International Journal of Artificial Intelligence,
ISSN 0974-0635; Int. J. Artif. Intell.
2024, Volume 22, Issue Number 2

Compact Thai Sign Language Translation by Deep Learning

Kietikul Jearanaitanakij[1], Piyathida Choojan[2] and Piyada Thongtem[3]

Department of Computer Engineering, School of Engineering,
King Mongkut's Institute of Technology Ladkrabang,
Bangkok, 10520 Thailand
Email : [1]kietikul.je@kmitl.ac.th, [2]faipiyathida882@gmail.com, [3]piyada.tht@gmail.com

ABSTRACT

Sign language translation is a challenging problem in natural language processing. Its principle involves machine translation from sign language images to spoken language text. Designing a good translation is not a trivial task since there are a large number of both input image pixels and output classes. We propose the deep learning model to translate static gestures of Thai sign language (TSL) to the corresponding Thai spoken words. The main objective is to design a compact model that delivers high performance so that it can be implemented on mobile devices. Several mobile convolutional neural networks (CNN) are investigated to find the best backbone architecture. We also attach additional layers to the selected CNN architecture to fine-tune its performance. The experiments on the dataset collected from twenty-four volunteers indicate excellent results; in terms of precision, recall, and f1-score, of the proposed model. The comparisons with the state-of-the-art models and the feature visualizations from convolution layers endorse its effectiveness.

Keywords: Deep Learning, Convolutional Neural Network, MobileNet Model, Language Translation, Thai Sign Language.

Mathematics Subject Classification: 68T07

Journal of Economic Literature (JEL) Classification: C45

1. INTRODUCTION

Sign language is a visual communication that uses arm/hand gestures to convey meaning among deaf and hearing-impaired people. They are not universal, although some of them are similar. Deaf people who understand one sign language may not readily understand other sign languages without prior knowledge or effort. Sign language translation is a challenging problem because it is an attempt to solve a problem by using multiple research fields, e.g., image processing, feature engineering, machine learning, natural language processing, etc. In 2018, there are about 375,680 deaf individuals in Thailand as reported by the Ministry of social development and human security. These people possess abilities to work in the same environment as other people and deserve a right to communication. Therefore, providing a tool allowing them to convey sign language to the hearing community is in need. There are various efforts to translate sign languages ranging from using sensory devices to deep

learning architecture. We will mention them in chronological order. Saengsri et al. (2012) propose a TSL recognition system by applying data gloves and motion tracking sensors. They improve the accuracy to recognize finger-spelling Thai alphabets by utilizing data segmentation and a neural network model. Rahaman et al. (2014) present the Bengali Sign Language prediction by applying the K-Nearest Neighbors classifier to the binary images of hand signs. Their experimental results indicate high recognition for both vowels and consonants. Adhan and Pintavirooj (2016) attach 6 sphere markers to a black glove and use a two-layer neural network to recognize 42 alphabets in TSL. Although their method achieves good performance, recognition of some alphabets needed to be improved since accuracies significantly drop. Pariwat and Seresangtakul (2017) recognize 15 Thai finger spelling alphabets by using SVM. They extract local and global features from the input finger image and feed them to the SVM classifier. The experimental results indicate that the RBF kernel function with the mixture of local and global features delivered the best accuracy among other functions. Jani et al. (2018) invent a hand glove by using Arduino and Raspberry Pi boards. Inputs from five flex sensors, an accelerometer, and a gyroscope are fed into a simple module to predict the English characters of American Sign Language. Rao et al. (2018) recognize selfie images of Indian sign language by the convolutional neural network (CNN). Four convolutional layers with different kernel sizes of 16×16, 9×9, 5×5, and 5×5 work best for their dataset. Their CNN model achieves a fairly high recognition rate compared to other off-the-shelf classifiers. Lim et al. (2019) combine the hand tracking method with pre-trained CNN to characterize the hand models. They use hand energy images to serve as a compact hand representation. According to the experimental results on the American sign language and sign language lexicon video corpus datasets, their method achieves good recognition in terms of accuracy. Sripairojthikoon and Harnsomburana (2019) propose TSL prediction by using a 3D CNN to extract both temporal and spatial features from Thai sign language vocabulary. Its input layer contains a stack of continuous video frames with a size of 64x64x15 from Microsoft Kinect. According to their experimental comparison of different numbers of frame depths, the suitable value that produces the best accuracy is 15. Xie and Ma (2019) utilize the Resnet50 CNN model to develop American sign language recognition. High-level features are efficiently captured by their model. Signal strengthen components are applied during the training period to improve the model's generalization. The experimental results on static sign words show superior accuracy over other pre-trained models. Wadhawan and Kumar (2020) apply a custom CNN architecture to recognize 100 static manual words of Indian sign languages. Their CNN model composes of 5 convolution layers, 2 dense layers, and 1 softmax layer. However, the number of trainable parameters in the CNN model is too large and inappropriate to apply to a mobile device with small computing power. Adithya and Rajesh (2020) propose the CNN architecture composed of 3 convolution layers and one of each softmax and dense layer to recognize hand postures. Their experimental results on the NUS hand posture dataset and the American fingerspelling dataset indicate high recognition accuracy. Wangchuk et al. (2021) present the CNN model which composes of 6 convolution layers to classify 10 static digits of Bhutanese sign language. Batch normalization, dropout, and early stopping are used to avoid overfitting problems. The experimental result indicates that their method and LeNet5 pre-trained model perform better than other non-CNN algorithms. Tornay et al. (2020) invent a multilingual sign language by using Kullback-Leibler divergence HMM. They

demonstrate this through an inspection of Turkish, Swiss, and German sign languages. According to their experimental result, a multilingual sign language recognition system can be efficiently implemented by using pooling resources from multiple sign languages. Phothiwetchakun and Rakthanmanon (2021) present a two-level approach to Thai fingerspelling recognition. They represent hand images with merely 21 points per hand by using hand landmarks. Similar hand gestures are discriminated by using key point clustering as a two-level classifier. Evaluation of one-stroke Thai fingerspelling datasets shows good performance on unseen data. Chaikaew et al. (2021) compare the accuracy among GRU, Bi-LSTM, and LSTM to recognize TSL. Their experimental results show that LSTM has the highest accuracy on the test data.

