Thai Finger-Spelling using Vision Transformer

Kullawat Chaowanawatee, Kittasil Silanon*, Thitinan Kliangsuwan
College of Computing, Prince of Songkla University Phuket Campus, Kathu, Phuket, Thailand

***Abstract*—In this paper, we present a finger-spelling recognition system that is based on Thai Sign Language (TFS) and employs a deep learning model called vision transformer. We extracted the 15 characters of the Thai alphabet from publicly available and our collected datasets to establish the recognition system. To train the learning model, we employed four EVA-02 vision transformer models, each of which showed impressive performance across different model sizes. We conducted four experiments to determine the most effective performance model. In Experiment 1, we directly trained the model to compare its performance. In Experiment 2, we used augmentation techniques to generate additional datasets. Experiment 3 utilized the Test-Time Augmentation (TTA) technique to generate test images with random variations. Lastly, in Experiment 4, we used Pseudo-Labelling (labeling labeled and unlabeled data) in each batch to train the model network. Furthermore, we developed a mobile application that collects user image data and provides helpful information related to finger-spelling, such as meanings, gestures, and usage examples.**

Keywords*—*Thai finger-spelling; vision transformer; deep learning; image recognition

I. INTRODUCTION

Sign language is a method of communication used by individuals who are deaf or hard of hearing. There are two primary types of sign language: vocabulary signs, which use hand shapes, movements, and facial expressions to convey word meanings, and finger-spelling, which uses only hand shapes to spell out technical words, such as names and locations. However, finger-spelling is not commonly used in everyday conversations, so many deaf individuals, especially children, struggle with it. Several finger-spelling recognition research studies have been proposed to help their skills. For instance, A. Deza and D. Hasan [1] used pre-processing techniques such as boosting contrast, edge enhancement, and the Convolution Neural Network (CNN) model to classify the American Sign Language (ASL) finger-spelling dataset. These techniques enabled the authors to achieve a validation accuracy of approximately 77%. R. A. Alawwad et al. [2] introduced Arabic Sign Language (ArSL) recognition systems using a Faster Region-based Convolutional Neural Network (R-CNN). The proposed approach yielded 93% accuracy and confirmed the robustness of background variations. K. R. Prajwal et al. [3] proposed a British Sign Language (BSL) finger-spelling recognition using transformer architecture to train models with noisy labels. Researchers employed a multi-stage training approach to achieve better performance. E. M. Martin and F. M. Espejo in [4] presented a system to interpret the Spanish Sign Language (LSE) finger-spelling. The pre-trained CNN model and Recurrent Neural Network (RNN) have been tested and compared. The CNN obtained a much better accuracy, with 96.42% being the maximum value. V. J. Schmalz [5] applied deep learning and fine-tuning techniques to build an automatic recognition system for Italian Sign Language (LIS) finger-spelling. The proposed CNN and VGG19 models were used for large-scale image and video recognition. The system obtained an accuracy of 99% with VGG19 and 97% with the CNN architecture. E. J. Ong et al. [6] suggested a Sequential Pattern Tree-based (SP-Tree) multi-class classifier for German Sign Language (DGS) and Greek Sign Language (GSL) finger-spelling recognition. Their proposed SP-Tree Boosting algorithm-based recognition model performs better than the Hidden Markov Model (HMM). X. Jiang et al. [7] proposed a novel finger-spelling identification method for Chinese Sign Language (CSL) via AlexNet-based transfer learning and Adam optimizer, which tested four different configurations of transfer learning. Although there has been research on finger-spelling in many languages, Thai Sign Language (TSL) finger-spelling will be the main focus of this work. P. Nakjai et al. [8] investigated a YOLO based on the convolution neural network architecture to localize and classify TSL finger-spelling. The system achieved the mean Average Precision (mAP) of 82.06% under a complex background and 84.99 % under a plain background. S. Lata and O. Surinta [9] proposed an end-to-end TSL finger-spelling recognition based on the YOLOv3 objection detection framework to create the most robust model with high recognition performance. P. Nakjai and T. Katanyukul [10] proposed automatic TSL finger-spelling recognition using CNN-based and Histogram of Oriented Gradients (HOG) based approaches. Experimental results have shown the viability of the proposed method, which achieves mAP at 91.26%. J. Sanalohit and T. Katanyyukul [11] invented MediaPipe Hands (MPH) trained model for hand-keypoint detection. The MPH can satisfactorily address single-hand schemes with accuracy of 84.57%

