Image Recognition based on a Sequenced Edge Grid Image Technique

Supathep Satiman
Faculty of Information Technology and Digital Innovation
King's Mongkut University of Technology North Bangkok
Bangkok, Thailand
supathep.s@kmutnb.ac.th

Phayung Meesad
Faculty of Information Technology and Digital Innovation
King's Mongkut University of Technology North Bangkok
Bangkok, Thailand
phayung.s@kmutnb.ac.th

***Abstract*—Currently there are many techniques and methods continuously proposed by researchers for Sign language recognition systems based-on machine learning. For data preprocessing for sign language, majority of researchers use single image of hands like static gesture images. Using only static hand images may not be efficient for real-world applications. In this paper, we propose an innovative technique for video processing called Sequenced Edge Grid Images (SEGI) for Sign Language recognition. The proposed SEGI is composed of images that represent the movement of hands within a single image, which can be applied to recognize a word or a sentence. To proof the concept, we have done several experiments with Thai sign language data collected from internet. SEGI was with existing techniques. Data are the Thai sign language learning video clips that are vocabularies to use in daily life. The proposed technique was implemented with convolutional neural network (CNN). For normal CNN, the experiments show that the best result based on SEGI with CNN approached up 99.95% recognition.**

Keywords—sequenced edge grid image, sign language recognition, video preprocessing, convolution neural network

I. INTRODUCTION

In the field of machine learning for sign language recognition systems, there are many techniques and methods that are being proposed by researchers around the world. However, many datasets that were experimented are features extracted such as detecting and cropping only the hand part, as well as storing hand motion tracked from sensor devices. As a result, machine learning for sign language recognition seems to have a high accuracy performance. However, there may be limitations in real-world applications. From the study, facial expressions and body gestures are more important to sign language communication in daily life. Moreover, some gestures of sign language have similar hand movements but different hand positions may change the meaning of communication.

Convolutional Neural Networks (CNN) have proven to be a very high-performance machine learning tool for computer vision tasks. Because in CNN architecture are down-sampling layers introduced to reduce the resolution of the feature map as well as the sensitivity of the output to shifts and distortions [1]. In field of sign language recognition, it is found that CNN is very powerful for sign language recognition.

For data preprocessing for sign language, researchers have proposed single static gesture images that may not be efficient for real-world applications. An innovative technique for digital image data preparation called Sequenced Edge Grid Images (SEGI) for Sign Language recognition is proposed in this paper.

The rest of this paper is organized as follows. Section II describes the literature reviews of related works. The proposed method is discussed in Section III. Section IV contains the results of our tests. Section V concludes the paper with the discussions on our method.

II. RELATED WORK

A. Data preprocessing / Sequence images

For many of the research in sign language recognition, there are several methods for generating and preprocessing the dataset. The features extraction techniques include trigger detection system [2], principal component analysis for feature reduction [3], hand segmentation and conversion of image to global and local feature extraction [4], sensor-based device with deciphers sign language of hand gesture [5]. In addition, in previous research the datasets were cropped to select only the hand part and cut off the other part of the image. The sign language datasets are generally available in the static gesture image form. Using static datasets may have limitations in their use in real-world applications. This is because a sign language is a continuous hand movement. Facial expressions and body gestures are more important to sign language communication in daily life. Moreover, some gestures of sign language have similar hand movements but different hand positions that may lead to different meaning of communication.

There have been some works proposed to solve the problem by creating continuous frames from video [6]. This approach allows dynamic gesture images that are continuous images from sign language gestures. In addition, there are also some research focused on optimization of the performance of dynamic gestures image by summarizing the gesture from sequences of frames of a video, capturing the key points of the resulting frames [7].

