ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 11, Issue 3, March 2024

Shan Handwritten Alphabet Recognition System Using Fine-Tuned VGG-16 Model

[1] Darli Myint Aung, [2] Dinesh Babu Jayagopi

[1] [2] International Institute of Information Technology Bangalore, Karnataka, India
Corresponding Author Email: [1] darlimyint.aung@iiitb.ac.in, [2] jdinesh@iiitb.ac.in

Abstract— ***The handwritten alphabet recognition system is currently important in any language. In Myanmar, there is less research on the Shan handwritten alphabet recognition system because of the requirement of a standard dataset. This paper proposes a novel Shan handwritten alphabet recognition system using a fine-tuned VGG-16 model. The aim of this paper is to provide an effective and efficient method for classifying Shan handwritten alphabets. This system uses a self-constructed dataset that contains 19 alphabet classes and a total of 19,000 images that were collected by 1000 Shan native participants. This dataset was divided into 80% 10% 10% for training, validation, and testing. In this system, image augmentation, fine-tuning, and L2 regularization are applied for classifying unseen images and reducing the risk of overfitting and underfitting problems. On the fine-tuned VGG-16 model, the different hyper parameters are changed in order to achieve higher accuracy. After selecting the optimal parameter values, the experimental results show that the rate of accuracy is 98.03%. Our proposed model outperforms the VGG16 without using fine-tuning for this dataset. Seventeen of the nineteen alphabets can be correctly for real world image.***

Index Terms— ***Shan handwritten alphabet, Self-constructed dataset, Fine-tuned VGG-16, Image augmentation.***

I. INTRODUCTION

The people who live in Myanmar speak 111 different languages. The most popular ethnic groups are Shan (spoken by 3.2 million), Karen (spoken by 2.6 million), Kachin (spoken by 1.1 million), Chin (spoken by 780,000), Mon (spoken by 750,000) and Rakhine (spoken by 730,000), together with Myanmar (Burmese) (spoken by 33 million) [1].

Among them, the second-largest ethnic group in Myanmar is the Shan people. As per the 2017 Myanmar population report, Shan is one of the eight major ethnic groups in Myanmar, with a population of around 6 million in Shan state. The majority of the Shan people speak Shan language, which is native to Shan State in Burma. Shan speakers can also be found in the Sagaing, Mandalay, Kachin, and Kayah regions. The Shan script is a Brahmic abugida that was developed from the Burmese alphabet to write the Shan language. Recent modifications have made the Shan character more phonetic than other alphabets derived from Burmese. It was not until the 1960s that all vowels and diphthongs were separated in the Shan alphabet, which had a single tone marker and could represent up to fifteen sounds. Because all tones are distinct once the alphabet is altered, the modern alphabet is easier to read. Shan language is spoken by the majority of Shan immigrants in Myanmar. The Shan alphabet has 19 consonants, 10 vowels, and 13 diphthongs. The majority of editors add a dummy consonant that is utilized in words with a vowel onset [2]. The 19 consonants expressed in Fig. 1.

ၵ	ၶ	င
ၸ	သ	ၺ
တ	ထ	ၼ
ပ	ၽ	ၾ
မ	ယ	ရ
လ	ဝ	ႁ
	ဢ	

Fig. 1. Shan consonents

Therefore, this Shan handwriting alphabet recognition system is helpful for numerous applications such as sign language and text-to-speech for natural language processing, etc.

The handwritten alphabet identification system has emerged as one of the most fascinating and difficult areas of image processing and pattern recognition study in recent years. A huge number of publicly accessible datasets in many different languages are available for handwritten letter recognition [3][4][5][6].

The rest of the paper is organized as follows: Section 2 present related work, section 3 gives the proposed model and section 4 describes experimental results and discussion. Lastly, the paper is concluded with future works in section 5.

II. RELATED WORK

In order to demonstrate how different deep learning techniques have been applied to recognize handwritten alphabets, a few significant previous research publications are presented in this section.

The authors [7] designed the Devanagari Handwritten characters classification using fine-tuned deep learning convolutional neural network. The authors created a new dataset for Devanagari handwritten characters. This dataset contains a total of 5800 isolated images of 58 unique character classes: 12 vowels, 36 consonants and 10 numerals. They implemented the two-stage VGG-16 deep learning model to recognize those characters using two advanced adaptive gradient methods. They compared the first model with the second fine-tuned model. They also tested their system on four different benchmark datasets of isolated characters. The authors reported that the highest classification accuracy of 96.55% using the fine-tuned VGG-16. Their finding suggested the effectiveness of the proposed approach.