In this paper, we approach another aspect of TSL. Our goal is to invent a compact TSL translator which can be easily embedded in the mobile application. We propose a compact CNN architecture that still produces high performance of recognition in terms of accuracy, precision, recall, and f1-score. The experimental results on the 100-class dataset collected from 24 volunteers show significantly higher performance compared to state-of-the-art approaches and other pre-trained CNN models. Our TSL dataset composes of 12000 images equally distribute over 100 Thai words used in daily life. The main contribution of this research is a quest for a compact-size and effective CNN architecture that can be applied to a mobile application for predicting static gestures of TSL.
We organize this paper into 6 sections. Section 2 explains central knowledge and existing pre-trained CNN models related to the proposed method. Sections 3 and 4 describe the Thai sign language dataset and the proposed model. Section 5 presents the experimental results and comparisons with other techniques. Finally, a discussion and conclusion, including a possible improvement of the proposed method are provided in section 6.

2. RELATED WORKS

We provide fundamental knowledge and briefs of all works related to our research in this section.

2.1. Thai sign language

The Ministry of Thai Education has designated TSL as the national sign language for Thai deaf people since August 1999. According to (Boonya, 2008), about 52 percent of words in TSL is adopted from American sign language when the first deaf education program as established in 1951 at a public school in Bangkok. There are minor variations of TSL depending upon different regions, gender, and age although they share most words in common. In spite of having a lot of Thai sign words, most deaf people feel difficult to express technical terms or new-emerging phrases. They also expect a sign language comprehension from normal hearing people, at least common words in daily life.

2.2. Convolutional neural network (CNN)

CNN is a well-known and widely used model in deep learning. It was inspired by brain's visual cortex that responds to stimuli in a restricted region of the receptive field. CNN is a great leap forward because

it can vastly reduce a huge number of parameters and the vanishing gradient which are two major problems in training a multilayer neural network (Bengio, 2016). Since our objective is to design a compact CNN for TSL translation application on mobile device, variations of MobileNet architecture are explained in the following subsections.

2.3. MobileNet

Howard et al. (2017) introduced the first version of MobileNet for mobile and embedded vision applications. Its lightweight deep architecture is based on depth-wise separable convolution. This kind of layer decomposes a normal convolution into point-wise and depth-wise convolutions. After a filter is applied to each input channel, a 1×1 convolution in the point-wise layer joins the depth-wise layer's outputs. This factorization can significantly reduce the cost of computation and the model size. The balance between latency and accuracy of MobileNet can be adjusted by its two parameters, i.e., width and resolution multipliers. MobileNet was investigated by applying it to different applications, e.g., geometric localization, fine-grained recognition, person recognition, and face embedding. MobileNet demonstrates promising accuracy on those tasks and drastically reduces the number of parameters compared to other existing pre-trained CNN models.

2.4. MobileNetV2

Sandler et al. (2019) launched the second version of MobileNet called MobileNetV2. The successor improves the performance of MobileNet by largely reducing unnecessary computations and memory requirements. Moreover, MobileNetV2 still retains the same recognition accuracy as its ancestor. The secret behind the success is the inverted residual module. It contains a special residual module that converts a low-dimensional input into a high-dimension output. After a depth-wise convolution, the resulting signals are linearly convolved back to low-dimensional features. In general, this module provides the ability to separate the network expressiveness from its capacity making it suitable for mobile applications.

2.5. MobileNetV3

Howard et al. (2019) proposed MobileNetV3 to optimize the accuracy-latency trade-off on mobile devices. NetAdapt algorithm is applied to search the optimal number of filters per layer. They improve the network architecture by redesigning the layers which have expensive computation in the network. The nonlinearity of the network is also changed to the h-swish function. They introduced two variations called MobileNetV3Large and MobileNetV3-Small to target high and low-resource machines, respectively. The experimental results on ImageNet indicate the superior Top-1 accuracy of MobileNetV3Large over MobileNetV3Small. Interestingly, the performance of MobileNetV2 positioned in-between MobileNetV3Large and MobileNetV3Small in terms of both accuracy and the number of parameters.

3. THAI SIGN LANGUAGE DATASET

We create the static sign dataset from 24 volunteers who express each word by slightly different gestures, e.g., hand positions, left or right hand.

Figure 1. Samples of static sign for the word “one” from 24 volunteers.

