ICIC Express Letters
Part B: Applications
Volume **13**, Number **9**, September **2022**

ICIC International ©2022 ISSN 2185-2766
pp. 949–957

DYNAMIC FINGERSPELLING RECOGNITION FROM VIDEO USING DEEP LEARNING APPROACH: FROM DETECTION TO RECOGNITION

Siriwiwat Lata[1], Sirawan Phiphitphatphaisit[2], Sarayut Gonwirat[3] and Olarik Surinta[1,*]

[1]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

[2]Department of Information Systems
Faculty of Business Administration and Information Technology
Rajamangala University of Technology Isan
150 Srichan Road, Mueang District, Khon Kaen 30000, Thailand
sirawan.ch@rmuti.ac.th

[3]Department of Computer and Automation Engineering
Faculty of Engineering and Industrial Technology
Kalasin University
62/1, Kasetsomboon Road, Mueang District, Kalasin 46000, Thailand
Sarayut.go@ksu.ac.th

Received January 2022; accepted March 2022

Abstract. *The World Health Organization found that more than 34 million people suffer from hearing loss and these people need to use sign language to communicate. Hence, the sign language recognition system is proposed to communicate with hearing loss people and others. In this paper, we aim to propose an end-to-end system to recognize the dynamic Thai fingerspelling from video. The proposed system includes two main processes. First, we use the YOLOv5 algorithm for the human detection task. Subsequently, a uniform distribution method is proposed to select the robust frames before applying robust frames to the detection algorithm. Second, we propose dynamic fingerspelling recognition that consists of two deep learning architectures: convolutional neural network (CNN) and long short-term memory (LSTM). We then combine CNN and LSTM, called CNN-LSTM architecture, followed by the recognition block. The recognition block comprises dropout, global average pooling, and softmax layers. For the CNN architectures, we evaluated three CNNs: MobileNetV2, ResNet50, and DenseNet201. We found that the proposed ResNet50-LSTM architecture achieved an accuracy of 88.42% on the test set of the dynamic Thai fingerspelling dataset and also prevented the overfitting problem.*
Keywords: Dynamic fingerspelling recognition, Convolutional neural network, Long short-term memory, Sequence pattern, Dynamic Thai fingerspelling dataset

1. **Introduction.** Language is essential for the communication of people worldwide. According to a World Health Organization survey in 2021 [1], over 34 million people worldwide suffer from hearing loss, and the number of people with hearing loss is increasing. These people need to use sign language for primary communication, thus resulting in the development of technology to learn sign language from the movements of the hands and arms to convey meaning. The sign language of each country around the world has a different identity.

DOI: 10.24507/icicelb.13.09.949

Deep learning research proposes a communication tool for effective communication between persons with disabilities and people without disabilities. Islam et al. [2] proposed deep learning to recognize English sign language. Their research used the convolutional neural network (CNN) to extract features of hand images and learn them with the support vector machine (SVM) method. Phong and Ribeiro [3] used a capsule network to detect hands and recognize them at the same time. However, there are different methods to find the region of interest (ROI), such as Faster R-CNN and YOLO [4,5]. These methods use CNN as the backbone, giving the algorithm to find ROI and recognize that particular ROI. In [3], their method is a static recognition of hand signs, which means that their proposed method can recognize from only one image. It is not possible to recognize a dynamic hand sign.

Related work. In 2016, Chansri and Srinonchat [6] presented a method for detecting hands from complex backgrounds based on the depth image captured by the Kinect sensor for high brightness value objects close to the Kinect sensor. It can detect the hand area because hands are closest to the Kinect sensor. The hand area was then extracted using a histogram of oriented gradients (HOG) and recognized by a neural network.

In 2017, Pariwat and Seresangtakul [7] proposed a method to recognize Thai sign language by performing only on the static hand sign language with a total of 15 Thai symbols. The Thai sign language data used in the test were hand-only images taken against a blue background, allowing segment hands by transforming the color space from RGB to HSV and then transforming HSV to grayscale. Hence, images were transformed from grayscale into binary images by selecting a threshold value of 0.45 to extract the hand area. Then, global and local features were extracted. Finally, all the features were learned using the SVM method with the RBF kernel and achieved with 91.20% accuracy.