In [8], the authors designed a model of Arabic Handwriting Recognition System using deep Convolutional neural networks transfer learning in order to categorize Hijja, Arabic and Arabic handwriting dataset. In their study, the authors used not only the data augmentation and dropout for reducing the overfitting problem but also some of the CNN architectures (ResNet, DenseNet, VGG16, VGG19, InceptionV3, and MobileNet) for creating Arabic writing recognition models. The authors pointed out the best model of VGG16 with Adam Optimizer to classify Arabic Handwritten Character and Hijja dataset.

The authors [9] proposed a model for Chinese handwritten character recognition using the combination of a convolutional neural network (CNN) and medium filtering. This paper mainly emphasizes the filtering method in order to smooth and reduce noise. The dataset of CASIA-HWDB1, which consists of 3755 classes of single Chinese words written by 300 participants, was used in the paper. According to their experimental results, an accuracy rate of 90.91% was achieved after training with 5000 epochs. In this paper, they tested their system in real time using five samples of characters. There were some issues: their training time took a long time, and they used only a small number of samples.

The authors [10] suggested a model that classify Kurdish alphabets using deep learning methods. A solution for offline Kurdish handwriting recognition is not yet available. The forty thousand images of all 34 Kurdish characters are included in the self-constructed dataset that the authors developed. They focused on only isolated characters. Their system compared similar models of handwriting recognition systems for other languages. The rate of testing and training accuracy is 96% and 97%, respectively.

A convolutional neural network (CNN)-based system for handwritten character identification in Tamil was demonstrated by the authors [11]. Out of 247 Tamil characters, they selected 121 to be important words. They had 100 samples of each of the 156 Tamil characters in their database. Some alphabets are gathered from the Kaggle platform. The authors used a total of 40,000 images. They utilized image resizing, image segmentation, and image normalization as data preprocessing techniques. The features are compared to the dataset's features following preprocessing. The Google Cloud Translation API was used to generate both the sound and the equivalent translation of the characters. Their system achieved the accuracy of 95%.

The authors [12] studied the deep learning models for recognizing Dogra Handwriting Text recognition. They created the own dataset of Dogra Character and this dataset consists of 39,000 characters. The authors implemented with various classification approaches such as Support Vector Machine, Convolution Neural Network, Random Forest, K Nearest Neighbors and Pre-trained VGG16 CNN. The authors demonstrated that RMSprop optimizer-tuned, pre-trained VGG16 performs better than other methods.

In [13], the authors introduced a transfer learning method for Devanagari alphabet recognition using a variety of pre-trained networks, including VGG 11, VGG-16, VGG-19, AlexNet, DenseNet 121, DenseNet 201 and Inception V3. They used different regularization techniques and the rate of accuracy is 99% in Inception V3 model. Since AlexNet performed better in terms of execution time, transfer learning is a good optimization technique for improving system efficiency. The authors [14] also discussed various of the deep convolutional neural networks (DCNNs) such as VGG Net, FractalNet, RestNet, DenseNet for recognition of handwritten Bangla character.

The authors [15] provided the offline handwritten Gurumukhi characters for improving the recognition results. In addition to hybrid features and adaptive boosting, a 96.2% accuracy rate was attained by applying the three classifiers—decision tree, random forest, and CNN—on 14,000 pre-segmented samples of Gurumukhi characters.

Previous work indicates that sufficient progress has been made on several language alphabets to identify handwritten alphabets using various datasets. Unfortunately, there is no recognition system for Shan State of Myanmar alphabets. Therefore, this paper contributes the following:

- A publicly available dataset, the Shan Handwritten Alphabet dataset, has been created [16] and
- Using the own dataset as the initial implementation for Shan handwritten alphabet, the fine-tuned VGG-16 model is utilized to recognize Shan handwritten alphabet in order to obtain improved accuracy.