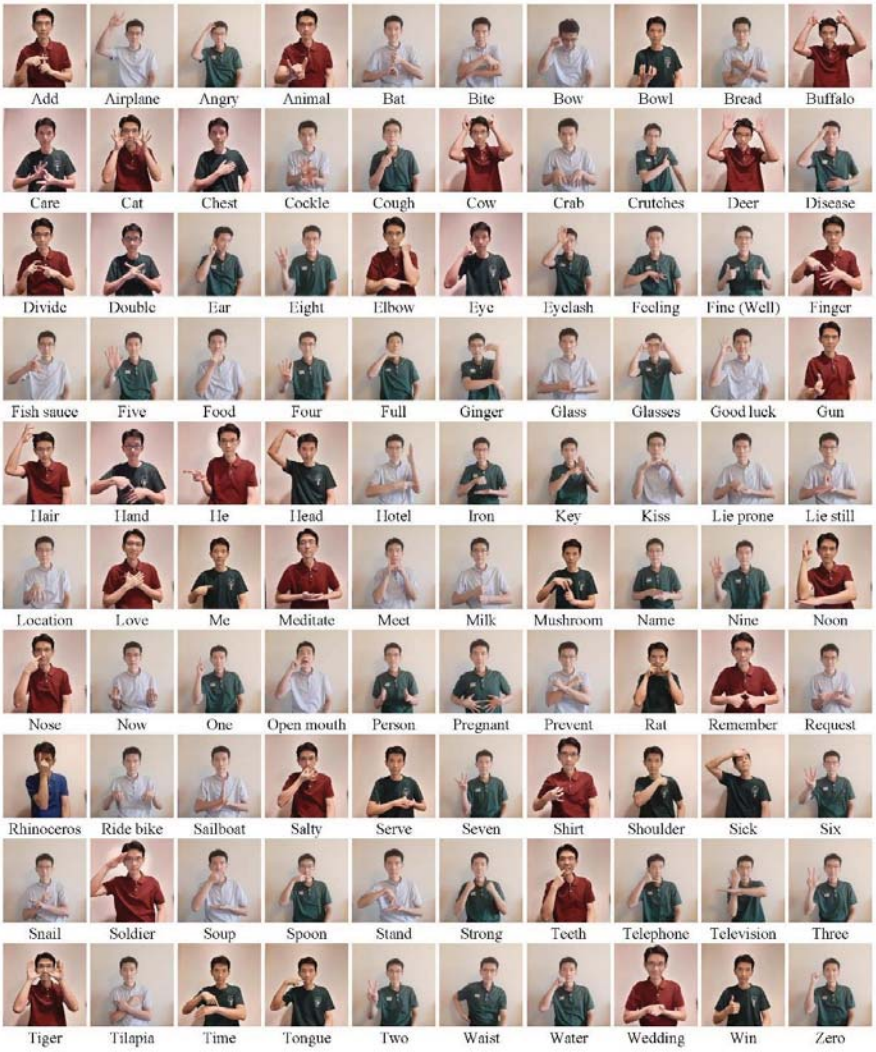

Figure 2. The complete static signs of 100 words.

The dataset composes of 12000 color images equally distributed over 100 classes (words). Each color image has a size of 224×224 pixels and was taken on a light-color background under adequate light. Figure 1 shows 24 samples of static signs for the word “one”. Although the faces of volunteers in Figure 1 are blurred for their privacy, the images used in the experiments do not contain any closure. The complete static signs of 100 words are given in Figure 2 with the consent of the corresponding author for his face disclosure. By taking a glance at the static signs in Figure 2, there are many similarities between different words. For example, {“buffalo”, “cow”}, {“angry”, “hair”}, {“meditate”, “serve”, “wedding”}, {“glass”, “milk”}, {“lie prone”, “lie still”}, {“now”, “ride bike”}, {“water”, “zero”}, {“hotel”, “noon”, “ginger”, “elbow”}, {“tilapia”, “prevent”}, {“teeth”, “tongue”}, and words representing digits 0–9. It is noticeable that finger features are very important to the classification process.

4. PROPOSED MODEL

As a convention in transfer learning, we attach extra layers to a backbone pre-trained model. The output signals from the pre-trained model are flattened and average pooled. Afterward, four consecutive series of batch normalization and dense layers are attached in order as shown in Figure 3.

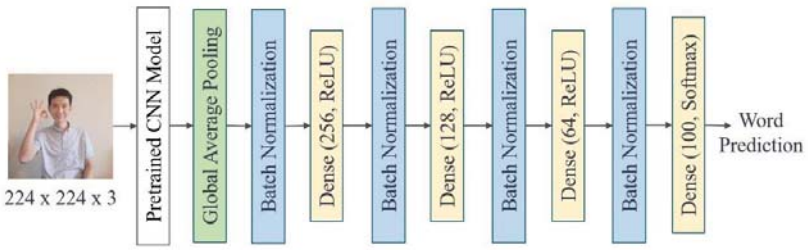

Figure 3. Proposed model.

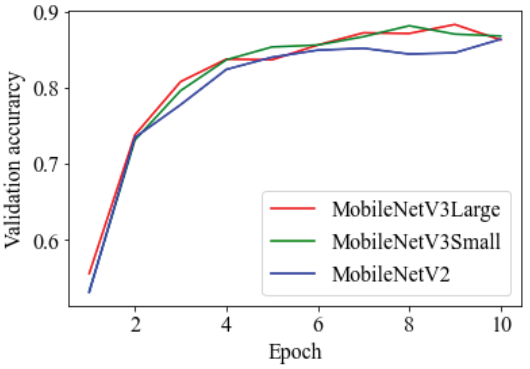

Figure 4. Validation accuracies of three pretrained MobileNets.

We conduct a preliminary experiment to find the best mobile CNN architecture for TSL translation. 12000 instances of the dataset are decomposed into portion of (70:10:20) for (training set: validation set: test set). Three recent models of MobileNet, e.g., MobileNetV2, MobileNetV3Small, and

MobileNetV3Large, are investigated on a preliminary run for 10 epochs. For simplicity, we use Adam optimization and 0.001 learning rate for all models. The preliminary results in Figure 4 indicate that three architectures of MobileNet are competitive in validation accuracy. Although MobileNetV3Large possesses the best performance, its size is double compared to other candidates. Hence, we eliminate MobileNetV3Large due to its unsuitable size for implementation on a mobile application. According to (Silva et al., 2021; Bhalgat et al., 2020), MobileNetV3Small architecture may encounter unstable problems from the h-swish activation function. Besides prediction performance and the model's size, the stability of the mobile application during runtime is another important feature that we pay attention to. As a result, we employ MobileNetV2 as a backbone of CNN based on its stable output, high validation accuracy, and compact size. Although MobileNetV2 has been chosen as a backbone of the proposed model, we also compare the results with other variations by substituting the backbone with other MobileNet architectures in the experimental section.

5. RESULTS

We conduct all experiments on the same PC with the following specifications: Intel Core i9-9900K, DDR4 64GB, SSD 500GB, and RTX2080 graphic card. To understand the difficulty in classifying the dataset, we randomly select 10000 images over 100 classes and plot the distributions of the three most important components of pixels in the image from Principal Component Analysis (PCA). Different colors in Figure 5 represent variations of classes in the dataset. Classification is very hard to accomplish since there are many overlapped classes. We found that the hardest part in classifying the dataset is the ability to capture features from fingers. Later in the discussion section, we demonstrate that the proposed model can correctly capture fingers which are essential features to distinguish similar gestures from different classes.