According to studies, the CNN model is the most commonly employed technology in finger-spelling research. Various model architectures [12], such as Xception, VGG16, InceptionV3, or ConvNeXt, are provided. However, the CNN model has some drawbacks, such as requiring a lot of data and computational resources and being sensitive to spatial distortions and variations. For this reason, a new deep learning technology called vision transformer (ViT) has been introduced for image recognition applications. The ViT offer a different approach by treating images as sequences of patches and applying self-attention to capture global dependencies and contextual information. This allows them to learn from less data and perform very well on image classification tasks. Therefore, we are interested in using the ViT approach due to its advantages in our work for TSL finger-spelling recognition. To find out the most effective performance recognition model. We performed four experiments. In Experiment 1, the model will be trained directly to compare model performance.

Experiment 2 will generate additional data sets using the augmentation technique. Experiment 3 uses the Test-Time Augmentation (TTA) technique to generate test images with random variations. Experiment 4 employed Pseudo-Labelling (labeled and unlabeled data) in each batch to train the model network. We organize the paper as follows. Section II describes the details of the dataset and ViT model. We present the experiment details and results in Section III. Section IV shows the implemented application using the recognition model as its classifier. Finally, we conclude this paper in SectionV.

II. Datasets and Vision Transformer Model for Image Classification

In this section, we first describe the classes of the dataset. Then we describe the details of vision transformer model and it's characteristic.

A. Datasets forTSL Finger-Spelling

This system recognizes 15 letters of TSL finger-spelling (see Fig. 1 and Table I) which use static-single-hand postures. We collected image hand postures from public datasets. The dataset was randomly partitioned into training and test sets, with an 80:20 ratios, resulting in 1,522 and 381 images in the respective subsets.

TABLE I. Static-Single-Hand Posture for TSL Finger-Spelling

Letter	Hand posture
" ก " (Ko kai)	Point the index and middle fingers with a thumb between them.
" ต " (To tao)	Form a fist with your thumb between the middle and index finger.
" ส " (So suea)	Make a fist and put your thumb on top of your fingers.
" พ " (Po phan)	Press your thumb against your middle finger with index pointed.
" ห " (Ho hip)	Stick out your middle and index fingers.
" บ " (Bo bimai)	Hold your fingers together with your thumb across your palm.
" ร " (Ro ruea)	Cross your index finger over your middle finger.
" ว " (Wo waen)	Hold up three fingers and spread them apart
" ด " (Do dek)	Touch your fingertips to your thumb and point.
" ฟ " (Fo fan)	Press your index finger and thumb together with straight fingers.
" ล " (Lo ling)	Form an L-shape with your thumb and index finger.
" ย " (Yo yak)	Stick out your pinkie and thumb.
" ม " (Mo ma)	Hold an invisible ball and poke your thumb through.
" น " (No nue)	Poke your thumb between your middle and ring.
" อ " (O ang)	Make a fist with your thumb pointing up.

B. Vision Transformer Model

The Vision Transformer (ViT) [13] are a type of deep learning architecture that uses the Transformer architecture. This is a significant departure from traditional computer vision architectures that rely on convolutional neural networks (CNNs). The ViT model works by first dividing the image into a sequence of patches. Each patch is then represented as a vector. These vectors for each patch are subsequently fed into a transformer encoder, a stack of self-attention layers. Self-attention is a mechanism that enables the model to learn long-range dependencies between the patches. This is crucial for image classification, as it allows the model to understand how different parts of an image contribute to its overall label. The output of the Transformer encoder is a sequence of vectors representing the image's features. The features are then used to classify the image. The ViTs have emerged as a powerful and promising alternative to CNNs in computer vision. Their ability to capture long-range dependencies, global attention, and data efficiency has made them a preferred choice for various computer vision tasks.