III. PROPOSED MODEL

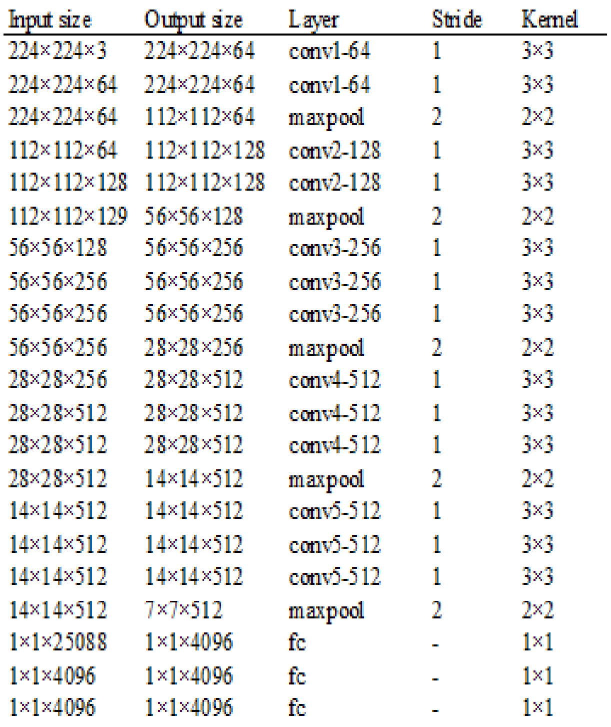
The pretrained deep learning model of fine-tuned VGG-16 is the main focus of this paper. The VGG (Visual geometry group) architecture is the first deep CNN that yields the most promising findings in the field of deep learning. The primary purpose of VGG Net is to extract baseline features from the input images. Additionally, it is not very deep to train with any features dropped at the network's end. It performs perfectly with the limited dataset we have generated. The VGG-16 architecture consists of 13 convolutional layers and 3 fully connected layers. The three primary layers of the

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 11, Issue 3, March 2024

VGG16 architecture are the convolutional layer, pooling layer and fully connected layer. Firstly, the convolution layer is its most important layer. Small learnable filters are used to extract different features from the input image. Secondly, the pooling layer is to decrease the training features. The five maximum pooling layers (2×2 kernel) perform the pooling process. Lastly, the fully connected layer acts primarily as a classification layer.



Input size	Output size	Layer	Stride	Kernel
224×224×3	224×224×64	conv1-64	1	3×3
224×224×64	224×224×64	conv1-64	1	3×3
224×224×64	112×112×64	maxpool	2	2×2
112×112×64	112×112×128	conv2-128	1	3×3
112×112×128	112×112×128	conv2-128	1	3×3
112×112×129	56×56×128	maxpool	2	2×2
56×56×128	56×56×256	conv3-256	1	3×3
56×56×256	56×56×256	conv3-256	1	3×3
56×56×256	56×56×256	conv3-256	1	3×3
56×56×256	28×28×256	maxpool	2	2×2
28×28×256	28×28×512	conv4-512	1	3×3
28×28×512	28×28×512	conv4-512	1	3×3
28×28×512	28×28×512	conv4-512	1	3×3
28×28×512	14×14×512	maxpool	2	2×2
14×14×512	14×14×512	conv5-512	1	3×3
14×14×512	14×14×512	conv5-512	1	3×3
14×14×512	14×14×512	conv5-512	1	3×3
14×14×512	7×7×512	maxpool	2	2×2
1×1×25088	1×1×4096	fc	-	1×1
1×1×4096	1×1×4096	fc	-	1×1
1×1×4096	1×1×4096	fc	-	1×1

Fig. 2. The detailed description of VGG-16 Model

The detail description of VGG-16 model is shown in figure 2. The input dimension for the first convolutional layer is 224 × 224 × 3, since the architecture's input layer by default needs that the image be 224 × 224 × 3. The first block has a 2 × 2 Max-pooling layer of stride 2 × 2, followed by two convolutional layers with 64 channels of 3 × 3 kernel size and the same padding. In the same way as the first block, the second block has two 128-channel convolution layers with a kernel size of 3 × 3, followed by a Max-pooling layer with a stride of 2 × 2. A Max-pooling layer comes after three convolutional layers in the last three blocks. Every convolutional layer in blocks 3, 4, and 5 has the same kernel size of 3×3, but its channel sizes are 256, 512, and 512, respectively. In every Max-pooling layer, the original input image is reduced in size to half its original size. The output feature map from the final Max-pooling layer is 7 × 7 × 512 in size, following the stack of convolutional and Max-pooling layers. A 1 × 25,088 feature vector has been created by adding a flatten layer. Additionally, a dense layer has been added and the classification vector that was obtained through FC is normalized at the end by a Softmax activation function.