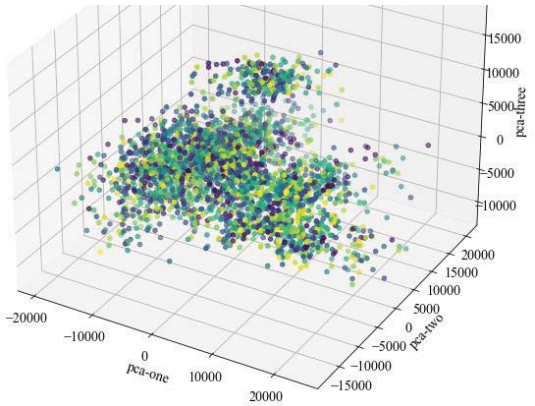

Figure 5. PCA plot for class distribution.

5.1. Experimental Results

Having 120 images per class may not be enough for training the deep learning model. We perform augmentation by adjusting brightness within (0.20, 1.20), rotation within (-10, +10), zoom within (1, 1.20), and flipping horizontally. As a result, the augmented dataset contains 114000 images uniformly distributed over 100 classes. The proportion for separating the dataset is 70% for the training set, 20% for the validation set, and 10% for the test set. The training conditions are listed as follows. Batch size:

128, loss function: sparse categorical cross entropy, metric: sparse categorical accuracy, learning rate: 0.001, and optimizer: Adam. To avoid the overfitting problem, we early stop the training when the validation accuracy cannot improve more than 0.01% within 10 epochs.

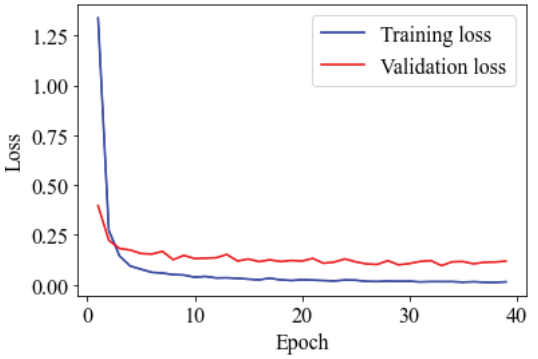

Figure 6. Training progress of the proposed model.

	P	R	F1		P	R	F1		P	R	F1		P	R	F1
Add	0.98	0.93	0.95	Eye	0.92	0.96	0.94	Location	0.98	0.99	0.98	Seven	0.9	0.87	0.89
Airplane	1	0.98	0.99	Eyelash	0.99	1	0.99	Love	0.98	0.97	0.98	Shirt	0.97	0.95	0.96
Angry	1	0.99	1	Feeling	0.99	0.98	0.99	Me	0.92	1	0.95	Shoulder	0.96	0.99	0.97
Animal	0.96	0.96	0.96	Fine	0.99	1	0.99	Meditate	0.98	0.97	0.98	Sick	1	0.98	0.99
Bat	0.99	0.97	0.98	Finger	0.98	0.98	0.98	Meet	0.98	0.99	0.99	Six	0.91	0.91	0.91
Bite	1	1	1	Fish Sauce	0.99	0.95	0.97	Milk	0.94	0.96	0.95	Snail	0.96	0.98	0.97
Bow the Head	0.99	1	0.99	Five	0.91	0.94	0.92	Mushroom	0.99	0.97	0.98	Soldier	0.97	1	0.98
Bowl	1	0.96	0.98	Food	0.99	1	1	Name	1	0.98	0.99	Soup	1	1	1
Bread	0.9	0.96	0.93	Four	0.91	0.91	0.91	Nine	0.91	0.94	0.92	Spoon	0.96	0.97	0.97
Buffalo	0.97	0.99	0.98	Full	0.99	0.99	0.99	Noon	0.99	1	0.99	Stand	0.99	0.99	0.99
Care	1	0.99	1	Ginger	1	1	1	Nose	0.94	0.89	0.91	Strong	0.98	0.99	0.98
Cat	0.99	0.99	0.99	Glass	0.96	0.98	0.97	Now	1	0.99	0.99	Teeth	0.94	0.93	0.94
Chest	0.95	0.99	0.97	Glasses	1	1	1	One	0.93	0.96	0.95	Telephone	0.99	0.98	0.98
Cockle	0.99	0.99	0.99	Good Luck	0.96	0.97	0.97	Open Mouth	1	0.99	1	Television	1	1	1
Cough	1	0.98	0.99	Gun	0.97	0.94	0.95	Person	0.99	1	1	Three	0.94	0.96	0.95
Cow	0.98	0.97	0.98	Hair	0.99	1	0.99	Pregnant	1	1	1	Tiger	0.98	0.99	0.98
Crab	0.98	1	0.99	Hand	0.97	0.98	0.98	Prevent	0.98	0.99	0.99	Tilapia	0.95	0.89	0.92
Crutches	0.99	0.98	0.98	He	0.98	0.98	0.98	Rat	0.99	0.99	0.99	Time	0.97	0.99	0.98
Deer	1	0.98	0.99	Head	0.99	0.99	0.99	Remember	0.99	0.98	0.99	Tongue	0.91	0.93	0.92
Disease	1	1	1	Hotel	0.99	0.99	0.99	Request	0.99	0.99	0.99	Two	0.94	0.9	0.92
Divide	0.95	0.98	0.96	Iron	1	0.98	0.99	Rhinoceros	0.97	1	0.98	Waist	1	1	1
Double	0.98	0.99	0.98	Key	1	0.97	0.98	Ride Mt.Cycle	0.99	0.98	0.99	Water	0.97	0.98	0.97
Ear	0.98	0.97	0.97	Kiss	0.99	1	0.99	Sailboat	0.97	0.99	0.98	Wedding	0.97	0.96	0.97
Eight	0.93	0.87	0.9	Lie Prone	0.97	0.97	0.97	Salty	1	0.98	0.99	Win	0.93	0.95	0.94
Elbow	0.99	0.98	0.98	Lie Still	1	0.95	0.97	Serve	0.96	0.96	0.96	Zero	0.98	0.96	0.97

Figure 7. Precision (P), Recall (R), and F1-Score (F1).