B. Edge detection

Image edge detection is an important basis for image recognition extraction. Edge detection can reduce an amount of information from the image to be processed. The procedure of edge detection is to compute gradient magnitudes and edge directions in an image, then to compute the edge strength on

the gradient magnitudes of brightness within the image to detect and extract the edges as output [8]. The Sobel operator is a widely used filter to compute gradients. The Sobel operator uses a pair of convolution matrices/masks as shown in Fig. 1, one for estimating the horizontal gradient and the other for the vertical one. For example, the horizontal gradient mask is constructed by multiplying a horizontal averaging vector with a horizontal differential vector [9].

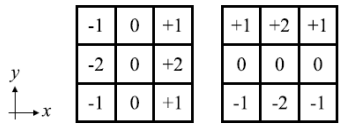


Fig. 1: Sobel masks of 3 × 3 dimensions: (a) horizontal, (b) vertical.

If we let a_{ij} be a brightness value on a cell (i,j) of the source image, and dx_{ij} and dy_{ij} be approximated horizontal and vertical gradients on the cell (i,j), respectively, they are computed by using the Sobel operator as follows:

$$dx_{ij} \equiv \begin{bmatrix} -10+1 \\ -20+2 \\ -10+1 \end{bmatrix} \cdot \begin{bmatrix} a_i & a_{ij+1} & a_{i+1j+1} \\ a_{i-1j} & a_{ij} & a_{i+1j} \\ a_{i-1j-1} & a_{ij-1} & a_{i+1j-1} \end{bmatrix},$$

$$dy_{ij} \equiv \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \cdot \begin{bmatrix} a_{i-1j+1} & a_{ij+1} & a_{i+1j+1} \\ a_{i-1j} & a_{ij} & a_{i+1j} \\ a_{i-1j-1} & a_{ij-1} & a_{i+1j-1} \end{bmatrix} \quad (1)$$

where $(\cdot)$ means inner product calculation. After calculating the approximated gradients, we can calculate a gradient magnitude d_{ij} and its direction θ_{ij} (hereinafter, this is called gradient direction) on the cell (i,j), by the following formula:

$$d_{ij} \equiv \sqrt{dx_{ij}^2 + dy_{ij}^2} \quad (2)$$

$$\theta_{ij} \equiv \tan^{-1}(dy_{ij} + dx_{ij}) \quad (3)$$

Another algorithm gained popularity is canny edge detector, which is an edge detection operator using a multi-stage method to detect a wide range of edges in images. It was developed by John F. Canny in 1986. The Canny Edge Detection Algorithm runs [10] in 5 steps as follow.

Smoothing: Blurring of the image to remove noise.

Finding gradients: The edges should be marked where the gradients of the image has large magnitudes.

Non-maximum suppression: Only local maxima should be marked as edges.

Double thresholding: Potential edges are determined by thresholding.

Edge tracking by hysteresis: Final edges are determined by suppressing all edges that are not connected to a very certain (strong) edge.

C. CNN

The CNN mainly consists of two parts: convolutional layers (CONV), and fully-connected layers (FULC) that follow (Fig. 2). CONV first extract and combine local features from the input image, and these features are then combined to output feature maps that represent a spatial arrangement of activations. Each unit in a CONV layer receives inputs from a set of units located in a small neighborhood in the previous layer. With a spatial arrangement, neurons can extract primitive visual features such as oriented edges, endpoints, and corners. In fact, CONV can support flexible image sizes and can generate feature maps of any sizes. On the other hand, the FULC needs to have fixed-size/length input by their definition. The inputs are processed in FULC and are converted into 1D feature vector (flatten). Then 1D features are represented as each neuron in the fully connected layer and multiplied with the weight of the neuron to produce the output [1].