An optimizer is a key component in deep learning that adjusts a neural network's parameters during training. Its main aim is to improve performance by reducing the error or loss function of the model. Different optimization techniques, also referred to as optimizers, use different approaches to effectively converge towards the optimal parameter values for better predictions. Adaptive Moment Estimation (Adam) optimizer is used in this paper. Equation (1) is the formula of Adam.

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\left[\frac{\partial L}{\partial w_t}\right] v_t = \beta_2 v_{t-1} + (1-\beta_2)\left[\frac{\partial L}{\partial w_t}\right]^2 \quad (1)$$

whereas

m_t = first moment vector at time step t

v_t = second moment vector at time step t

β_1 = the exponential decay rate for the first moment estimates

β_2 = the exponential decay rate for the second moment estimates

In order to “fine-tune” the trainable layers alongside the fully connected layer, some of the pre-trained layers are loaded as "trainable" and image data is sent through them. The last convolutional and pooling layers in this system are unfrozen to enable training. Lastly, the predicted result is produced by the activation function “Softmax,” which also determines the probability for each of the classes.

Figure 3 shows that the proposed model of fine-tuned VGG-16 is used in this paper. ImageNet's pre-trained weight is used to perform the VGG-16 for the fine-tuning process. Since we only have a small number of Shan handwritten alphabet images for training, it helps to solve the over-fitting issue. In training phase, the Shan handwritten alphabet dataset enter as input images. Some pre-processing methods are used for this system. After pre-processing, the augment images work with fine tuning. After training our dataset, the trained model of Shan Handwritten alphabet can be generated. In testing phase, the data augmentation is considered as Test-Time Augmentation (TTA) for improving the accuracy in particular applications [17].

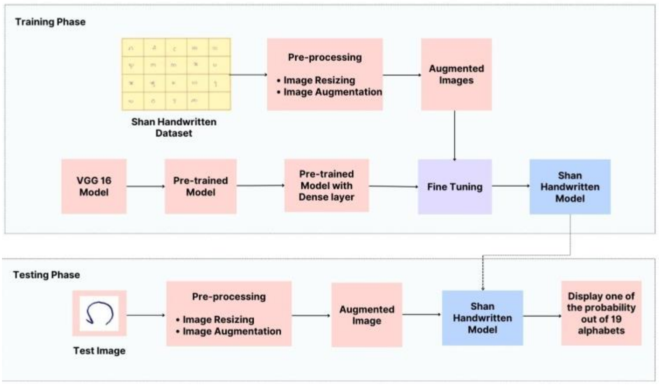


Fig. 3. Proposed Block Diagram of the Fine-Tuned VGG-16 model for Shan Handwritten Alphabet Recognition

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 11, Issue 3, March 2024

IV. EXPERIMENTAL RESULTS AND DISCUSSION

A. Dataset Description

The self-constructed dataset [16] is used in this study. This dataset consists of a total of 19,000 images from 19 different classes. Each class has 1,000 images that consist of labelled images. Figure 4 displays the samples of different writing styles of the Shan handwritten alphabet images.

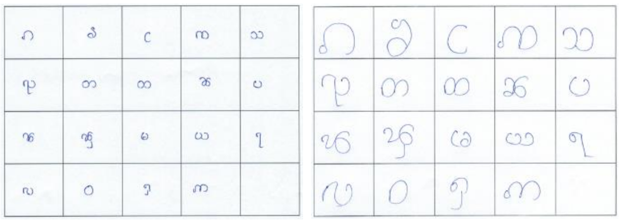


Fig. 4. Samples different writing styles of two participants

B. Pre-Processing

Firstly, the images of Shan handwritten alphabet that were gathered from standard fully labelled dataset are resized to a fixed pixel 224 × 224 size. To improve the learning model's generalizability because of the small number of training samples, data augmentation is performed on own dataset.