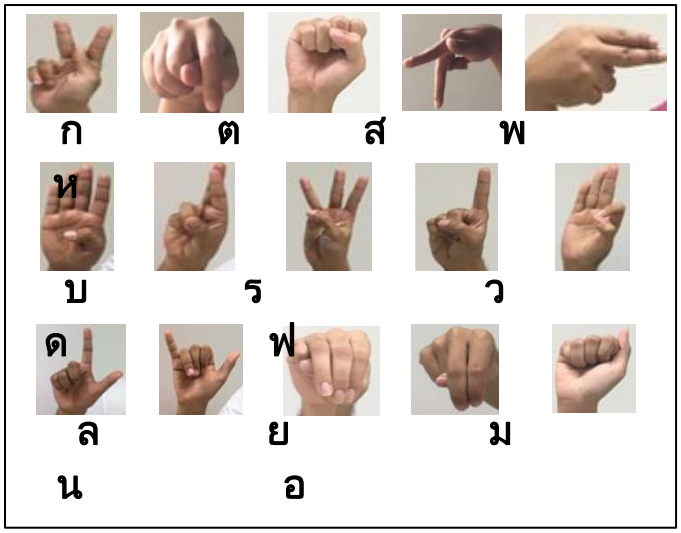


Fig. 1. Static-single-hand posture for TSL finger-spelling.

Currently, the ViT model that is gaining attention is a model named EVA02, presented by Y. Fang et al. [14]. The EVA-02 architecture is designed to be more efficient than previous Transformer-based models, enabling it to process images with greater speed and reduced computational resources. This superior performance makes it a valuable tool for tasks like object detection, semantic segmentation, and visual understanding. There are four variations of the EVA-02 model, with sizes ranging from 6M to 304M parameters. These models perform magnificently in image recognition tasks (see Table II).

TABLE II. Four Variations of EVA-02 Models

EVA-02 Model	Parameters (Million)	FLOPs (Billion)	Top-1 Accuracy on ImageNet (%)
EVA-02 Tiny	10M	0.6	76.2
EVA-02 Small	31M	1.6	77.5
EVA-02 Base	101M	3.3	79
EVA-02 Large	304M	7.6	81.4

III. Experimental Works

Four experiments were conducted to assess the classification performance of TSL finger-spelling recognition as discussed in this section.

A. Vision Transformer Training

In this experiment, we utilized four variations of the EVA-02 model, namely Tiny, Small, Base, and Large. All models

were trained directly on the dataset, with the input image size set to 224x224 pixels. We considered some crucial hyper parameters, including the learning rate of 2e-3, batch size set to 16, cross-entropy loss function, and 30 epochs for training the model. The K-Fold Cross Validation technique in [15] evaluates the predictive model. The dataset is divided into five subsets or folds. The model is trained and evaluated five times, using a different fold as the validation set. This technique aids in model assessment, selection, and hyper parameter tuning, providing a more reliable measure of a model's effectiveness. The model's generalization performance is evaluated using accuracy and weighted F1-score. Accuracy is a simple measure of overall correctness. At the same time, the weighted F1-score is a more sophisticated metric that considers class imbalances. Testing model results are shown in Table III. The large EVA-02 model at 1st fold achieved the best performance at 94.6% accuracy and 94.4% weighted F1-score. Fig. 2 shows the classification results of all 15 classes as a part of class performance analytics. Most errors occur in class "Mo ma" which produces incorrect predictions for classes "So suea" and "To tao" as a consequence of similar hand posture shapes, as displayed in Fig. 3. To improve the model's performance, we will discuss in the next section about the possibility of increasing the number of training examples.

