Improving Recognition of Thai Handwritten Characters with Deep Convolutional Neural Networks

Sarayut Gonwirat
PhD Student, Multi-agent Intelligent Simulation Laboratory
Department of Information Technology
Faculty of Informatics, Mahasarakham University
Maha Sarakham, Thailand
61011262003@msu.ac.th

Olarik Surinta
Multi-agent Intelligent Simulation Laboratory
Department of Information Technology
Faculty of Informatics, Mahasarakham University
Maha Sarakham, Thailand
Olarik.s@msu.ac.th

ABSTRACT

For handwritten character recognition, a common problem is that each writer has unique handwriting for each character (e.g. stroke, head, loop, and curl). The similarities of handwritten characters in each language is also a problem. These similarities have led to recognition mistakes. This research compared deep Convolutional Neural Networks (CNNs) which were used for handwriting recognition in the Thai language. CNNs were tested with the THI-C68 dataset. This research also compared two training methods, Train from scratch and Transfer learning, by using VGGNet-19 and Inception-ResNet-v2 architectures. The results showed that VGGNet-19 architecture with transfer learning can reduce learning time. Moreover, it also increased recognition efficiency up to 99.20% when tested with 10-fold cross-validation.

CCS Concepts

• Applied computing → Optical character recognition
• Computing methodologies → Neural networks.

Keywords

Handwritten Character Recognition; Convolutional Neural Network; VGGNet; Inception-ResNet; Transfer Learning.

1. INTRODUCTION

Character recognition is fundamental to research that can lead to document analysis, text transcription, or development of automatic reading systems [12]. The recognition method can be beneficial in many fields, e.g. historical document recognition systems, text image retrieval, signature verification, and traffic-sign recognition.

In general, the data used in recognition research about Handwritten Character Recognition (HCR) includes digit, vowel, consonant, and special characters which depend on the writing style of each country [8, 11, 19]. The widespread traditional method is the feature extraction method, including Histogram of Oriented Gradients (HOG), Scale-Invariant Feature Transform (SIFT) [19], and Local Binary Pattern (LBP) [7]. Subsequently, the extracted data are made as an input for various types of machine learning, including K-Nearest Neighbors (KNN) [19], Support Vector Machine (SVM) [5, 19], Multi-layer Perceptron (MLP) etc.

The Convolutional Neural Network (CNN) method [10], which is a deep learning algorithm for fixing the problems in HCR [8, 11], has higher recognition efficiency than traditional machine learning. The differences is that the convolution process in CNN can calculate and find special features automatically which makes CNN Architecture have more layers; for example, VGGNet [18] consists of 16 and 19 layers, ResNet [2] consists of 50, 101 and 152 layers. This directly affects the amount of parameter in calculation. Some research has developed architecture for reducing the number of parameters, for example the Squeeze and Excitation Module [3] and Global Average Pooling Layer (GAP) [16, 20]. The regularization can also be used as an adjustment for weight [21], dropout, and batch normalization [6] in order to increase the efficiency of deep CNN architectures and decrease the data overfitting problem. Furthermore, the data augmentation method is a method for increasing the amount of information used in learning of network and transfer learning. The learning process uses weight values from the model that have previously been learned, then the researcher improved the weight values. The new weight values will be consistent with new information resulting in reduction of learning time and increased network efficiency.

The challenge of handwriting character recognition is the writing style of each person, e.g. emphasizing weight while writing, curve, head of alphabet, and differences in tail-line drawing (stroke, head, loop, and curl). Some characters are similar to other characters. The writing style of the same person at different times is also unstable. Figure 1 shows some characters which share some similarities. In Figure 1(a) the characters have some similar structure, but there are differences at the head of the letter and traits of the tail lines. For Figure 1(b) there are zigzag at the head of the letter. If the writer writes it quickly, the wavy line might not be clear. Then, it will be considered as another character.