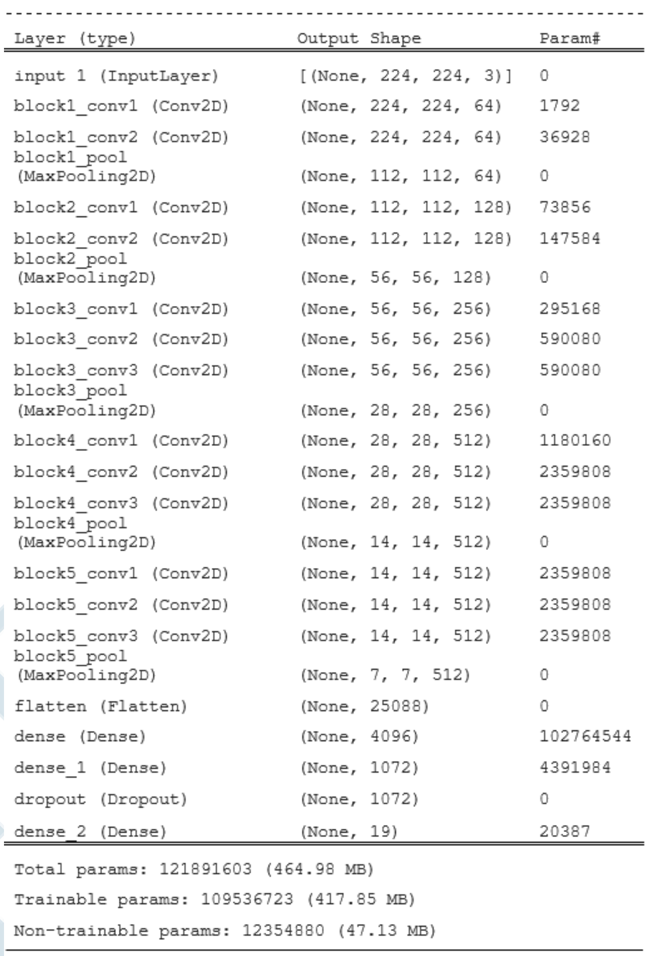
Data augmentation is the technique of creating new data sample from preexisting data samples in order to artificially increase the amount of data. The geometric transformations (rotation, flipping, shifting, shear, zoom and preprocessing function) are used in this paper. Table 1 describes the parameters of data augmentation. The model of learning capability and generalization is enhanced by data augmentation. Additionally, using information exclusively for training and raising the amount of training data is to prevent overfitting problems [18].

Table. I. Parameters for Shan handwritten alphabet images

Geometric Transformation	Parameters
Rotation	25
Horizontal_flip & Vertical_flip	True
Height_shift_range & Width_shift_range	True
Shear_range	0.2
Zoom_range	0.1