Figure 6 illustrates the training progress of the proposed model. It learns patterns quickly as the training loss suddenly drops during the early period. The training process stops at the 39th epoch since there is no improvement in the validation accuracy within 10 epochs. Figure 7 shows results of precision, recall, and f1-score for each class in the test dataset. In every class, the proposed model produces high

relevancy on all predictions. It also attains a high ratio between relevant predicted instances and all relevant instances. As a result, f1-scores for all classes are impressively high. Moreover, 13 out of 100 classes are easy to classify since the proposed model achieves a perfect f1-score. The most difficult word which has the lowest f1-score, but still has a high value, is “seven”. It is a three-finger gesture that is similar to those of the words “eight”, “nine”, “six”, and “three”. This result implies that finger gestures play an important role in TSL classification. Appropriate CNN architecture needs to capture these important features and maintain them throughout deep convolution layers. In the discussion section, we visually illustrate the convolution filters and the activation images from various convolution layers to demonstrate that the proposed method can handle important features from the image very well.

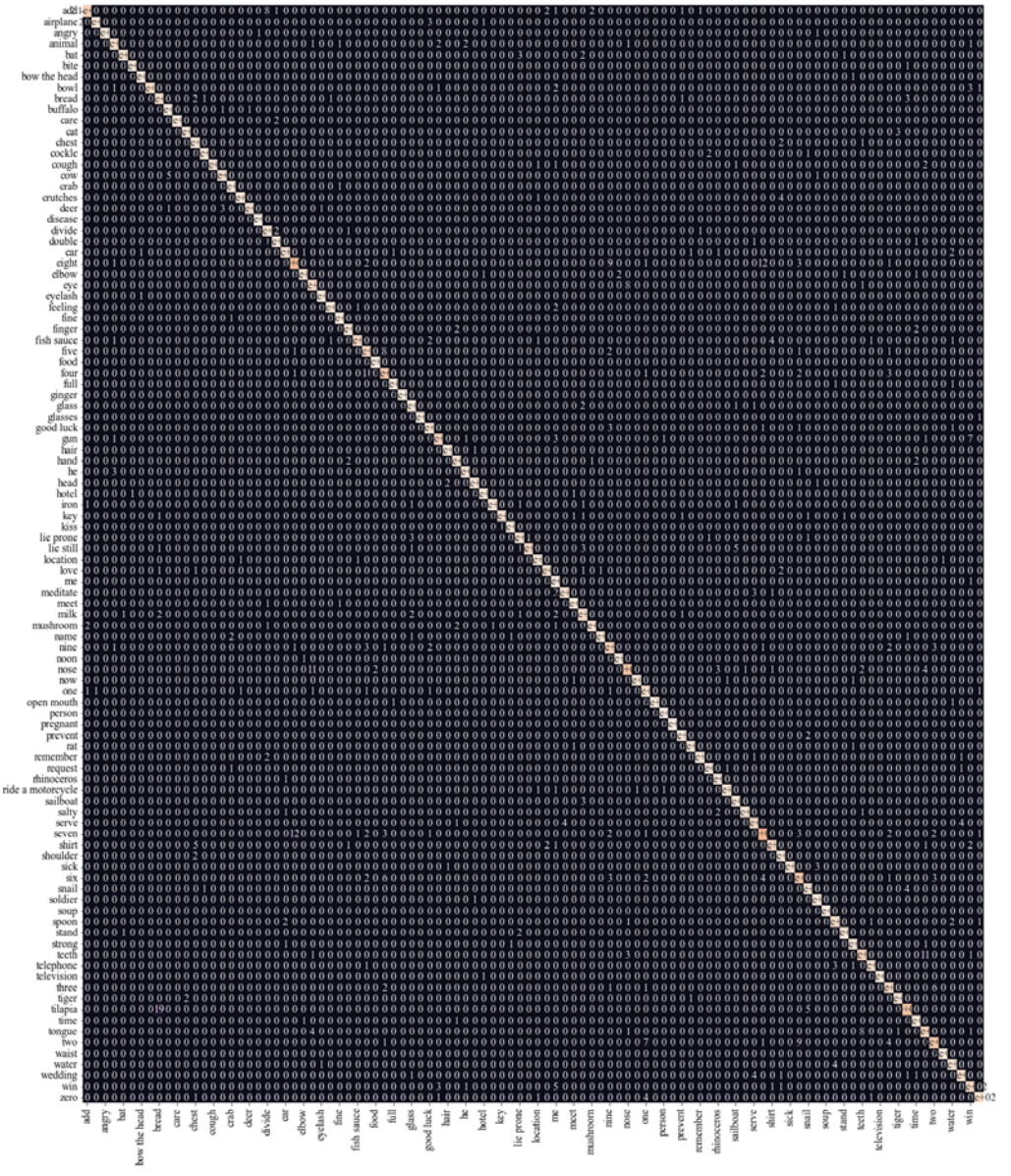

Figure 8. Confusion matrix.

To visually understand a summary of prediction results on classification, the confusion matrix is illustrated in Figure 8. The matrix indicates almost perfect results of the proposed method as entries on the main diagonal contain strong intensity indicating a high degree of matching between predictions and ground truths.

5.2. Experimental Comparisons

The comparison of the training progress among models is illustrated in Figure 9. We represent each work by the surname of the first author appeared in the paper for the sake of convenience; Wadhawan: (Wadhawan et al., 2020), Rao: (Rao et al., 2018), Xie: (Xie and Ma, 2019), and Masood: (Masood et al., 2018). To prevent models from overfitting problem, the stopping criterion for all models is to early stop when the validation accuracy improvement does not exceed 0.01% within 10 consecutive epochs. Three MobileNet architectures spend around 39 – 45 epochs for weight training while Wadhawan, Rao, Xie, and Masood take 86, 33, 69, and 95 epochs, respectively. Both loss and accuracy of most models vary similarly during the training period, except for those of Xie which produces ripple results. MobileNet architectures swiftly learn the dataset within a few epochs and gradually improve loss and accuracy in the later period while other models slowly learn the dataset.