In 2019, Nakjai and Katanyukul [8] presented a method of hand sign recognition for Thai fingerspelling. Their research used a hand extraction method by transforming RGB color images to YCbCr. Then, the skin region was examined by the brightness values from the chroma channels. Then, extract the hand area and learn with CNN, which can recognize 25 classes. The results showed a recognition accuracy of 91.26%.

The limitation of Thai fingerspelling sign language is static hand sign which is only 15 Thai alphabets [7] and in a recognition process designed to recognize only from one image so that it is unable to recognize dynamic hand signs as it requires more than one input image.

In 2020, Sugandi et al. [9] proposed hand gesture recognition using a back-propagation neural network (BPNN) based on visual tracking in a real-time environment. They used skin color detection based on a YCbCr color filter to extract the hand region from the background and then converted it to grayscale and binary images to speed up the processing time. The hand feature from the binary image of the hand region was divided into six regions. Finally, six regions of the hand feature of each gesture became the input data of the BPNN. The results showed that the BPNN achieved an accuracy of 86.67%.

In 2021, Pariwat and Seresangtakul [10] designed a Thai sign language recognition system that could recognize dynamic hand signs by receiving video inputs, which can recognize Thai hand sign in up to 42 Thai symbols. Moreover, Nakjai and Katanyukul [11] designed an automatic Thai finger spelling transcription system that can transcribe Thai sign language from the video by being able to classify Thai sign language with 20 Thai characters and 20 vowels. Their system consists of three steps: alphabet separation, sign recognition, sign-sequence classification.

In this paper, an automated end-to-end fingerspelling recognition system is proposed to solve human detection and dynamic fingerspelling recognition tasks. First, we select robust frames from the video using the uniform distribution method and send those frames to detect humans using the YOLOv5 algorithm automatically. The sequence frames are

given to extract the sequence pattern and temporal features using CNN and LSTM architectures. For that, we combine state-of-the-art CNN with an LSTM network, called CNN-LSTM, including MobileNetV2-LSTM, ResNet50-LSTM, and DenseNet201-LSTM. Further, we add a recognition block containing; 1) a dropout layer to prevent the overfitting, 2) a global average pooling layer to average output and reduce the dimension of feature maps, and 3) the softmax activation function to recognize dynamic fingerspelling from video. The proposed system also enhances the accuracy and computation time. In addition, we collected 3,025 videos of dynamic Thai fingerspelling, consisting of 42 Thai alphabets, that were taken from 14 volunteers who are proficient in Thai sign language.

The rest of this paper is organized as follows. Section 2 presents the end-to-end system of dynamic fingerspelling recognition. The Thai fingerspelling dataset is described in Section 3. The experimental results are presented in Section 4. The conclusion and future direction are concluded in the last section.

2. **Dynamic Fingerspelling Recognition System Architecture.** We divided the proposed architecture into three main parts: frame selection, human detection, and dynamic fingerspelling recognition. The proposed architecture is shown in Figure 1 and the details of the three parts are in the following sections.