C. Implementation

The dataset is split into training, validation, and testing datasets in order to implement the Shan handwritten alphabet recognition system using Google Colab. The 80 random samples are for training each class, and the 10 random samples are for validation and testing, respectively. The pre-trained ImageNet weight is loaded and trained from the first layer with handwritten alphabet images. The model summary of proposed fine-tuned VGG-16 is shown as in figure 5.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param#
=================================================================
input 1 (InputLayer)         [(None, 224, 224, 3)]     0
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
block1_pool
 (MaxPooling2D)              (None, 112, 112, 64)      0
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
block2_pool
 (MaxPooling2D)              (None, 56, 56, 128)       0
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
block3_pool
 (MaxPooling2D)              (None, 28, 28, 256)       0
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
block4_pool
 (MaxPooling2D)              (None, 14, 14, 512)       0
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
block5_pool
 (MaxPooling2D)              (None, 7, 7, 512)         0
flatten (Flatten)            (None, 25088)             0
dense (Dense)                (None, 4096)              102764544
dense_1 (Dense)              (None, 1072)              4391984
dropout (Dropout)            (None, 1072)              0
dense_2 (Dense)              (None, 19)                20387
=================================================================
Total params: 121891603 (464.98 MB)
Trainable params: 109536723 (417.85 MB)
Non-trainable params: 12354880 (47.13 MB)
_________________________________________________________________
```

Fig. 5. Model summary of proposed fine-tuned VGG-1

This Shan handwritten alphabet recognition system is trained with fine-tuned VGG-16 model and the last two layer are trained on our dataset. Since the max value is passed to the following layer in this model, the trainable parameter in the max-pooling layer is zero. The model's training parameters are presented below.

(1) optimizer: Adam
(2) batch size: 224
(3) learning rate: 0.0001
(4) Regularization l2: 0.01
(5) no. of epochs: 50
(6) dropout: 0.5

The accuracy and loss of fine-tuned VGG-16 model is shown in figure 6. It illustrates a smooth training procedure where accuracy rises and loss eventually drops. Furthermore, it is evident that the models do not overfit since the accuracy of the training and validation phases generally do not differ significantly from one another.

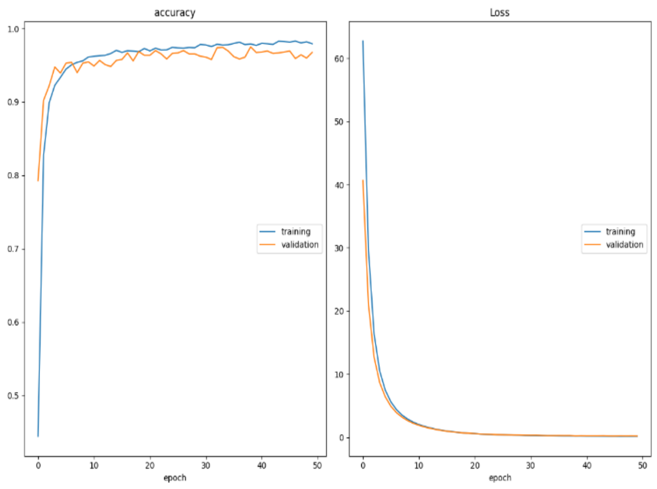


Fig. 6. Accuracy and loss of fine-tuned VGG-16 model on own dataset

There are two measures, such as precision and recall, that are used to evaluate a recognition system's effectiveness. Equation 2 provides the precision (P), which is defined as the proportion of correctly classified images over all classified images.

$$P = \frac{TP}{TP+FP} \tag{2}$$

whereas TP = total positive of images
FP = false positive of images

Equation 3 shows that recall (R) is calculated the number of true positives (TP) images divided by the sum of the number of true positives images and the number of false negatives images (FP).

$$R = \frac{TP}{TP+FN} \tag{3}$$

Equation 4 shows that the average of precision and recall is measured by the F1-score.

$$F1 = 2 \times \frac{P \times R}{P+R} \tag{4}$$

Table. II. Parameters for Shan handwritten alphabet images

Alphabet	Precision	Recall	F1-score
1: ka	1.00	0.97	0.98
2: kha	0.96	0.99	0.98
3: nga	0.96	0.99	0.98