Feature Extraction is a part that makes high accuracy rate for character recognition. Studies of Thai handwritten character recognition [5, 19] have used various methods to find unique characteristics. Surinta et al. [19] used two local descriptor methods; Scale-Invariant Feature Transform Descriptor (siftD) and Histogram of Oriented Gradients (HOG). The feature vectors from both descriptor methods were sent to a classifier, including K-Nearest Neighbors (KNN) and Support Vector Machine (SVM) by using Radial Basis Function (RBF) Kernel. The experimental

ICISS 2020, March 19–22, 2020, Cambridge, United Kingdom
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-7725-6/20/03…$15.00
https://doi.org/10.1145/3388176.3388181

results show that siftD with SVM was the most effective method at 94.34% accuracy.

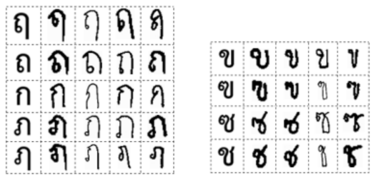

(a) (b)

Figure 1. Examples of similar character groups. (a) Characters with different tail traces and (b) characters with different indentation at head positions.

Inkeaw et al. [5] have developed a method for finding special features called Gradient Features of Discriminative Regions (GFoDRs), which use HOG to calculate the gradient values. This method was called HOGFoDRs. The special features were sent to the SVM classifier for character classification. The HOGfoDRs were designed for discrimination of similar characters. The accuracy rate of this method was 98.76%.

Contribution: The objective of this research is to perform the efficiency of deep CNN on character recognition of Thai handwritten character. The architecture of CNN in this research is composed of VGGNet [18] and Inception-ResNet [20], which do not need to calculate special characteristics because convolutional layers in deep CNN calculates lower-level feature. The test compares both learning style, including scratch learning and transfer learning in order to find the most suitable model for Thai handwritten analysis. We did not use data augmentation to increase training data for learning the deep CNN in both architectures due to compare the experimental results with the siftD+SVM [19] and HOGfoDRs methods [5]. The experiment found VGGNet Architecture with transfer learning were the most effective in recognition while compare to other methods. Therefore, this method is suitable for solving the problem of character recognition in Thai handwritten.

Paper outline: This paper is organized as follows: in Section 2, the background of convolutional neural networks is explained. Two deep CNN architectures are described in Section 3. Section 4 describes the Thai handwritten character dataset that is used in the experiments. The experimental results of the deep CNN methods and other methods are presented in Section 5. The conclusion and future work are presented in Section 6.

2. BACKGROUNDS

2.1 Convolutional Neural Network

The convolutional neural network (CNN) presented by LeCun [11] for English character recognition. CNN has become popular in image recognition after Krizhevsky et al. [9] presented AlexNet Architecture and won the ImageNet Challenge in 2012. After that, the researchers developed various CNN architectures in different series, e.g. VGGNet, GoogLeNet, ResNet, DenseNet [4], and MobileNet [16]. Each CNN had different architecture and different name, e.g. number of convolutional layers, inception module [6, 20], shortcut connection module [2, 16, 20], and depthwise convolutional filters [16]. The basic structure of CNN architecture describes as follows;

2.1.1 Convolutional Layer

The Convolutional Layer (Conv) is the main layer which is used for calculating feature extraction. The convolution process is to find dots from the input layer (Image) or output of previous convolutional layer as shown in Equation 1. The input layer is required to have feature map (x_p), while p is the hierarchy of the layer in CNN. The CNN has amount of parameters equal to $w_p \times h_p \times d_p$, while w_p is length, h_p is width, d_p is channel. From calculating convolution and filter kernel (K), the result is a feature map (x_{p+1}) which has size equal to $d_k \times d_k \times d_p \times d_{p+1}$ while d_p is the width and length of kernel (K) in the hierarchy p.