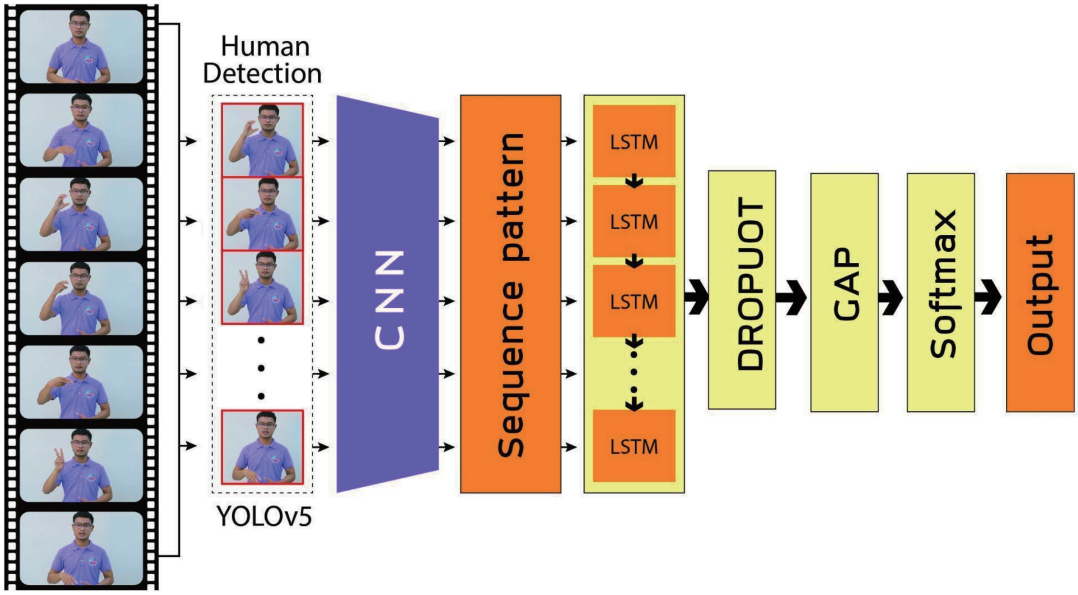

FIGURE 1. Illustration of the dynamic fingerspelling recognition system

2.1. **Frame selection.** Generally, videos were recorded with unequal length. In this paper, we first cut frames at the start and end of the video with 10% and 5%, respectively, because these parts contain unnecessary information. Second, we computed the jump ratio over the video frame using the following equation: *jump ratio* = *floor*(*No. of all frames/No. of selected frames*), where *No. of selected frames* in our experiment is 32. For example, if the jump ratio was 5, we used frames no. $1, 6, 11, \ldots, N$, where N is the last frame. Finally, we randomly selected only 32 frames using the uniform distribution method, in which the probability of each frame is equal chances to select.

2.2. **Human detection.** To build an end-to-end fingerspelling recognition system, this paper first aims to detect humans from the videos using a deep learning approach. We proposed to use the fast and accurate YOLOv5 (You Only Look Once) algorithm because we required an algorithm that detects humans in a real-time condition. In YOLOv5

[12], the latest update algorithm, the pre-trained model[1] that was trained on the COCO dataset was proposed to detect the human from the video. In this paper, we did not experiment on human detection. We only applied the YOLOv5 algorithm.

As shown in Figure 1, after we extracted images from the video, we applied the YOLOv5 algorithm to detecting humans as the region of interest (ROI) and sending that ROI to the CNN model.

2.3. **Dynamic fingerspelling recognition.** Dynamic fingerspelling recognition combines two deep learning approaches: CNN and RNN. The details of the deep learning approaches are described as follows.

CNN. We used different architectures of CNN: MobileNetV2, ResNet50, and DenseNet201, to extract robust deep features from a short video. In this research, the video was first extracted into a single frame (image) and then given to the CNNs to extract the spatial features from the image. We briefly described three CNN architectures in the following section.

MobileNetV2. We used the lightweight MobileNetV2 architecture in our experiments. MobileNet was invented for mobile visual applications [13] because it used depthwise separable convolution to reduce the number of CNN parameters. In addition, MobileNetV2 was designed based on an inverted residual structure and linear bottleneck. The bottleneck block contains three layers (1×1 conv2D with ReLU6, depthwise convolution with ReLU6, and linear 1×1 conv2D) and each layer has the expansion factor that expands the convolutional layers. Further, the residual was proposed to connect two bottleneck layers, like ResNet architecture.

ResNet50. Plain architectures, such as VGGNet and AlexNet, could have problems of vanished and exploding gradients when using very deep layers. The residual network (ResNet) was proposed to solve these problems by introducing the shortcut connection that does not transform the input from the previous to the following building blocks (called bottleneck design) [14]. In the bottleneck, the 1×1 conv2D is added at the beginning and end of the block to reduce the number of parameters in the network. In our experiments, we used ResNet50 which contains only 50 layers.

DenseNet201. The idea of connections between layers was invented in DenseNet201 and was the same as the ResNet. In the DenseNet architecture [15], instead of shortcut connections, dense connections were used to connect between the current and following layers. Hence, the feature maps from the current layer were concatenated and passed to the next layers during forwarding propagation. At the same time, the backpropagation process sent back the error signal from the final classification layer to the earlier layers to adjust the parameters. In this paper, we used the DenseNet201 architecture that contained convolution, pooling, four dense blocks, three transition layers, and a classification layer.

In our proposed network, the last layer of CNN architectures was the global average pooling (GAP) because the fully-connected layer (FC) and softmax function were removed from the CNN architectures. After applying the GAP layer, the size of the feature map in each CNN was as follows: MobileNetV2 $= 1 \times 1280$, ResNet50 $= 1 \times 2048$, and DenseNet201 $= 1 \times 1920$. Furthermore, we extracted 32 images from the video and then computed sequence patterns from sequence images. Hence, the sequence patterns given to the recurrent neural network (RNN) of each CNN were as follows: MobileNetV2 $= 32 \times 1280$, ResNet50 $= 32 \times 2048$, and DenseNet201 $= 32 \times 1920$.