4: tsa	0.96	0.98	0.97
5: sa	0.99	0.99	0.99
6: nya	0.99	1.00	1.00
7: ta	1.00	0.99	0.99
8: tha	0.99	1.00	1.00
9: na	0.99	1.00	1.00
10: pa	0.98	0.95	0.96
11: pha	0.89	0.97	0.93
12: fa	1.00	1.00	1.00
13: ma	1.00	0.98	0.99
14: ya	0.95	0.83	0.89
15: ra	1.00	0.98	0.99
16: la	1.00	0.96	0.98
17: wa	0.90	0.98	0.94
18: ha	0.99	0.96	0.97
19: a	0.98	1.00	0.99
Accuracy (Train)			**0.98**
Accuracy (test)			**0.97**
Macro avg	0.98	0.97	0.97
Weighted avg	0.98	0.97	0. 97

The F1-score, recall, and precision values are provided in Table 2 according to each class.

Moreover, the confusion matrix is used to determine the distribution of predicted images in different classes (see figure 7). In this matrix, this alphabet (ယ) is less recognizable than others. The two alphabets are different, as the alphabet was eleven times mistaken with alphabet (ထ). Even humans are capable of experiencing this. The alphabet (ထ) is misclassified with this alphabet (ထ) as six times mistakes. And then it (ထ) is also misclassified with (ဆ).

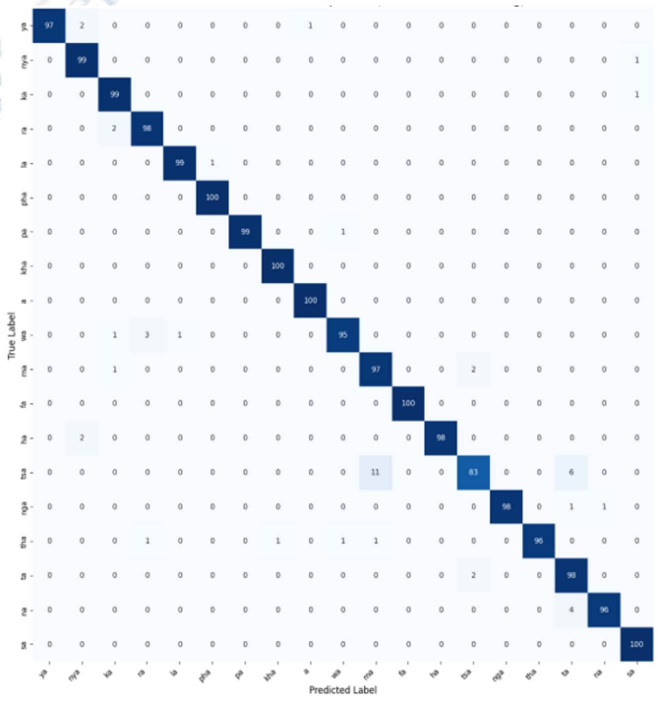


Fig. 7. Confusion matrix of fine-tuned VGG-16 model on Shan handwritten alphabet dataset

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 11, Issue 3, March 2024

Table. III. Comparison of fine-tuned and without fine-tuned VGG-16 Model

Image Size	Fine Tune Parameter	Data Augmentation	Epoch	Batch Size	Dropout	L2	Training Accuracy	Validation Accuracy	Testing Accuracy
224× 224	0	Yes	50	224	0.5	0.01	90.73%	88.97%	92.47%
224× 224	2	Yes	50	224	0.5	0.01	98.03%	97.10%	97.47%

As we can see in table 3, the proposed fine-tuned VGG-16 model is compared with VGG-16 model (without fine-tuning). In terms of training accuracy, the VGG-16 model obtained 90.73%, but the proposed model achieved 98.03% using 50 epochs. For validation, the accuracy of the proposed model is higher than the VGG-16 model without fine-tuning. The proposed model achieved good accuracy after applying fine-tuning. This study demonstrates that our proposed model outperforms than VGG-16 model even if the same parameters are used.

The real-world images are tested in order to improve the performance of this model. Nineteen alphabets were written in real-time to serve as test samples for this test, which is based on a model that was trained, as illustrated in figure 8.

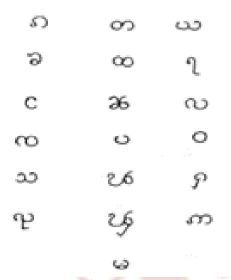


Fig. 8. Real-world images of Shan handwritten alphabet

As can be seen in figure 9, the results indicate that seventeen samples were correctly identified, despite two samples having the wrong classification. The reason is that these two alphabets are similar shapes.

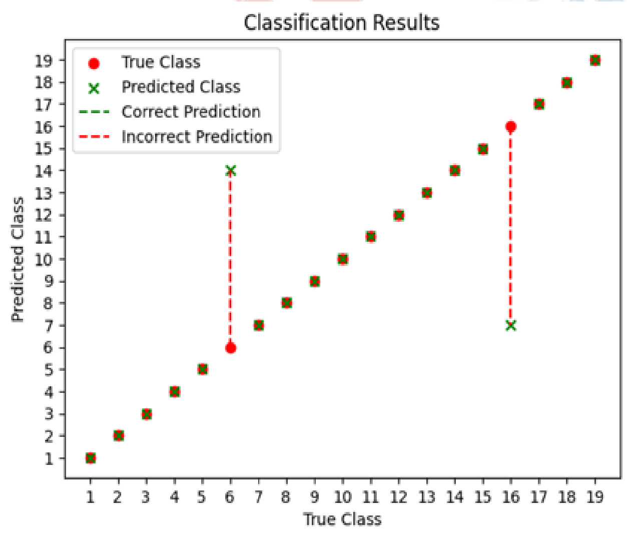


Fig. 9. Real-world images of Shan handwritten alphabet

V. CONCLUSION

Shan handwritten alphabet is difficult and complicated because of each person’s unique writing style. There is no fully standard label dataset for the Shan handwritten alphabet. Therefore, we created a labelled dataset, and there is no published research for this handwritten alphabet recognition system using machine learning and deep learning methods. In this study, the proposed fine-tuned VGG-16 model has been developed to classify the Shan handwritten alphabets. The self-created dataset consists of a total of 19,000 images from 19 classes that contain 1,000 images each. The highest accuracy of this system is obtained at 98.03% for training, 97.10% for validation, and 97.47% for testing. Some data augmentation and regularization techniques are implemented in this system. There is still a limitation in that using a Canon MF240 Series scanner leads to good accuracy because our dataset was created by this scanner. In the future, the proposed model will train on a large dataset and evaluate more architectures for Shan handwritten alphabets.