$$X_{k,l,n}^{p+1} = \sum_{i,j,m} K_{i,j,m,n}\, x_{k+i-1,l+j-1,m}^{p} \quad (1)$$

Output or feature map from each layer was sent to the activation function in a Rectified Linear Units (ReLU): as shown in Equation 2 [13]. Then, it was sent to batch normalization (BN) process [6]. BN Layer normalizes the input data by scaling all data in order to provide data in the same range. This speeds up the learning and reduces data overfitting. As a result, the dropout configuration can be set to a low level, resulting in reduction of information lost during the dropout.

$$ReLU(x) = \max(0, x) \quad (2)$$

2.1.2 Pooling Layer

A Pooling Layer is a spatial computation layer in the feature map layer which helps reduction of parameter sizes in the architecture by finding of maximum, minimum, and average values.

2.1.3 Fully Connected Layer

A Fully Connected Layer (FC) is a connection of every node from one layer to every node of the next Layer. This is the same process as Multi-Layer Perceptron (MLP) while the output layer of FC layer has the number of nodes equal to the number of categories. Softmax function was used for output calculation (shown in Equation 3).

$$softmax(x) = \frac{\exp(x)}{\sum_i^N \exp(x_i)} \quad (3)$$

2.2 Optimization Method

The processing of the CNN results in the most probability type of recognition, but sometimes the answers do not match with the expectations. It is error value. Therefore, the error value could be minimized by adjusting weight parameters. In this research, the researcher uses Stochastic Gradient Descent (SGD) with momentum [15] for weight parameters adjustment (shown in Equation 4).

$$\theta_{t+1} = \theta_t + v_{t+1} \quad (4)$$

$$v_{t+1} = \mu v_t - \alpha \nabla f(\theta_t) \quad (5)$$

where μ is momentum coefficient, α_t is learning rate, and $\alpha \nabla f(\theta_t)$ is error gradient for weight parameter θ adjustment. Learning rate will be reduced when epoch of the learning increase, show in Equation 6.

$$\alpha_t = \frac{\alpha_0}{1+dt} \quad (6)$$

where α_0 is the initial learning rate, d is learning rate decay.

2.3 CNN based Scratch and Transfer Learning

CNN learning method was divided into 2 processes; comprising learning from scratch and transfer learning. Learning from scratch [14] is a complex process and takes a long time to learn due to the learning beginning with creation of a random weight, by sending batches of images (batch) to learn. The sizes can be small or large depending on the computer used in the learning. Weights are calculated and adjusted in each round depending on the input data. Finally, this process produces a model for prediction.

Transfer learning [14, 17] is applying knowledge from previous domains that have been learned, to solve problems with the same characteristics or maybe a new problem. It is assumed that the parameters from the original model can be used as a starting point to learn new information. It is called the *Pre-trained model* which directly results in faster training and higher effectiveness. This is because of pre-trained model was created from the ImageNet data set, that contains over a million images, in which sample data is organized in up to 1,000 categories. Therefore, if we want to use the Pre-trained Model for further processing with another dataset, the output node of the FC layer must be adjusted until it match the amount of that category.

3. CNN ARCHITECTURES

Since 2012, researchers have developed high effective CNN Architectures with structural adjustment methods, e.g. VGGNet [18]. In addition, the layers were increased up to 16 and 19 layers. GoogLeNet architecture [6, 20] designed Inception module. The module was assigned to use multi-size convolution including, 1x1, 3x3, and 5x5 which is called a filter. The output of each Inception module is put through each filter together (Filter Concatenation). ResNet architecture [2] was designed Residual block which is a shortcut connection that makes training processes able to skip more than one layer. ResNet was designed to have from 18, 34, 50 and 101, to 152 layers. The trend in CNN architectures development is to increase the number of layers, but to decrease the amount of parameters, e.g. Inception-ResNet [20] and DenseNet [4]. In this paper, two CNN architectures were tested. These were VGGNet and Inception-ResNet.