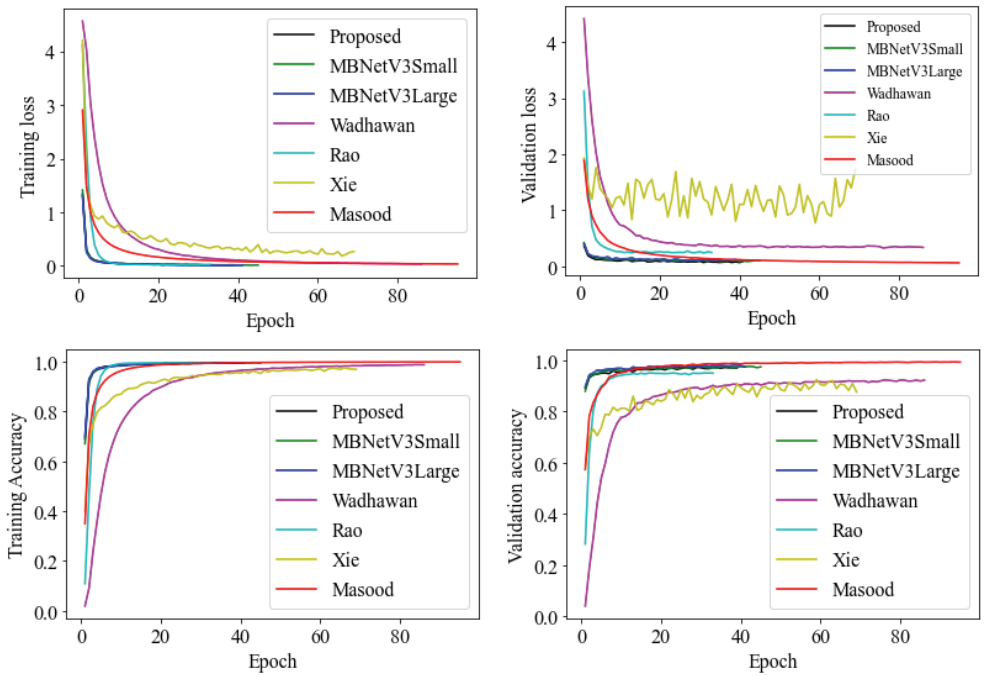

Figure 9. Comparison of training progress among seven methods.

The comparison of test accuracy, total parameters, and size among models is shown in Table 1. MobileNet architectures and Masood model achieve competitive and considerably high accuracies. However, Masood and MobileNetV3Large models contain a significantly larger number of parameters

and model size making them inappropriate for implementing on a mobile application. MobileNetV3Small looks competitive with the proposed method in terms of both accuracy and size. However, an unstable issue of its accuracy loss from h-swish activation may malfunction during runtime. As a result, the proposed model is the most suitable representative of the MobileNet architecture.

Table 1: Test accuracies of four methods.

Model	Test accuracy	Total parameters	Size (MB)
Proposed (MobileNetV2)	97.32	2,640,484	13.81
MobileNetV3Small	97.39	1,845,908	10.39
MobileNetV3Large	98.04	4,608,932	21.83
Wadhawan	91.82	4,073,588	15.94
Rao	95.11	6,749,476	26.40
Xie (ResNet50)	92.12	33,623,012	210.26
Masood (VGG16)	98.02	17,223,588	67.35

To have the in-depth analysis beyond test accuracy, comparisons on precision, recall, and f1-score of all models are given in Figure 10, 11, and 12, respectively. Both proposed and Masood models are competitive in classifying unseen data. Their predictions are so accurate as graphs hardly drop. Conversely, Wadhawan and Xie models have poor results on precision, recall, and f1-score for over ten classes. Another fact that can be concluded from Figures 10 – 12 is that the performance of most models drops in digit classes, e.g., two, three, four, five, six, seven, eight, and nine, since hand gestures from those classes are very similar. Therefore, having an extra mechanism that takes special care of finger features is essential for better recognition.

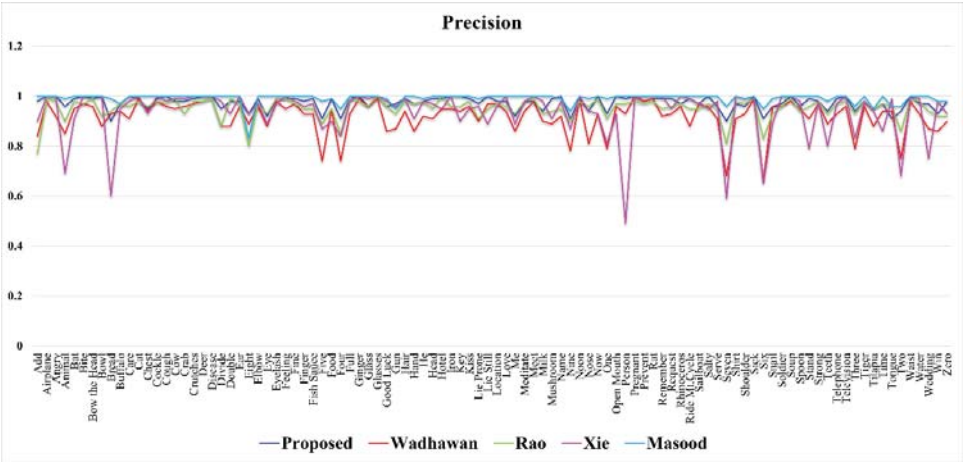

Figure 10. Comparison of precision.

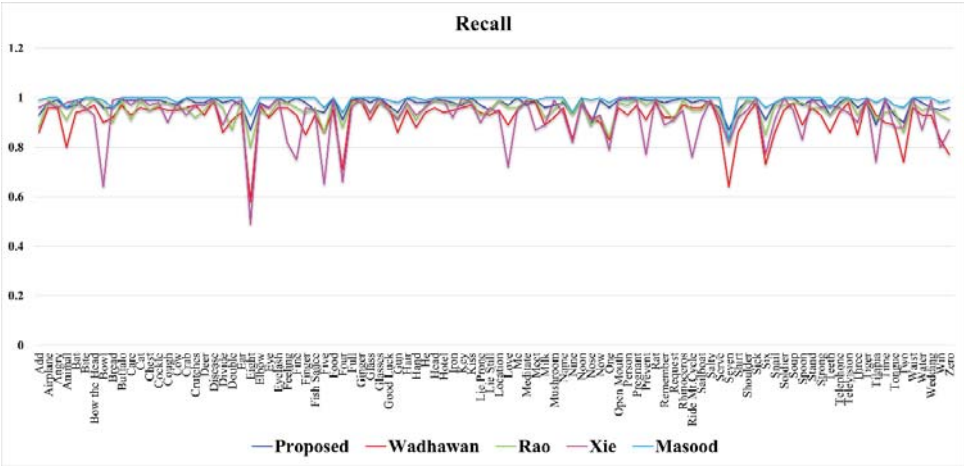

Figure 11. Comparison of recall.