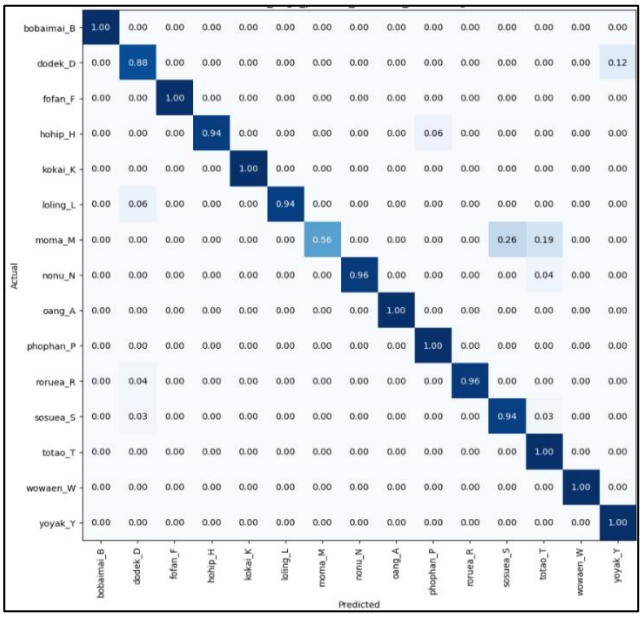


Fig. 2. Confusion matrix of the large EVA-02 at 1st fold.

Fig. 3. Classification error example of the large EVA-02 at 1st fold.

B. Data Augmentation

Data augmentation [16] is typically used during model training that expands the training set with modified copies of samples from the training dataset. This experiment simulates horizontal/vertical flipping, space transformation, random focus, noise addition, and slight rotation to train the model on more generalized data. Fig. 4 shows an example of a dataset after augmentation. Testing model results with the augmentation dataset are shown in Table IV. The large EVA-02 model at 3rd fold achieved the best performance at 94.6% accuracy and 94.5% weighted F1-score. Fig. 5 shows the classification results of all 15 classes as a part of class performance analytics. The model performance was improved when using the data transformation method, especially class "Mo ma"; the accuracy result increased to 70%. Class "So suea" and class "To tao" recognition errors decreased to 15% and 7%, respectively. For other classes, such as class "Po phan" or class "Yo yak", the accuracy dropped to 77% and 94% individually. In addition to augmenting the training dataset to improve the model's performance, we can also increase the number of testing datasets to enhance the accuracy of the model, which we will discuss in the following experiment.

TABLE III. MODEL PERFORMANCE EVALUATION

Model	Fold	Test dataset	
		Accuracy	Weighted F1
Tiny	1	0.6846	0.6693
	2	0.6590	0.6467
	3	0.6692	0.6608
	4	0.6821	0.6770
	5	0.7077	0.6835
Small	1	0.5897	0.5729
	2	0.6051	0.5809
	3	0.5462	0.5322
	4	0.6436	0.6366
	5	0.5590	0.5416
Base	1	0.8846	0.8784
	2	0.8513	0.8496
	3	0.8744	0.8723
	4	0.8897	0.8874
	5	0.9231	0.9210
Large	1	0.9462	0.9441
	2	0.8513	0.8377
	3	0.9128	0.9093
	4	0.8872	0.8832
	5	0.9359	0.9357

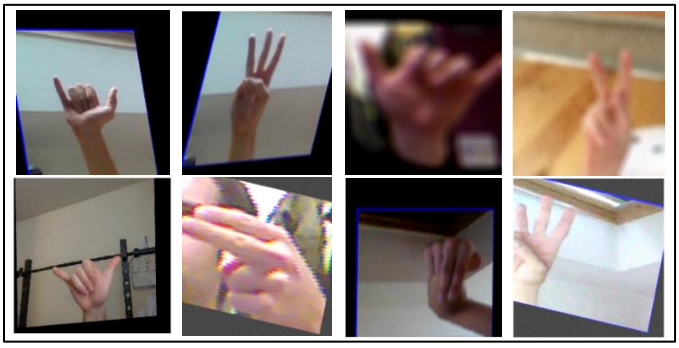


Fig. 4. Example of the data augmentation for training set.

