and was not detected in six images from the 2,880 test images of the TFS dataset.

For 1-stage Thai fingerspelling recognition, we received the detected result of the hand image from YOLOv3 directly and then sent it to the CNN models for recognition. In this case, the detected hand images may localize at not the exact location as in the ground-truth bounding box. The result showed that the recognition accuracy of the TFS dataset was decreased to 93.99% and 92.81% when using DenseNet121 and MobileNetV2, respectively. The Unseen-TFS dataset with these CNN models (DenseNet121 and MobileNetV2) achieved low accuracy performance with only 82% and 74.67%, respectively. Therefore, we conclude that the DenseNet121 can cope well with all TFS datasets, especially with the Unseen-TFS dataset.

5. **Conclusion.** This paper aims to introduce an end-to-end Thai fingerspelling framework that performs hand detection and 1-stage fingerspelling of Thai sign language using deep convolutional networks. We first proposed to use state-of-the-art YOLOv3 for training on the Thai fingerspelling (TFS) datasets, Thai fingerspelling (TFS), KKU-TFS, and Unseen-TFS, to detect and localize the hand. Second, we trained five convolutional neural networks (CNNs), consisting of MobileNetV2, DenseNet121, InceptionResNetV2, NASNetMobile, and EfficientNetB2. The main purpose was to decrease the training examples while training the CNN models. So, we experimented with a small training set (60%, 50%, 40%, and 30%). The results showed that DenseNet121 and MobileNetV2 outperformed other CNN models. MobileNetV2 and DenseNet models, when training with only 30% of the training set, obtained an accuracy above 92%. However, when training with 60%, we obtained an accuracy of 98% from these two CNN models. Finally, we evaluated our end-to-end Thai fingerspelling framework on three TFS datasets. We found that our framework could detect hands from the image quite well and gave a high mAP value on the TFS dataset. However, the mAP value significantly decreased when evaluated on the KKU-TFS and Unseen-TFS datasets. Furthermore, our framework still detects hands at the correct location and our framework showed an accuracy above 90% on TFS and KKU-TFS datasets and slightly reduced accuracy to 82% on the Unseen-TFS dataset.

In the future, it will be important to increase the performance of our end-to-end Thai fingerspelling recognition framework, including recognition of multi-stage Thai fingerspelling, or recognition of hand sign from the video (called action recognition). Another direction for future work will be to consider the training of CNN models with limited training set size and still provide high recognition accuracy. In addition, we will apply the CNN to extracting the features [20], called the deep feature extraction method.

Acknowledgments. This research project was financially supported by Mahasarakham University. We would like to thank all participants from Rajabhat Mahasarakham University for their time and interest in contributing to this research. Without engagement from participants in collecting the dataset process, this research would not be possible.

REFERENCES

[1] World Health Organization, *World Report on Hearing*, Geneva, Switzerland, 2021.

[2] D. Bajpai, Two way wireless data communication and American sign language translator glove for images text and speech display on mobile phone, *The 5th International Conference on Communication Systems and Network Technologies (ICCSNT)*, pp.578-585, 2015.

[3] Q. V. Hoang, T. H. Le and S. C. Huang, An improvement of RetinaNet for hand detection in intelligent homecare systems, *International Conference on Consumer Electronics-Taiwan (ICCE-Taiwan)*, pp.1-2, 2020.

[4] J. Yang, Y. Zhao, J. C. W. Chan and C. Yi, Hyperspectral image classification using two-channel deep convolutional neural network, *International Geoscience and Remote Sensing Symposium (IGARSS)*, pp.5079-5082, 2016.

[5] S. Salian, Proposed system for sign language recognition, *International Conference on Computation of Power, Energy Information and Commuincation (ICCPEIC)*, pp.58-62, 2017.

[6] T. Pariwat and P. Seresangtakul, Thai finger-spelling sign language recognition using global and local features with SVM, *The 9th International Conference on Knowledge and Smart Technology (KST)*, pp.116-120, 2017.

[7] P. Nakjai and T. Katanyukul, Hand sign recognition for Thai finger spelling: An application of convolution neural network, *Journal of Signal Processing Systems*, vol.91, no.2, pp.131-146, 2019.

[8] K. Liu and N. Kehtarnavaz, Real-time robust vision-based hand gesture recognition using stereo images, *Journal of Real-Time Image Processing*, vol.11, no.1, pp.201-209, 2016.

[9] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, You only look once: Unified, real-time object detection, *The Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.779-788, 2016.

[10] T. H. Le, D. W. Jaw, I. C. Lin, H. B. Liu and S. C. Huang, An efficient hand detection method based on convolutional neural network, *The 7th International Symposium on Next Generation Electronics (ISNE)*, pp.1-2, 2018.

[11] S. R. Bose and V. S. Kumar, Hand gesture recognition using faster R-CNN Inception V2 model, *Advances in Robotics (AIR)*, pp.1-6, 2019.

[12] F. Yasir, P. W. C. Prasad, A. Alsadoon, A. Elchouemi and S. Sreedharan, Bangla sign language recognition using convolutional neural network, *International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, pp.49-53, 2017.

[13] G. A. Rao, K. Syamala, P. V. V. Kishore and A. S. C. S. Sastry, Deep convolutional neural networks for sign language recognition, *Conference on Signal Processing and Communication Engineering Systems (SPACES)*, pp.194-197, 2018.

[14] T. L. Dang, G. T. Nguyen and T. Cao, Object tracking using improved Deep_Sort_YOLOv3 architecture, *ICIC Express Letters*, vol.14, no.10, pp.961-969, 2020.

[15] G. Huang, Z. Liu, L. Van Der Maaten and K. Q. Weinberger, Densely connected convolutional networks, *The Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.4700-4708, 2018.

[16] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L. C. Chen, MobileNetV2: Inverted residuals and linear bottlenecks, *The Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.4510-4520, 2018.

[17] C. Szegedy, S. Ioffe, V. Vanhoucke and A. A. Alemi, Inception-v4, Inception-ResNet and the impact of residual connections on learning, *The 31st AAAI Conference on Artificial Intelligence*, pp.4278-4284, 2017.

[18] B. Zoph, V. Vasudevan, J. Shlens and Q. V. Le, Learning transferable architectures for scalable image recognition, *The Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.8697-8710, 2018.

[19] M. Tan and Q. V. Le, EfficientNet: Rethinking model scaling for convolutional neural networks, *International Conference on Machine Learning (ICML)*, pp.6105-6114, 2019.

[20] J.-Y. Zhao, J. Gong, S.-T. Ma and Z.-M. Lu, Curvature gray feature decomposition based finger vein recognition with an improved convolutional neural network, *International Journal of Innovative Computing, Information and Control*, vol.16, no.1, pp.77-90, 2020.