In 2D CNN is performed at the convolutional layers to extract features from local neighborhood on feature maps in the previous layer. Formally, the value of a unit at position (x,y) in the ith feature map in the ith layer, denoted as v_{ij}^{xy}, is given by

$$v_{ij}^{xy} = tanh\left(b_{ij} + \sum_m \sum_{p=0}^{P_i-1} \sum_{q=0}^{Qi-1} w_{i\,jm}^{pq}\, v_{(i-1)m}^{(x+p)(y+q)}\right) \quad (4)$$

where $tanh(\cdot)$ is the hyperbolic tangent function, b_{ij} is the bias for this feature map, m indexes over the set of feature maps in the $(i-1)$th layer connected to the current feature map, w_{ijk}^{pq} is the value at the position (p,q) of the kernel connected to the kth feature map, and P_i and Q_i are the height and width of the kernel, respectively [11].

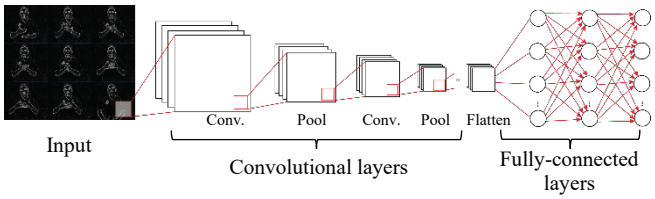


Fig. 2: CNN structure

III. The Proposed SEGI Dataset

In this paper, we propose an innovative technique for digital image data preparation called Sequenced Edge Grid Images (SEGI) for Sign Language recognition. The proposed approach is for video data-preprocessing, frame extraction from video files, data-preprocessing and reconstruct new dataset form. SEGI is aimed to solve static gesture image problems. Sequence images are constructed to store the details of gesture movements in sign language. Optimal SEGI size and input size are then used to train CNN to classify Thai sign language recognition.

The video preprocessing to SEGI shown in Fig. 3 is a semi-automated method with 4 steps as follow:

Image Frames: Image frames are captured from a video clip and the number of frames is set as desired.

Cropped Image: This step is to get an optimal number of image frames. First step is to cut off empty areas and borders of each image frame.

Edge Detection: This step converts the image frames (RGB mode) to gray scale to perform edge detection. This process will reduce information of images because in this process is to remove color and texture (high-frequency data).

Concatenate sub-image: This step constructs SEGI by combining sub-image (edge image converted) and order from a top-left to top-right.

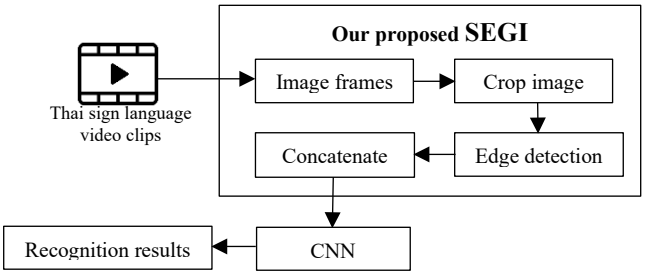


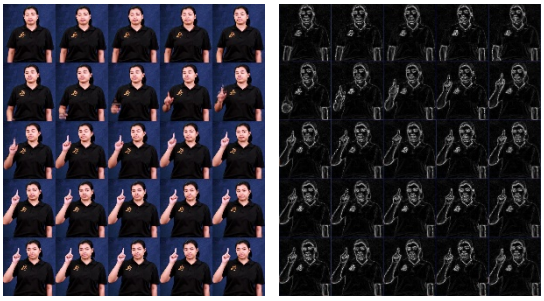
Fig. 3: Block diagram representation of the proposed SEGI dataset

IV. Performance Evaluation

The following experimental details are constructed to evaluate the performance of the proposed SEGI dataset for sign language recognition.

A. SEGI datset vs Sequenced grid images

First step we constructed SEGI size 5×5 (rows × column) from a video clip. Fig. 4 shows sequenced grid images in RGB and SEGI mode.