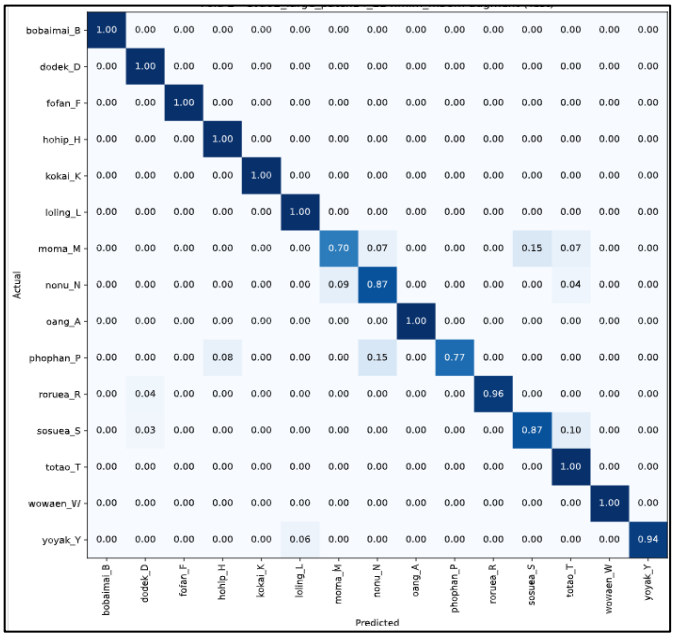


Fig. 5. Confusion matrix of the large EVA-02 at 3rd fold with data augmentation.

TABLE IV. DATA AUGMENTATION MODEL PERFORMANCE EVALUATION

Model	Fold	Test dataset	
		Accuracy	Weighted F1
Tiny	1	0.6590	0.6455
	2	0.7667	0.7521
	3	0.7231	0.7213
	4	0.8385	0.8303
	5	0.7821	0.7755
Small	1	0.8205	0.8082
	2	0.7051	0.6923
	3	0.8051	0.8016
	4	0.8308	0.8243
	5	0.8359	0.8270
Base	1	0.8641	0.8549
	2	0.9077	0.9034
	3	0.9231	0.9217
	4	0.9231	0.9219
	5	0.8769	0.8714
Large	1	0.9462	0.9435
	2	0.9462	0.9428
	3	0.9462	0.9450
	4	0.7513	0.7364
	5	0.9051	0.9041

C. Test-Time Data Augmentation

Test-Time Data Augmentation (TTA) [17] is a technique that can boost a model performance by applying augmentation to the test dataset. TTA is performed on multiple augmented copies of each image in the test dataset, having the model predict each and then returning an ensemble of those predictions to get a higher overall accuracy. For example, the image in Fig. 6 applies two transformations (rotation and color change) together with the original image. All of these images are passed to the same model and the results are averaged.

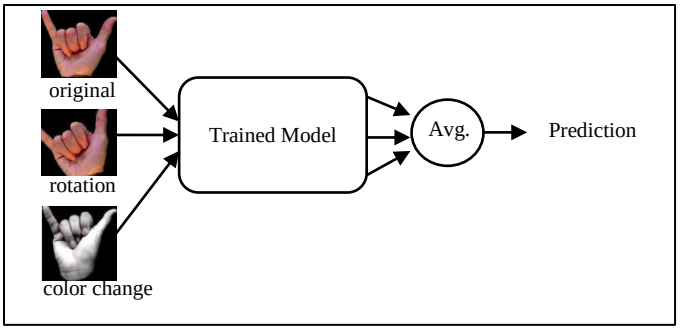


Fig. 6. Test-time data augmentation example.

TTA can be implemented to a model that has already been trained because, unlike training dataset augmentation, no model changes are required. We compared performance results of test dataset without TTA from previous experiment and test dataset using the TTA technique of models. From Table V, the large EVA-02 model at 3rd fold using TTA technique achieved best performance at 95.9% accuracy and 95.8% weighted F1-score. Fig. 7 shows the classification results of all 15 classes as a part of class performance analytics. Class "**Mo ma**" was improved, with accuracy increasing to 77%, while the other classes almost all exceeded 80% accuracy. In the following experiment, we discussed the ways to combine labeled and unlabeled dataset to improve the model's performance.