3.1 VGGNet Architecture

In 2014, a research team sent VGGNet [18] to compete in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). The architecture has as many as 19 layers, tiled into stacks, connected with 3 layers of FC. The first 2 layers have 4,096 nodes. The third layer has 1,000 output nodes. The highlight of VGGNet is the use of a convolution filter that is very small, only 3x3 filter when using convolution processing. When we compare with the AlexNet architecture, it can be seen that there are more layers, but it has higher efficiency.

Table 1 shows VGGNet architecture with 16 and 19 layers. VGG16 consists of convolution layer (Conv) with 3x3 (Conv3), 13 filter layers, and 3 FC layers, total 16 layers. The amount of feature maps has increased to 64, 128, 256, 512, 512, and 512 layers consequently. Max Pooling Layers were added between convolutions in order to reduce the dimension of width and length. The VGG19 also consists of 16 convolution layers and 3 FC layers.

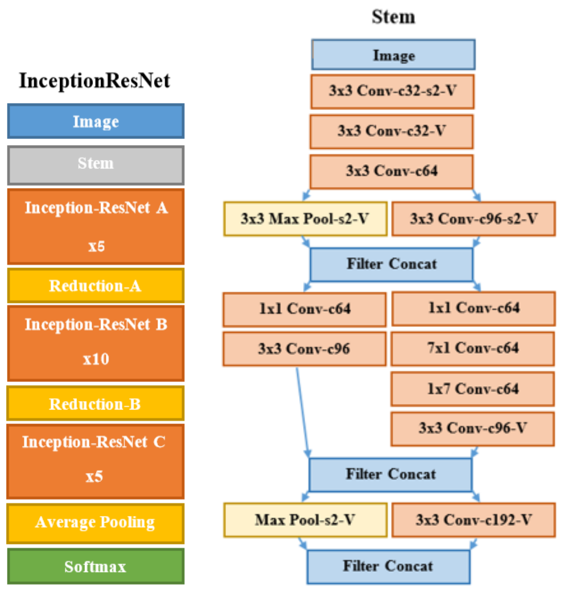
3.2 Inception-ResNet-v2 architecture

Inception-ResNet-v2 [20] was developed using batch normalization for improving the training speed. Only 7% of training steps can increase the effectiveness of the architecture. It uses Factorization to reduce the filter size, resulting in reduction of the overfitting problem, number of parameters were also reduced. The increasing of Residual block between Inception module leads to large number of Inception modules.

The main structure of Inception-ResNet-v2 divides the work function as a block, including Stem, Inception-ResNet, and Reduction blocks as shown in Figure 2(a).

Table 1. Configuration of the VGG16 and VGG19 architectures

VGG16	VGG19
Input	Input
Conv3, c64x2	Conv3, c64x2
Max Pooling	Max Pooling
Conv3, c128x2	Conv3, c128x2
Max Pooling	Max Pooling
Conv3, c256x3	Conv3, c256x4
Max Pooling	Max Pooling
Conv3, c512x3	Conv3, c512x4
Max Pooling	Max Pooling
FC-4096	FC-4096
FC-4096	FC-4096
FC-1000, Softmax	FC-1000, Softmax



(a) (b)

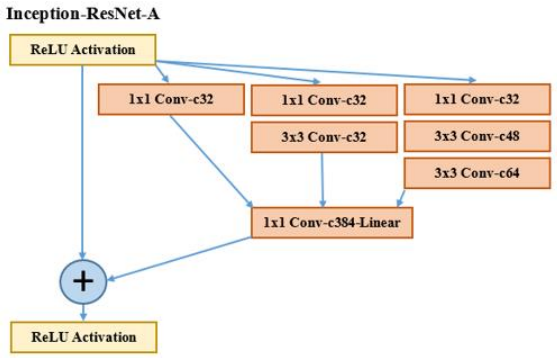
Figure 2. Inception-ResNet-v2 architecture. (a) Core architecture and (b) detail of the Stem block.