REFERENCES

[1] Wikipedia Contributors, “Languages of Myanmar”. Wikipedia.com.https://en.wikipedia.org/w/index.php?title=Languages_of_Myanmar&oldid=1198447602 (accessed January. 24, 2024).

[2] Ager and Simon, “Shan alphabet, pronunciation and language,” https://www.omniglot.com/writing/shan.htm. (retrieved Dec. 28, 2018).

[3] K. Manjusha, M. Anand Kumar and K. P. Soman, “On developing handwritten character image database for Malayalam language script,” Engineering Science and Technology, an International Journal, vol. 22, Issue 2, pp. 637-645, February 2019.

[4] C. -L. Liu, F. Yin, D. -H. Wang and Q. -F. Wang, “CASIA online and offline chinese handwriting databases,” International Conference on Document Analysis and Recognition, Beijing, China, 2011.

[5] Ujjwal Bhattacharya. “ISI indian script databases”. Isical.ac.in. http://www.isical.ac.in/ujjwal/download/database.html.

[6] Data Science and Machine Learning Club. “Hand-Written Burmese characters and digits”. kaggle.com.

[7] S. P. Deore and A. Pravin, “Devanagari handwritten character recognition using fine-tuned deep Convolutional Neural Network on trival dataset”, Indian Academy of Sciences, September 2020, https://doi.org/10.1007/s12046-020-01484-1.

[8] S. Masruroh, M. F. Syahid, F. Munthaha, A. T. Muharram, and R. A. Putri, "Deep Convolutional Neural Networks Transfer Learning comparison on Arabic handwriting recognition system," International Journal on Informatics Visualization, JOIV: Int.J. Inform. Visualization, 7(2) – June 2023 330-337.

[9] Y. Zhuang et al., "A handwritten Chinese character recognition based on Convolutional Neural Network and Median Filtering," MEMAT 2021, Journal of Physics: Conference Series, doi:10.1088/1742-6596/1820/1/012162

[10] R. M. Ahmed et al., "Kurdish handwritten character recognition using Deep Learning Techniques," Gene Expression Patterns, vol. 46, pages:119278, DOI: https://doi.org/10.1016/j.gep.2022.119278from ScienceDirect.

[11] P. Gnanasivam, G. Bharath, V. Karthikeyan and V. Dhivya, "Handwritten Tamil character recognition using Convolutional Neural Network," Sixth International Conference on Wireless Communications, Signal Processing and Network (WiSPNET), 2021.

[12] Jagdish Kumar and Apash Roy, "Dogra handwritten text recognition using machine and deep learning models," ISSN 2063-5346, Eur. Chem. Bull. 2023,12 (Special Issue 1), 985-996.

[13] N. Aneja and S. Aneja, "Transfer learning using CNN for handwritten Devanagari character recognition," Proceeding of the 1st International Conference on Advances in Information Technology. Chikmagalur, India, pp. 293–296.

[14] Z. Alom, P. Sidike, M. Hasan, T. M. Taha, and V. K. Asari, "Handwritten Bangla character recognition using the state-of-the-art deep convolutional neural networks," Comput. Intell. Neurosci. 2018: 1–13.

[15] M. Kumar, M. K. Jindal, R. K. Sharma, S. R. Jindal and H. Singh, "Improved recognition results of offline handwritten Gurumukhi characters using hybrid features and adaptive boosting," Soft Computing 25, 11589–11601, 2021, doi: 10.1007/s00500-021-06060-1.

[16] Darli Myint Aung and G R Sinha, "Creation of Dataset for Shan Handwritten Alphabet Recognition System of Myanmar", 2nd International Conference on Intelligent Data Communication Technologies and Internet of Things, IDCIoT-2024.

[17] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," ICLR, arXiv:1409.1556v6 [cs.CV] 10 Apr 2015.

[18] Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," arXivpreprint arXiv:1712.04621, 2017.