RNN. The RNN architecture was proposed to learn the sequence pattern of the spatial features extracted by CNN architectures. In the RNN architecture, we added dropout and GAP layers to avoid the overfitting problem and to decrease the size of the feature map

[1]The YOLOv5 pre-trained model was downloaded from the website: https://github.com/ultralytics/yolov5.

before giving them to the softmax layer. We experimented with two RNN architectures: LSTM and GRU. The details are presented as follows.

Long short-term memory. LSTM is a sequential network proposed to learn sequence patterns and address the vanishing gradient problem [16]. It has memory blocks that contain three gates: input, output, and forget gates. The LSTM also has a memory cell that can be recurrently self-connected. The memory blocks are designed to control the information inside the LSTM unit that has an advantage in remembering past information and forgetting unnecessary information from the network.

Gated recurrent unit. The GRU was designed to be similar to the LSTM network, but simpler than it [17]. The GRU network could also solve the vanishing gradient problem that constantly occurs when using the RNN architecture. It contains only two gates: reset (short-term memory) and update gates (long-term memory). The GRU can store sequence patterns from long ago and release irrelevant patterns to the output. Further, the GRU trains faster than the LSTM network. In this paper, we compared the performance of two hidden state sizes (32 and 64 hidden states) of LSTM and GRU.

Recognition block. After extracting the temporal features using RNN architecture, we decided to include two extra layers followed by the softmax layer – 1) the dropout layer to prevent the model from overfitting problems, and 2) a layer to calculate the average output of each feature map using the GAP operation. It also could reduce the dimension of the feature map. The softmax layer is the last layer that computes the final probabilities and recognizes the output.

3. **Dynamic Thai Fingerspelling Dataset.** Dynamic Thai fingerspelling is a short video database with 3,025 videos and 42 classes. The videos of dynamic Thai fingerspelling was recorded using a DSLR camera using a frame rate of 50 frames per second. We recorded the videos with 14 volunteers who can use Thai sign language fluently. The recorded length of the videos was around 1-5 seconds. Examples of the DTF dataset are shown in Figure 2.

FIGURE 2. Examples of dynamic Thai fingerspelling dataset

4. **Experimental Results and Discussion.** The proposed framework was implemented based on the TensorFlow platform running on Intel(R) Core(TM) i7-4790 Processor 3.6GHz, 16GB DDR4 RAM. We trained all deep learning models with these parameters: stochastic gradient descent (SGD) optimizer, the momentum of 0.9, rectified linear unit (ReLU) activation function, and 500 epochs training. Furthermore, we performed the 5-fold cross-validation (5-CV), test accuracy, and testing time as for the evaluation metrics.

4.1. **Experiments with CNN models using different learning rates.** We experimented by extracting the deep sequence features using three CNN models: MobileNetV2, ResNet50, and DenseNet201. Then, all features were given to the LSTM architecture using 32 units to extract temporal features and classify by using the softmax function. In

this experiment, the CNN parameters were adjusted according to the previous section. Further, we mainly study the impact of the learning rate used while training the CNN-LSTM model using 0.001, 0.0001, and adaptive learning rate that reduces the value from 0.01 to 0.0001. We trained the CNN models on the training set and used the validation set for evaluation. The experimental results are shown in Table 1.

Table 1. Experiment with different learning rates

CNN-LSTM model	Test accuracy (%) with different learning rates		
	0.001	0.0001	Adaptive
MobileNetV2-LSTM	**81.98**	80.98	79.01
DenseNet201-LSTM	**82.97**	72.14	77.76
ResNet50-LSTM	**87.93**	82.14	81.81

Table 1 shows the experimental results from three CNN models, when using a learning rate of 0.001 which achieved the highest accuracy on the validation set. As a result, the best CNN model was the ResNet50 which achieved 87.93% accuracy. Note that we subsequently used the learning rate of 0.001 while training the CNN-LSTM models in the following experiments.

4.2. **Experiments with CNNs and RNNs using different RNN sizes.** In this section, we evaluated the performance of the CNNs and RNNs (LSTM and GRU) using two different numbers of units: 32 and 64 units.