TABLE V. TEST-TIME DATA AUGMENTATION MODEL PERFORMANCE EVALUATION

Model	Fold	Test dataset without TTA		Test dataset using TTA	
		Accuracy	Weighted F1	Accuracy	Weighted F1
Tiny	1	0.6590	0.6455	0.6744	0.6597
	2	0.7667	0.7521	0.7872	0.7753
	3	0.7231	0.7213	0.7205	0.7152
	4	0.8385	0.8303	0.8538	0.8436
	5	0.7821	0.7755	0.7744	0.7677
Small	1	0.8205	0.8082	0.8282	0.8176
	2	0.7051	0.6923	0.7103	0.7005
	3	0.8051	0.8016	0.8026	0.7965
	4	0.8308	0.8243	0.8333	0.8229
	5	0.8359	0.8270	0.8179	0.8082
Base	1	0.8641	0.8549	0.8769	0.8679
	2	0.9077	0.9034	0.9154	0.9127
	3	0.9231	0.9217	0.9205	0.9196
	4	0.9231	0.9219	0.9359	0.9349
	5	0.8769	0.8714	0.8949	0.8888
Large	1	0.9462	0.9435	0.9513	0.9491
	2	0.9462	0.9428	0.9436	0.9402
	3	0.9462	0.9450	0.9590	0.9586
	4	0.7513	0.7364	0.7718	0.7588
	5	0.9051	0.9041	0.9026	0.9011

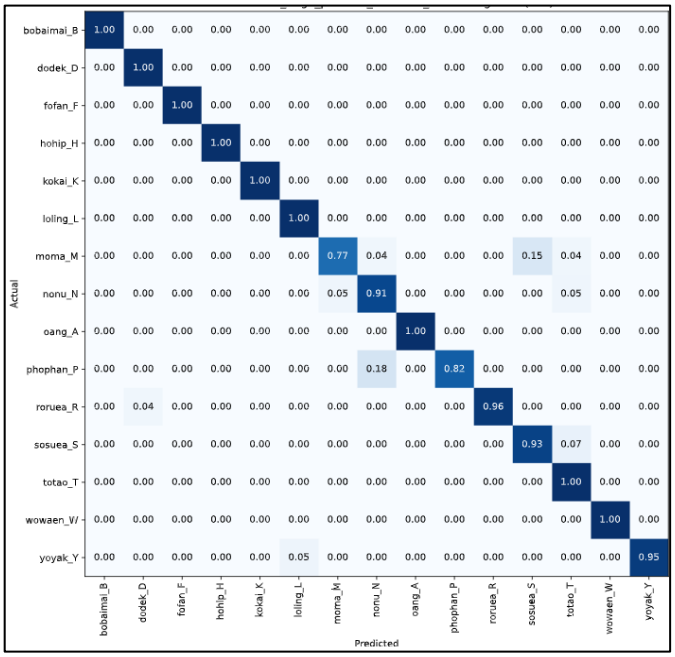


Fig. 7. Confusion matrix of the large EVA-02 at 3rd fold with TTA data augmentation.

D. Pseudo-Labeling

Pseudo labeling [18] is the process of using the trained model to predict labels for unlabeled data. A model has trained with the dataset containing labels and that model is used to generate pseudo labels for the unlabeled dataset. Finally, both the original labels dataset and pseudo labels dataset are combined to retrained the model. In this work, the best performance model from previous experiment is selected to predict labels for unlabeled data. The unlabeled data is collected from our subjects who were asked to stand in front of a white background. The labels dataset and pseudo labels dataset are used to retrain EVA-02 model. Finally, the new retrained model is used to predict the test dataset with TTA technique. Table VI shown performance model using Pseudo labeling. The large EVA-02 model at 1st fold using Pseudo labeling technique achieved best performance at 96.7% accuracy and 96.6% weighted F1-score. Fig. 8 shows the classification results of all 15 classes as a part of class performance analytics. Class "Mo ma" continues to be improved with accuracy increasing to 78%, while for some classes the accuracy has decreased but is still within the acceptable range of at least 80%. This experiment produced good recognition results, as indicated by 11 classes with 100% accuracy, more than any previous experiment. Based on the experiment, the most effective model will be selected as the recognition model which will be used in the program we will develop to recognize TSL finger-spelling recognition. The best EVA-02 model for user will depend on work specific needs. If users are looking for a model that is fast and efficient, then the EVA-02 Tiny or Small models may be a good choice. If user needs the best possible accuracy, then the EVA-02 Base or Large model is the best option.