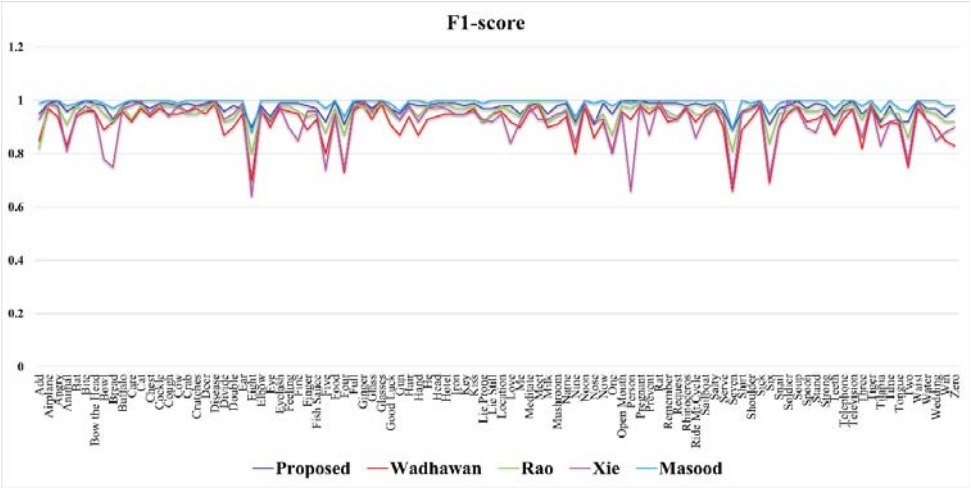

Figure 12. Comparison of f1-score.

6. DISCUSSION AND CONCLUSION

We provide a concrete example showing that the proposed model can captures crucial features from fingers in the input image. We visualize patterns that responded to CNN filters in our model compared to those from other approaches. Similar to the deconvolution network in (Zeiler and Fergus, 2014), we reconstruct an approximate version of the features from the convolutional layer by creating an input image that maximizes the activation of each filter to see the pattern that it responds to. Figure 13 shows pattern visualizations from the first 9 filters of the deep convolution layer before passing the activations through dense layers. According to a visualization technique (Zeiler and Fergus, 2014), filters in deeper layers capture delicate patterns from the previous activation. The area of the filter which does not respond to the input activation generates black or grey blobs without any pattern or texture, i.e., dead

features. This incorrect representation happens due to an inadequate sampling rate. It is perceptible from Figure 13 that Wadhawan, Rao, and Xie models contain dead features resulting in mispredictions on the input image as shown in Figure 14. They cannot preserve important finger features and predict wrong results. On the other hand, the proposed and Masood models do not contain any dead features. They extract essential features, as shown in Figure 14 that all three fingers are still preserved, before transferring them to the dense layer resulting in a superior prediction result. It is worth noting that activations in Figure 14 are derived from applying filters in Figure 13 to the corresponding activation in the previous layer. In addition, activation sizes in Figure 14 from all models are approximately the same.

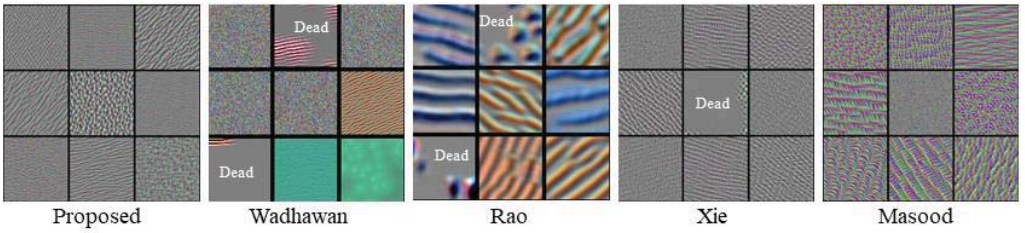

Figure 13. Filter visualization from the deep convolution layer.

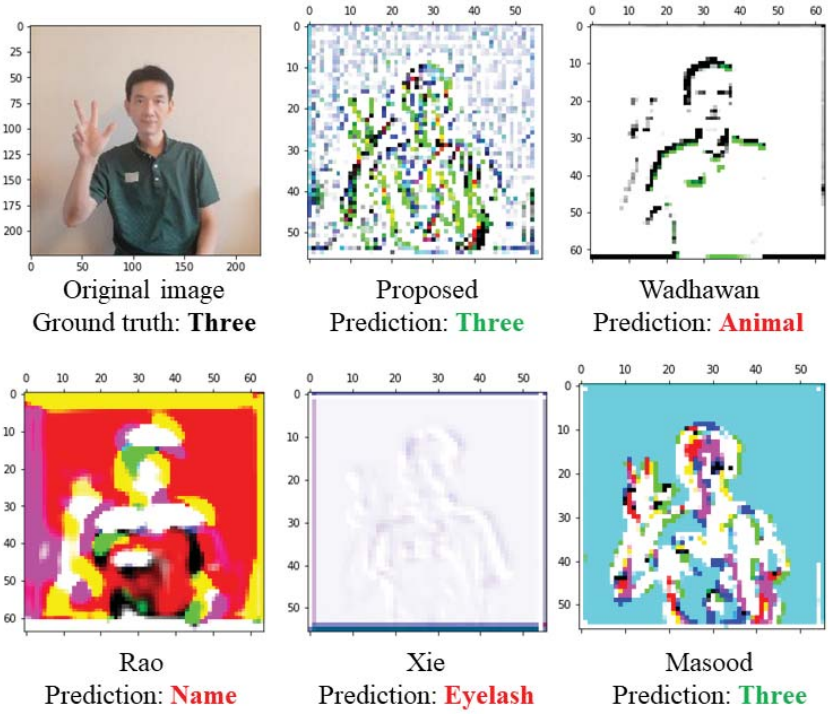

Figure 14. Activation visualization of the deep convolution layer.