As shown in Table 2, it undoubtedly shows that the LSTM, when using both 32 and 64 units, outperformed the GRU. The maximum accuracy of the ResNet50-GRU with 64 units was only 77.68%. In contrast, the ResNet50-LSTM with 64 units had the best performance and achieved 89.16% accuracy on the validation set. In addition, the accuracy obtained above 81% when the CNN model combines with the LSTM. Note that we chose the number of units in the LSTM as 64 units in the following experiments.

Table 2. Comparison between CNN-LSTM and CNN-GRU with different RNN units

CNN models	No. of units in RNNs			
	LSTM (%)		GRU (%)	
	32	64	32	64
MobileNetV2	81.98	**83.96**	71.23	72.23
DenseNet201	82.97	**84.62**	72.14	74.24
ResNet50	87.93	**89.16**	77.52	77.68

4.3. **Experiments with CNN-LSTM.** In this experiment, as shown in Figure 1, we decided to add the extra densely connected layer with 64 units between the GAP layer and softmax activation function. In the densely connected layer, the dot product was applied as a non-linear transformation between the weighted parameters and the output of the GAP layer.

The experimental results comparing CNN-LSTM and CNN-LSTM with one densely connected layer are shown in Table 3. The results showed that the LSTM without a densely connected layer outperformed LSTM with one densely connected layer. As a result, the ResNet50-LSTM achieved the best accuracy on both the 5-CV and test set with 89.15% and 88.42%. We confirmed that the proposed network with a combination between CNN and LSTM prevented the overfitting problem.

The validation loss values of CNN-LSTM architecture are shown in Figure 3. Figure 3(a) shows that the loss values decreased quickly at approximately epoch 50. Also, the

TABLE 3. Comparison between CNN-LSTM and CNN-LSTM with a densely connected layer

CNN models	LSTM			LSTM with a densely connected layer		
	5-CV	Test accuracy (%)	Testing time (seconds per video)	5-CV	Test accuracy (%)	Testing time (seconds per video)
MobileNetV2	83.95 ± 2.525	83.41	6.70 s	82.54 ± 3.006	82.14	6.73 s
DenseNet201	84.64 ± 2.462	82.97	7.04 s	79.45 ± 1.426	76.85	7.07 s
ResNet50	89.15 ± 2.571	**88.42**	9.50 s	86.45 ± 2.146	85.45	9.53 s

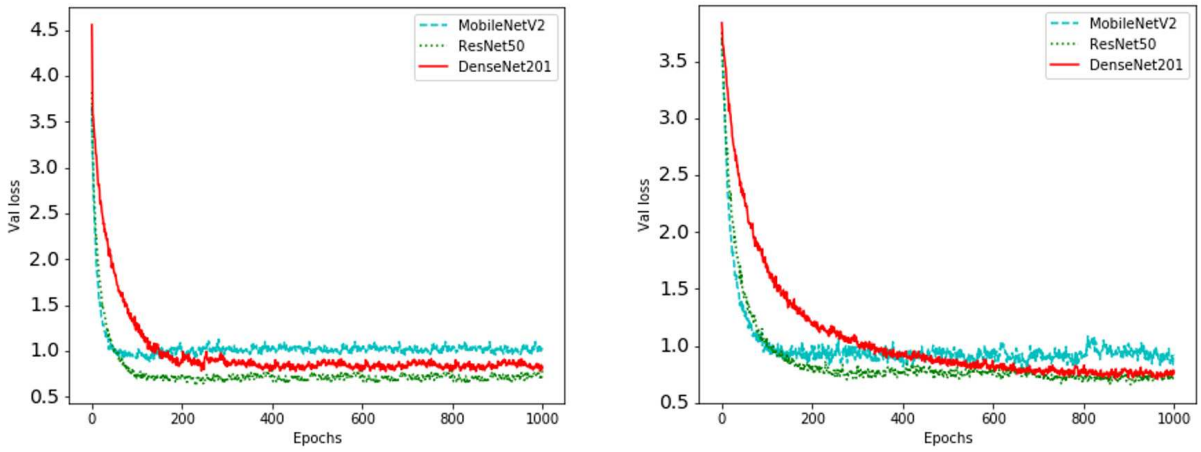

FIGURE 3. Illustration of the validation loss values of CNN-LSTM models. The loss values of CNN-LSTM (a) without and (b) with a densely connected layer.

loss values of Figure 3(a) are smoother than Figure 3(b). Consequently, the loss values of Figure 3(a) indicated that the CNN-LSTM without a densely connected layer was trained with the optimal architectures and parameters.