TABLE VI. PSEUDO-LABELING MODEL PERFORMANCE EVALUATION

Model	Fold	Pseudo-labeling	
		Testing dataset using TTA	
		Accuracy	Weighted F1
Tiny	1	0.7515	0.7332
	2	0.8090	0.7978
	3	0.7885	0.7713
	4	0.8583	0.8509
	5	0.7947	0.7873
Small	1	0.8871	0.8830
	2	0.8604	0.8508
	3	0.8973	0.8942
	4	0.8480	0.8391
	5	0.9014	0.8992
Base	1	0.9425	0.9418
	2	0.9630	0.9628
	3	0.9405	0.9365
	4	0.9405	0.9372
	5	0.9507	0.9492
Large	1	0.9671	0.9663
	2	0.9651	0.9633
	3	0.9446	0.9395
	4	0.9405	0.9382
	5	0.9425	0.9406

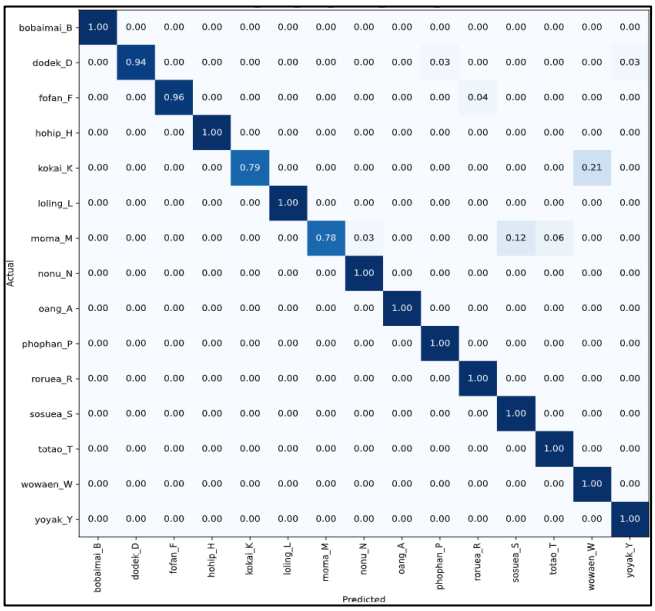
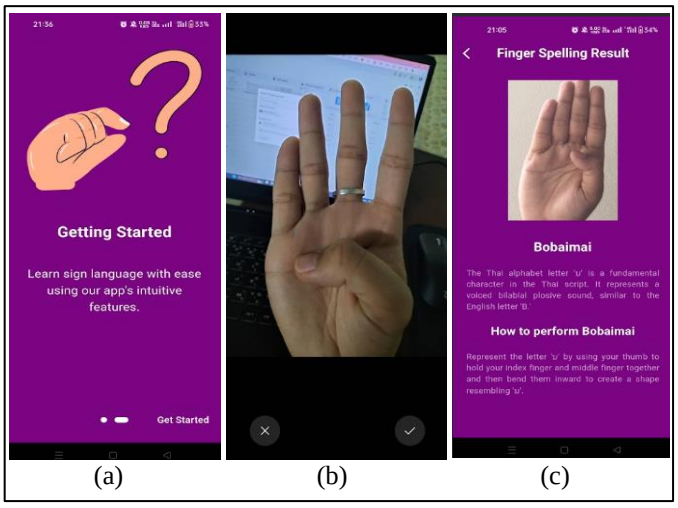


Fig. 8. Confusion matrix of the large EVA-02 at 1st fold with pseudo labeling.

IV. APPLICATION

In this section, we explain our mobile application that users can upload and classify their hand posture images with this application. The application is named "Thai Finger-Spelling Recognition" (see Fig. 9 (a)). We used the Flutter framework to build the front-end part of our mobile application to interact with user, while Python and the Flask framework were employed to develop the backend part to image classification process. The best classification model from experiments is deployed on the server. Users can either capture their own hand posture images or choose from their mobile gallery using the application's main interface (see Fig. 9 (b)). Subsequently, the application uploads the image to the server. The server then prompts the trained model to predict the most likely classes of Thai finger-spelling. The output class retrieves additional details from the database, including example images, class names, alphabet information, and hand posture demonstrations (see Fig. 9 (c)).



(a) (b) (c)

Fig. 9. Thai finger-spelling application: (a) User interface (b) Image capturing (c) Classification information.