Although the proposed and Masood models are competitive in performance, the proposed model is about five times smaller than Masood model. As a result, our model is suitable for learning the TSL dataset on a mobile application that requires low memory and computing power consumption.

In conclusion, we propose a compact deep learning model for TSL translation that can be applied to the mobile application. Our model not only attains excellent recognition performance but also holds a reasonably low number of parameters, hence a small-size model. We employ MobileNetV2, which is more stable than other MobileNet models, as the backbone and attach it with four series of batch normalization and dense layers. The experimental results on the 100-class dataset collected from 24 volunteers indicate superior accuracy, precision, recall, and f1-score of the proposed model over four state-of-the-art approaches. The feature visualization from the activation of convolution layers indicates that the proposed model can preserve essential features from fingers throughout the convolution path. One possible future work is to invent a more advanced technique for finger extraction to improve the digit prediction (0–9) which seems to be the most difficult classification among all classes.

7. REFERENCES

Adhan, S., Pintavirooj, C., 2016, Thai sign language recognition by using geometric invariant feature and ANN classification. Biomedical Engineering International Conference (BMEiCON), Laung Prabang, Laos, 1–4.

Adithya, V., Rajesh, R., 2020, A Deep Convolutional Neural Network Approach for Static Hand Gesture Recognition. Procedia Computer Science **171**, 2353–2361.

Bengio, Y., 2016, Deep Learning. Adaptive Computation and Machine Learning Series. London, England, MIT Press.

Bhalgat, Y., Lee, J., Nagel, M., Blankevoort, T., Kwak, N., 2020, LSQ+: Improving low-bit quantization through learnable offsets and better initialization. IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, USA, 2978–2985.

Boonya, R., 2008, Sign Language: Language of the Deaf. Journal of Ratchasuda College for Research and Development of Persons with Disabilities **4(1)**, 77-94.

Chaikaew, A., Somkuan, K., Yuyen, T., 2021, Thai Sign Language Recognition: An Application of Deep Neural Network. Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunication Engineering, Cha-am, Thailand, 128-131.

Howard, A., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017, MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv:1704.04861.

Howard, A., Sandler, M., Chu, G., Chen, L. C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q. V., Adam, H., 2019, Searching for MobileNetV3, IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 1314–1324.

Jani, A. B., Kotak, N. A., Roy, A. K., 2018, Sensor Based Hand Gesture Recognition System for English Alphabets Used in Sign Language of Deaf-Mute People. IEEE SENSORS, New Delhi, India, 1–4.

Lim, K. M., Tan, A.W.C., Lee, C.P., Tan, S. C., 2019, Isolated sign language recognition using Convolutional Neural Network hand modelling and Hand Energy Image. Multimed Tools Appl. **78**, 19917–19944.

Masood, S., Thuwal, H., Srivastava, A., 2018, American Sign Language Character Recognition Using Convolution Neural Network. Smart Computing and Informatics **78**, 403–412.

Pariwat, T., Seresangtakul, P., 2017, Thai finger-spelling sign language recognition using global and local features with SVM. International Conference on Knowledge and Smart Technology (KST), Chonburi, Thailand, 116–120.

Phothiwetchakun, W., Rakthanmanon, T., 2021, Thai Fingerspelling Recognition Using Hand Landmark Clustering. International Computer Science and Engineering Conference (ICSEC), Chiang Rai, Thailand, 256–261.

Rahaman, M. A., Jasim, M., Ali, M. H., Hasanuzzaman, M., 2014, Real-time computer vision-based Bengali Sign Language recognition. International Conference on Computer and Information Technology (ICCIT), Dhaka, Bangladesh, 192–197.

Rao, G. A., Syamala, K., Kishore, P. V. V., Sastry, A. S. C. S., 2018, Deep convolutional neural networks for sign language recognition. Conference on Signal Processing And Communication Engineering Systems (SPACES), Vijayawada, India, 194–197.

Saengsri, S., Niennattrakul, V., Ratanamahatana, C. A., 2012, TFRS: Thai finger-spelling sign language recognition system. International Conference on Digital Information and Communication Technology and it's Applications (DICTAP), Bangkok, Thailand, 457–462.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L. C., 2018, MobileNetV2: Inverted Residuals and Linear Bottlenecks. The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, USA, 4510–4520.

Silva, D., Sousa, A., Costa, V., 2021, A Comparative Analysis for 2D Object Recognition: A Case Study with Tactode Puzzle-Like Tiles. Journal of Imaging **7(4):65**, 1–20.

Sripairojthikoon, N., Harnsomburana, J., 2019, Thai Sign Language Recognition Using 3D Convolutional Neural Networks. Proceedings of the 7th International Conference on Computer and Communications Management, Bangkok, Thailand, 186–189.

Tornay, S., Razavi, M., Magimai.-Doss, M., 2020, Towards Multilingual Sign Language Recognition. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 6309–6313.

Wadhawan, A., Kumar, P., 2020, Deep learning-based sign language recognition system for static signs. Neural Comput & Applic. **32**, 7957–7968.

Wangchuk, K., Riyamongkol, P., Waranusast, R., 2021, Real-time Bhutanese Sign Language digits recognition system using Convolutional Neural Network. ICT Express **7(2)**, 215–220.

Xie, M., Ma, X., 2019, End-to-End Residual Neural Network with Data Augmentation for Sign Language Recognition. IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chengdu, China, 1629–1633.

Zeiler, M.D., Fergus, R., 2014, Visualizing and Understanding Convolutional Networks. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds) Computer Vision – ECCV 2014. Lecture Notes in Computer Science **8689**, 1–11.