Additionally, the receiver operating characteristic (ROC) curves for CNN models are shown in Figure 4. The ResNet50 had the greatest AUC (0.9891) in dynamic fingerspelling recognition.

4.4. **Accuracy of fusion CNNs-LSTM architecture.** In other papers, the fusion architecture between two CNNs and LSTM outperformed the single CNN and LSTM. We demonstrate the impact of the fusion CNN-LSTM architectures, as shown in Table 4.

Table 4 shows the results of experiments with combinations of CNNs: MobileNetV2+ DenseNet201, MobileNetV2+ResNet50, and ResNet50+DenseNet201, using the concatenation operation before sending the sequence patterns to the LSTM network. The combination of MobileNetV2 and ResNet50 outperformed other combination CNNs. The fusion CNNs (MobileNetV2+ResNet50)-LSTM achieved an accuracy of 89.42% and 88.62% on the 5-CV and test sets, respectively.

Consequently, the Fusion CNN-LSTM showed slightly higher accuracy compared with the single CNN-LSTM, but the result was not significant at $p < 0.05$. However, the Fusion CNN-LSTM architecture spent much more time on recognition than using only a single CNN-LSTM architecture.

5. **Conclusion.** We presented an automated end-to-end dynamic fingerspelling recognition system that contains frame selection, human detection, and fingerspelling recognition. We focused on the deep learning approaches: the YOLOv5 algorithm for human

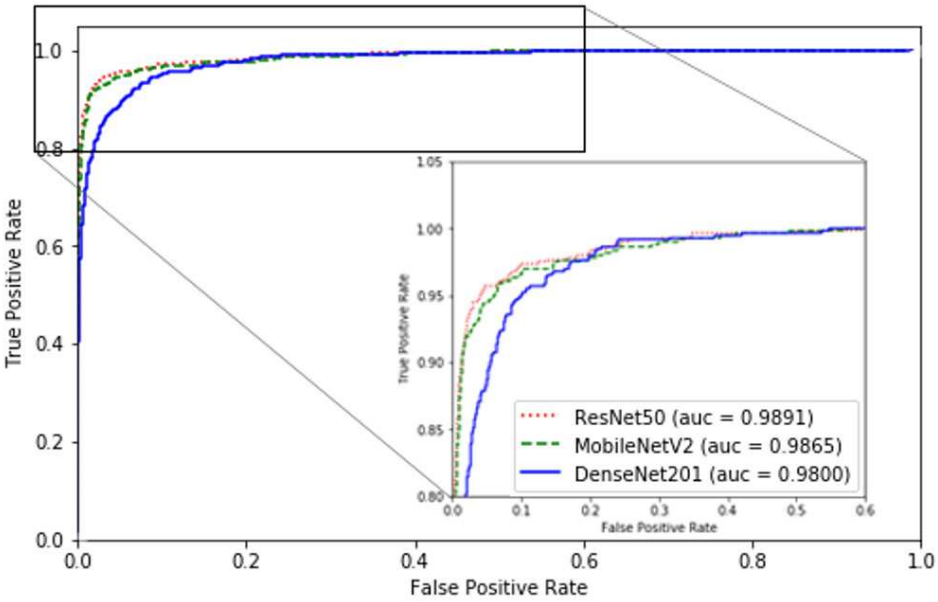

FIGURE 4. Illustration of the ROC curves for dynamic fingerspelling recognition using different CNNs

TABLE 4. Evaluation of fusion CNNs-LSTM

Fusion CNNs	LSTM		
	5-CV	Test accuracy (%)	Testing time (seconds per video)
MobileNetV2+DenseNet201	85.29 ± 3.395	84.29	13.74 s
MobileNetV2+ResNet50	89.42 ± 3.670	**88.62**	16.20 s
ResNet50+DenseNet201	88.36 ± 2.917	87.76	16.54 s