(a) (b)

Fig. 4: (a) Sequence image size 5×5 (RGB mode) and (b) SEGI size 5×5

In the preliminary experiments, we performed experiments on Thai sign language for count number one to nine. We measured the performance of a CNN image recognition based on SEGI and SGI (sequenced grid image). We used input images with size 64×64 pixels, to generate SEGI and SGI size 5×5. There were 126 resulted images with 9 classes. We split dataset into training set and validation set: total 99 SEGI (11 SEGI/class) and validation set to 27 SEGI.

TABLE I. PERFORMANCE OF THAI SIGN LANGUAGE RECOGINTION COMPARED TO SEQUENCE IMAGE AND SEGI

Epochs	Image dataset	Time (sec.)	Train acc.	Train loss	Valid acc.	Valid Loss
30	SGI	9896.59	0.9947	0.0170	0.5772	4.0453
	SEGI	2340.73	0.9936	0.0244	0.7269	1.7833
60	SGI	15654.05	0.9948	0.0354	0.4537	5.6329
	SEGI	7975.11	0.9939	0.0505	0.7289	3.0534

The results showed that the SEGI can increase the performance for Thai sign language recognition when compared to SGI. In addition, it was found that SEGI also reduces the cost of processing time with approximately 49% for 60 Epochs of training.

B. Optimal SEGI for Thai sign language recognition

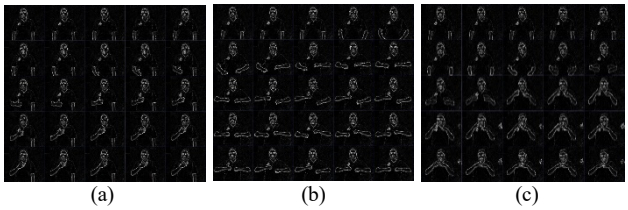
We focused on seeking optimal size of SEGI. Observing SEGI dataset, it was found that the sub-images was too redundant (as shown in Fig. 4). This is because each sign language vocabulary has a simple gesture and short movements. Initially, we used video clips taking 25 frames/second to transform in to 5×5 sub-images in one SEGI. Next, we experimented on rescaling the SEGI size from 5×5 to 3×3 and 4×4. The performance of each scaled image seize with different complexity levels for Thai sign language vocabularies was tested. We split the dataset to the length of the vocabulary as follows:

Dataset 1 was one-syllable SEGI dataset consisted of: 30 class, total equal to 750 SEGI, split to training set equal to 450 SEGI, validation set equal to 150 SEGI and test set equal to 150 SEGI.

Dataset 2 was two-syllable SEGI dataset consisted of: 33 class, total equal to 825 SEGI, split to training set equal to 495 SEGI, validation set equal to 165 SEGI and test set equal to 165 SEGI.

Dataset 3 was three and more syllable SEGI dataset consisted of: 10 class, total equal to 250 SEGI, split to training set equal to 150 SEGI, validation set equal to 50 SEGI and test set equal to 50 SEGI.

We also tested three input sizes: 32×32, 64×64, and 128×128 pixels. We used the images as inputs to CNN adding dropout to learn SEGI dataset with 30 epochs. In order to find the optimal input size and SEGI size to obtain best recognition performance. The test results are shown in Tables 2 – 4.



(a) (b) (c)

Fig. 4: Example SEGI size 5×5 according to the length of Thai sign language vocabulary (a) SEGI of one-syllable, (b) SEGI of two-syllable and (c) SEGI of three and more syllable