3.2.1 Stem Block

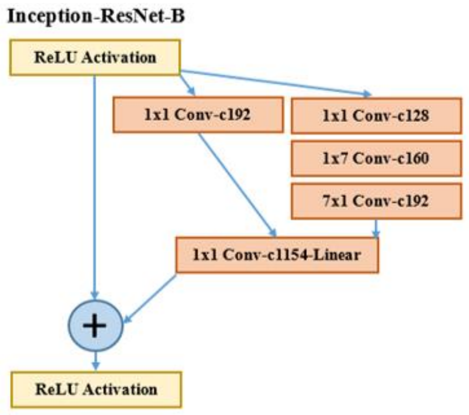
The Stem block is the first layer of architecture. It is a layer before the Inception module. The convolution filter in the Stem block is 3x3, stride values are 2 (s2), therefore the feature map would become smaller, which will directly decrease the parameter values as shown in Figure 2(b)

3.2.2 Inception-ResNet Block

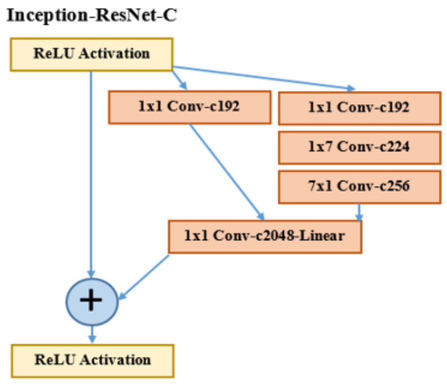
The advantage of Inception module is the combination of ResNet Architecture and Inception layer. That is why it has been called Inception-ResNet block. The Inception-ResNet has 3 blocks, which are called blocks A, B, and C as shown in Figure 3. The gap between Inception-ResNet blocks are separated by Reduction blocks due to parameter reduction.



(a)



(b)



(c)

Figure 3. Architecture details of the Inception-ResNet. block (a) A, (b) B, and (c) C.

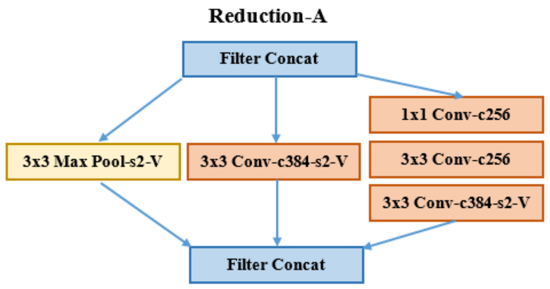
3.2.3 Reduction Block

The purpose of the Reduction block at the gap between Inception-ResNet blocks is to reduce the feature map size. Inception-ResNet architecture has 2 Reduction blocks. These are Reduction block A and B as shown in Figure 4.

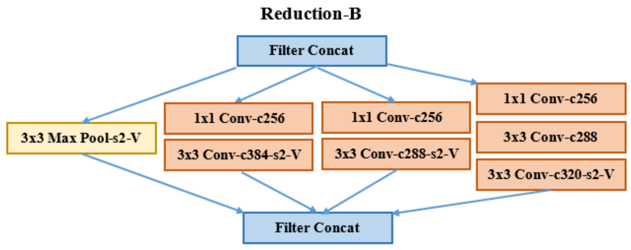
4. THAI HANDWRITTEN CHARACTER DATASET

The Thai handwritten character dataset in this research is ALICE-THI dataset [19], which includes 78. types of Thai characters; consonants, vowels, tones and digits. The dataset contains writing from 150 people, aged 20-23, who were studying in a university. This research used only the THI-68 dataset which eliminated the number. Therefore, the number of characters used in recognition was 68 Characters. The data size was 14,490 characters, including 44 consonants, 17 vowels, 4 tones, and 3 symbols as show in Figure 5.

Surinta et al. [19] used special features siftD take it to learn by SVM algorithm. The accuracy rate was 94.37%. Moreover, Inkeaw et al. [5] used special feature HOGFoDRs with SVM algorithm. The accuracy rate was 98.76%. Due to the similarity of characters, this dataset is challenged for higher effective rate.