detection and CNN-LSTM architecture for recognizing the dynamic fingerspelling from the video. First, 32 video frames were selected using the uniform distribution method and sent to extract the sequence patterns using CNN architecture. Second, the sequence patterns were given to the LSTM architecture, followed by three layers: dropout, GAP, and softmax activation function. Moreover, we mainly evaluated the recognition performance of CNN-LSTM architectures. The results obtained from the ResNet50-LSTM architecture achieved the highest accuracy on the validation set. We experimented with the 5-CV and test set to confirm that the ResNet50-LSTM architecture outperformed MoblieNetV2-LSTM and DenseNet201-LSTM architectures. Additionally, we compared the MobileNetV2-LSTM with the fusion CNN-LSTM architecture. The results showed that the fusion CNN-LSTM slightly outperformed MobileNetV2-LSTM. The fusion CNN-LSTM spends more computational time on recognition. As a result, we are not recommending the use of fusion CNN-LSTM architecture for real-time video fingerspelling recognition.

In the future work, we plan to apply the proposed method to recognizing multiple words and sentences of the Thai sign language. The lightweight 3D-CNN might be included in future experiments to enhance accuracy and computation time.

Acknowledgments. This research project was financially supported by Mahasarakham University. We would like to thank all participants from Rajabhat Mahasarakham University for their interest in contributing to this research.

REFERENCES

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

[2] M. R. Islam, U. K. Mitu, R. A. Bhuiyan and J. Shin, Hand gesture feature extraction using deep convolutional neural network for recognizing American sign language, *Proc. of the 4th International Conference on Frontiers of Signal Processing*, pp.115-119, 2018.

[3] N. H. Phong and B. Ribeiro, Advanced capsule networks via context awareness, *Proc. of International Conference on Artificial Neural Networks*, pp.166-177, 2019.

[4] A. Mujahid et al., Real-time hand gesture recognition based on deep learning YOLOv3 model, *Applied Sciences*, vol.11, no.9, DOI: 10.3390/app11094164, 2021.

[5] X. Yu, Hand gesture recognition based on Faster-RCNN deep learning, *J. Comput.*, vol.14, no.2, pp.101-110, 2019.

[6] C. Chansri and J. Srinonchat, Reliability and accuracy of Thai sign language recognition with Kinect sensor, *Proc. of the 13th Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, pp.1-4, 2016.

[7] T. Pariwat and P. Seresangtakul, Thai finger-spelling sign language recognition using global and local features with SVM, *Proc. of the 9th International Conference on Knowledge and Smart Technology*, pp.116-120, 2017.

[8] 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.

[9] B. Sugandi, S. E. Octaviani and N. F. Pebrianto, Visual tracking-based hand gesture recognition using backpropagation neural network, *International Journal of Innovative Computing, Information and Control*, vol.16, no.1, pp.301-313, 2020.

[10] T. Pariwat and P. Seresangtakul, Multi-stroke Thai finger-spelling sign language recognition system with deep learning, *Symmetry*, vol.13, no.2, DOI: 10.3390/sym13020262, 2021.

[11] P. Nakjai and T. Katanyukul, Automatic Thai finger spelling transcription, *Walailak Journal of Science and Technology*, vol.18, no.13, DOI: 10.48048/wjst.2021.11233, 2021.

[12] G. Jocher, A. Stoken, J. Borovec et al., Ultralytics/YOLOv5: V4.0 - nn.SiLU() activations, Weights & Biases logging PyTorch Hub integration, *Zenodo*, DOI: 10.5281/zenodo.4418161, 2021.

[13] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L.-C. Chen, MobileNetV2: Inverted residuals and linear bottlenecks, *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp.4510-4520, 2018.

[14] K. He, X. Zhang, S. Ren and J. Sun, Deep residual learning for image recognition, *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp.770-778, 2016.

[15] G. Huang, Z. Liu, L. Van Der Maaten and K. Q. Weinberger, Densely connected convolutional networks, *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp.2261-2269, 2017.

[16] M. M. Soliman, M. H. Kamal, M. A. El-Massih Nashed, Y. M. Mostafa, B. S. Chawky and D. Khattab, Violence recognition from videos using deep learning techniques, *Proc. of the 9th International Conference on Intelligent Computing and Information Systems*, pp.80-85, 2019.

[17] H. Chen, C. Hu, F. Lee, W. Yao, L. Chen and Q. Chen, A supervised video Hashing method based on a deep 3D convolutional neural network for large-scale video retrieval, *Sensors*, vol.21, no.9, DOI: 10.3390/s21093094, 2021.