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
rescaling_1 (Rescaling)       (None, 32, 32, 3)         0
_________________________________________________________________
conv2d (Conv2D)              (None, 32, 32, 32)        896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 16, 16, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 16, 16, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 8, 8, 64)          0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 8, 8, 128)         73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 4, 4, 128)         0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 4, 4, 64)          73792
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 2, 2, 64)          0
_________________________________________________________________
dropout (Dropout)            (None, 2, 2, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 256)               0
_________________________________________________________________
dense (Dense)                (None, 128)               32896
_________________________________________________________________
dense_1 (Dense)              (None, 96)                12384
=================================================================
Total params: 212,320
Trainable params: 212,320
Non-trainable params: 0
```

Fig. 5: 2D CNN structure

Tables 2 - 4 show experimental results. It can be observed that the smallest size of SEGI, 3×3, provides the best

performance compared to bigger sizes like 4×4 and 5×5. Although the input size is scaled to a minimum of 32×32 pixels. When the input size is scaled up, it increases the performance of Thai sign language recognition as shown in Tables 2, 3, and 4. Using input size 128×128 images, the SEGI size 4×4 has higher performance. That is because the SEGI is not over-compressed when the input is large size. As a result, SEGI size 4×4 is higher performance than SEGI size 3×3. Using larger images will cost higher computing time as well; a high-performance computing device is needed to process. However, taking longer time to process does not mean the performance is better. It is recommended to use input 64×64 pixels and SEGI size 3×3 as the default size.

TABLE II. THE EGI RECOGNITION PERFORMANCE OF ONE-SYLLABLE VOCABULARY

Input size (px.)	SEGI size (row × col)	Performance evaluation					
		Time (sec.)	*Train acc.*	*Train loss*	*Val acc.*	*Val loss*	*Test acc.*
32×32	3×3	11.57	0.98	0.06	0.65	1.36	75.50
	4×4	16.60	0.95	0.17	0.38	2.65	72.02
	5×5	23.73	0.97	0.10	0.19	3.74	64.90
64×64	3×3	12.14	0.98	0.05	0.94	0.20	92.09
	4×4	17.3	0.98	0.07	0.85	0.81	83.81
	5×5	23.91	0.97	0.08	0.64	1.41	77.25
128×128	3×3	36.99	0.998	0.005	0.996	0.008	99.50
	4×4	44.57	0.996	0.007	0.991	0.025	99.24
	5×5	49.55	0.996	0.012	0.953	0.120	96.12

TABLE III. THE EGI RECOGNITION PERFORMANCE OF TWO-SYLLABLE VOCABULARY

Input size (px.)	SEGI size (row × col)	Performance evaluation					
		Time (sec.)	*Train acc.*	*Train loss*	*Val acc.*	*Val loss*	*Test acc.*
32×32	3×3	12.60	0.99	0.05	0.677	1.109	79.11
	4×4	18.50	0.95	0.17	0.335	2.993	68.25
	5×5	25.87	0.99	0.05	0.245	4.208	63.99
64×64	3×3	13.6	0.996	0.013	0.939	0.168	94.08
	4×4	19.32	0.991	0.037	0.760	0.820	86.45
	5×5	26.79	0.994	0.027	0.626	1.369	77.20
128×128	3×3	21.30	1.000	0.001	0.996	0.010	99.23
	4×4	27.51	0.999	0.002	0.990	0.031	98.70
	5×5	35.30	0.993	0.025	0.905	0.339	94.11

TABLE IV. THE EGI RECOGNITION PERFORMANCE OF THREE AND MORE SYLLABLE VOCABULARY

Input size (px.)	EGI size (row × col)	Performance evaluation					
		Time (sec.)	*Train acc.*	*Train loss*	*Val acc.*	*Val loss*	*Test acc.*
32×32	3×3	5.66	0.993	0.038	0.847	0.374	87.36
	4×4	7.55	0.978	0.087	0.700	0.804	81.34
	5×5	9.74	0.976	0.113	0.427	2.034	70.02
64×64	3×3	6.11	0.998	0.006	1.000	0.016	99.01
	4×4	8.06	0.989	0.033	0.940	0.181	95.29