(a)



(b)

Figure 4. The reduction block (a) A and (b) B.

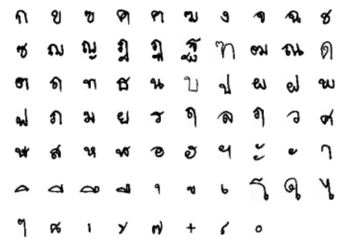


Figure 5. Example of 68 Thai handwritten characters.

5. EXPERIMENTAL RESULTS

This research performed the handwritten character recognition of Thai characters with ALICE-THI dataset by choosing a specific test on THI-C68 dataset which has 14,490 characters in 68 classes. This research used deep CNNs, consists of VGGNet [18] and Inception-ResNet-v2 architectures to compare the performance between both deep CNN architectures. The effectiveness of the methods were compared with siftD-SVM [19] and HOGFoDRs-SVM [5] on computer Intel(R) Core-i5, 7400 CPU @ 3.00GHz, 8GB RA, GPU GeForce GTX 1080Ti, Memory 16GB, Linux Operating system. The experiment divided the data into 2 sets. (Training and Test sets), including 5-fold, and 10-fold cross-validation. The 5-fold data and 10-fold data were set at the following ratios; Train:Valid:Test, 7:1:2 and 8:1:1, respectively.

The CNNs experiment resized all images to 128x128 pixels, which is the smallest input size of Inception-ResNet-v2. Training processes had 100 Epochs, used SGD Learning Method, Learning Rate = 0.001, Decay Rate = 0.0001, and momentum = 0.9. Learning processes were divided into 2 types which were training from scratch and transfer learning. Weight Parameters in transfer learning were derived from previous learning process by ImageNet Dataset [1]. It was called Fine-tuned.

Table 2. Performances of different models on THI-C68 dataset

Methods	Accuracy Rate (%)	
	10-cv	**5-cv**
SiftD-SVM [19]	94.34	-
HOGFoDRs-SVM [5]	-	98.76
VGGNet-Scratch	97.93 ±0.55	96.93 ±0.48
Inception-ResNet-Scratch	98.15 ±0.24	97.79 ±0.29
VGGNet-Transfer	**99.20 ±0.27**	**98.81 ±0.25**
Inception-ResNet-Transfer	98.88 ±0.24	98.61 ±0.14

To ensure comparison equality, data augmentation was not used in [5, 19]. This research also did not use data augmentation due to several studies e.g. [14] , reporting that data augmentation increases efficiency of CNN Architectures.

Table 2 is a comparison of the efficiency and accuracy of Thai handwriting characters in 6 different methods. When we analyze only deep CNN architectures, it shows that VGGNet-Transfer had the highest efficiency in both 5-fold and 10-fold with 99.2% and 98.81% accuracy rate. It was found that VGGNet in the experiments was VGG-19, which has 19 layers. Transfer learning increased the accuracy rate for CNN architecture by 1-2% when compare to the training from scratch method. The VGGNet-Scratch learning process compare to 10-fold cross-validation achieved an accuracy rate of 97.93%, which is 3% higher than siftD-SVM. However, when the researchers tested with 5-fold cross-validation, VGGNet-Transfer (98.81%) achieved an insignificantly higher effective rate than HOGFoDRs-SVM (98.76%).

From these experiments, comparison of VGGNet-19 and Inception-ResNet-v2 found that VGGNet-19 learning by transfer learning has the highest recognition rate. The size of this model is only 160.6 MB comparing with Inception-ResNet-v2 which is 437.5MB. The number of parameters comparison, VGGNet-19 has only 20M parameters which is almost 3 times less. Finally, when comparing the test speed, VGGNet-19 speed was 0.0014 second per image, while Inception-ResNet-v2 speed was 0.0043 second per image. We can conclude that VGGNet-19 speed was up to 3 times faster.