V. CONCLUSION

In the study, we proposed training model experiments for Thai sign language finger-spelling recognition. The experiment consisted of vision transformer model training, training with data augmentation, test-time data augmentation, and pseudo-labeling. Our study achieved good performance compared with the experimental phases. In addition, we developed a mobile application. User can upload image data and process from classification to receive useful information related to finger-spelling, such as example image, hand postures, and usage information. For future work, we will collect more data and experiment for several types of Thai finger-spellings such as alphabets, vowels, and tones. Furthermore, we will be testing out other deep learning models like CNN, Capsule Networks, or Generative Adversarial Networks (GANs). The purpose is to compare their recognition performance with the ViT model and reduce model size while maintaining accuracy. This will make it possible to deploy the model in environments with limited resources such as mobile devices or embedded systems.

ACKNOWLEDGMENT

The authors would like to thank College of Computing and Prince of Songkla University, Phuket Campus, Thailand for their support and for providing a working area to test our system.

REFERENCES

[1] Deza and D. Hasan, "MIE 324 final report : sign language recognition," The computer engineering research group (the university of Totonto) [online] March 17 2023 Available: https://www.eecg.utoronto.ca/~jayar/mie324/asl.pdf

[2] R. A. Alawwad, O. Bchir, M. M. B. Ismail, "Arabic sign language recognition using Faster R-CNN," International Journal of Advanced Computer Science and Applications, vol. 12, no. 3, 2021

[3] K. R, H. Bull, L. Momeni, S. Albanie, G. Varol, and A. Zisserman, "Weakly-supervised Fingerspelling Recognition in British Sign Language Videos," 33rd British Machine Vision Conference, 2022.

[4] E. Martinez-Martin and F. Morillas-Espejo, "Deep learning techniques for Spanish sign language interpretation," Computational Intelligence and Neuroscience, vol. 2021, 2021.

[5] V. J. Schmalz, "Real-time Italian Sign Language Recognition with Deep Learning," in CEUR Workshop Proceedings, 2022, vol. 3078, pp. 45–57.

[6] E.-J. Ong, H. Cooper, N. Pugeault, and R. Bowden, "Sign language recognition using sequential pattern trees," in 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2012, pp. 2200–2207.

[7] X. Jiang, B. Hu, S. Chandra Satapathy, S.-H. Wang, and Y.-D. Zhang, "Fingerspelling identification for Chinese sign language via AlexNet-based transfer learning and Adam optimizer," Scientific Programming, vol. 2020, pp. 1–13, 2020.

[8] P. Nakjai, P. Maneerat, and T. Katanyukul, "Thai finger-spelling localization and classification under complex background using a YOLO-based deep learning," in Proceedings of the 11th International Conference on Computer Modeling and Simulation, 2019, pp. 230–233.

[9] S. Lata and O. Surinta, "An end-to-end Thai fingerspelling recognition framework with deep convolutional neural networks," ICIC Express Letters, vol. 16, no. 5, pp. 529–536, 2022.

[10] 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, pp. 131–146, 2019.

[11] J. Sanalohit and T. Katanyukul, "Thai finger-spelling recognition: Investigating MediaPipe Hands potentials," arXiv preprint arXiv:2201.03170, 2022.

[12] Keras, "Keras Documentation: Keras applications," Keras. [Online]. Available: https://keras.io/api/applications. [Accessed: 25-Sep-2023]

[13] A. Dosovitskiy et al., "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," ICLR, 2021.

[14] Y. Fang, Q. Sun, X. Wang, T. Huang, X. Wang, and Y. Cao, "EVA-02: A Visual Representation for Neon Genesis," arXiv preprint arXiv:2303.11331, 2023.

[15] S. Pandian, "K-Fold Cross Validation Technique and its Essential," The Analytics Vidhya [online] September 25 2023 Available: https://www.analyticsvidhya.com/blog/2022/02/k-fold-cross-validation-technique-and-its-essentials/

[16] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and Flexible Image Augmentations," Information, vol. 11, no. 2, 2020, doi: 10.3390/info11020125.

[17] M. Kimura, "Understanding test-time augmentation," in International Conference on Neural Information Processing, 2021, pp. 558–569.

[18] D.-H. Lee and others, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in Workshop on challenges in representation learning, ICML, 2013, vol. 3, no. 2, p. 896.