	5×5	10.30	0.996	0.021	0.847	0.430	86.43
128×128	3×3	15.59	0.993	0.015	1.000	0.001	99.74
	4×4	17.61	1.000	0.003	1.000	0.001	99.95
	5×5	19.47	1.000	0.003	1.000	0.011	97.71

V. CONCLUTION

In this work, we propose the new technique of video processing called Sequenced Edge Grid Images (SEGI) for Sign Language recognition. SEGI dataset is constructed as a sequence image with edge detection. SEGI is used with CNN that learns to recognize Sing Languages. Testing with Thai sing language, the recognition performance of CNN based on SEGI has higher accuracy than that SGI based CNN has. In addition, with the reduced size SEGI, we found that using SEGI size 3×3 images yield higher accuracy than using SEGI sizes 4×4 and 5×5 with input image size 32×32 pixels and 64×64 pixels. Moreover, scaling up input size to 128×128 pixels, it is found that SEGI size 4×4 yields higher performance.

In the future work, we will use the SEGI size 3×3 with an increasing number of sign language vocabularies (for example computer vocabularies, medical vocabularies, etc.) and increase the number of sign language interpreters. In addition, we will use transfer learning method, which can incrementally learn new Thai sign language dataset. The transfer learning is especially useful when the data is not enough, or training time and computing resources are restricted.

REFERENCES

[1] J. Fan, W. Xu, Y. Wu, and Y. Gong, "Human Tracking Using Convolutional Neural Networks," *IEEE Trans. Neural Netw.*, vol. 21, no. 10, pp. 1610–1623, Oct. 2010.

[2] D. Chakraborty, D. Garg, A. Ghosh, and J. H. Chan, "Trigger Detection System for American Sign Language using Deep Convolutional Neural Networks," in *Proceedings of the 10th International Conference on Advances in Information Technology - IAIT 2018*, Bangkok, Thailand, 2018, pp. 1–6.

[3] J. P. Sahoo, S. Ari, and S. K. Patra, "Hand Gesture Recognition Using PCA Based Deep CNN Reduced Features and SVM Classifier," in *2019 IEEE International Symposium on Smart Electronic Systems (iSES) (Formerly iNiS)*, Dec. 2019, pp. 221–224.

[4] S. Saengsri, V. Niennattrakul, and C. A. Ratanamahatana, "TFRS: Thai finger-spelling sign language recognition system," in *2012 Second International Conference on Digital Information and Communication Technology and it's Applications (DICTAP)*, May 2012, pp. 457–462.

[5] A. B. Jani, N. A. Kotak, and A. K. Roy, "Sensor Based Hand Gesture Recognition System for English Alphabets Used in Sign Language of Deaf-Mute People," in *2018 IEEE SENSORS*, Oct. 2018, pp. 1–4.

[6] K. Bantupalli and Y. Xie, "American Sign Language Recognition using Deep Learning and Computer Vision," in *2018 IEEE International Conference on Big Data (Big Data)*, Dec. 2018, pp. 4896–4899.

[7] A. Neyra-Gutiérrez and P. Shiguihara-Juárez, "Feature Extraction with Video Summarization of Dynamic Gestures for Peruvian Sign Language Recognition," in *2020 IEEE XXVII International Conference on Electronics, Electrical Engineering and Computing (INTERCON)*, Sep. 2020, pp. 1–4.

[8] L. Sathiya and V. Palanisamy, "Minor finger knuckle print image edge detection using second order derivatives," in *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, Apr. 2019, pp. 1227–1231.

[9] T. Fujimoto, T. Kawasaki, and K. Kitamura, "Canny-Edge-Detection/Rankine-Hugoniot-conditions unified shock sensor for inviscid and viscous flows | Elsevier Enhanced Reader," vol. 396, pp. 264–279.

[10] P. Prathusha, S. Jyothi, and D. M. Mamatha, "Enhanced Image Edge Detection Methods for Crab Species Identification," in *2018 International Conference on Soft-computing and Network Security (ICSNS)*, Feb. 2018, pp. 1–7.

[11] S. Ji, W. Xu, M. Yang, and K. Yu, "3D Convolutional Neural Networks for Human Action Recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, Art. no. 1, Jan. 2013.