Surprisingly, InceptionResNet-v2 architecture [20] tested with ImageNet dataset achieved 5.7% higher efficiency than VGGNet-19. In contrast, when it was tested with the THI-C68 dataset, which is Thai character, we found that VGGNet with transfer learning achieved a higher accuracy rate. Therefore, in this experiment VGGNet-19 is an appropriate model to solve the problems of "*Thai Handwritten Character Recognition*" due to the smaller size model, less number of parameter, faster speed in the experiment, and highest accuracy rate. The most important is the model that achieved the highest accuracy rate.

6. CONCLUSIONS

This research compares CNN Architectures that are effective in recognizing Thai handwritten characters with a high rate of recognition. The two models are VGGNet-19 and Inception-ResNet-v2 architectures. Both models were evaluated with THI-C68 dataset. In this experiment, the learning method was determined in two types, which are training from scratch and transfer learning. Transfer learning is a way to reduce learning time and increasing the efficiency of recognition. The research has shown that VGGNet-19 architecture with transfer learning has an accuracy rate at 99.20%. In addition, it was higher than Inception-ResNet-v2 architecture. In this regard, VGGNet-19 architecture is a deep learning that has only 19 layers. It has been designed to be stacked together due to make it easier to learn from the network and for increasing the recognition speed.

In future work, researchers will design deep CNN architecture that reduces the number of parameters and reduce learning time. However, the quality must still be equivalent or better performance than with the previous architecture and it will be tested with handwriting characters in other languages such as Bangla, Lanna etc.

7. REFERENCES

[1] Deng, J., Dong, W., Socher, R., Li, L.-J., Kai Li and Li Fei-Fei 2009. ImageNet: A large-scale hierarchical image database. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Jun. 2009), 248–255.

[2] He, K., Zhang, X., Ren, S. and Sun, J. 2016. Deep Residual Learning for Image Recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Dec. 2016), 770–778.

[3] Hu, J., Shen, L. and Sun, G. 2018. Squeeze-and-Excitation Networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Dec. 2018), 7132–7141.

[4] Huang, G., Liu, Z., Maaten, L. van der and Weinberger, K.Q. 2017. Densely Connected Convolutional Networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Nov. 2017), 2261–2269.

[5] Inkeaw, P., Bootkrajang, J., Marukatat, S., Gonçalves, T. and Chaijaruwanich, J. 2019. Recognition of Similar Characters using Gradient Features of Discriminative Regions. *Expert Systems with Applications*. 134, (Nov. 2019), 120–137.

[6] Ioffe, S. and Szegedy, C. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *The 32nd International Conference on International Conference on Machine Learning* (2015), 448–456.

[7] Joseph, F.J.J. and Anantaprayoon, P. 2018. Offline Handwritten Thai Character Recognition Using Single Tier Classifier and Local Features. *The 3rd International Conference on Information Technology (InCIT)* (Oct. 2018), 8–11.

[8] Kim, I.J. and Xie, X. 2014. Handwritten Hangul Recognition using Deep Convolutional Neural Networks. *International Journal on Document Analysis and Recognition*. 18, 1 (2014), 1–13.

[9] Krizhevsky, A., Sutskever, I. and Hinton, G.E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems 25* (2012), 1090–1098.

[10] Lecun, Y., Bengio, Y. and Hinton, G. 2015. Deep learning. *Nature*. 521, 7553 (May 2015), 436–444.

[11] Lecun, Y., Bottou, L., Bengio, Y. and Haffner, P. 1998. Gradient-Based Learning Applied to Document Recognition. *IEEE*. 86, 11 (1998), 2278–2324.

[12] Marinai, S. 2008. Introduction to Document Analysis and Recognition. *Studies in Computational Intelligence*. Springer Verlag. 1–20.

[13] Nair, V. and Hinton, G.E. 2010. Rectified Linear Units Improve Restricted Boltzmann Machines Vinod Nair. *The 27th International Conference on Machine Learning* (2010), 807–814.

[14] Okafor, E., Pawara, P., Karaaba, F., Surinta, O., Codreanu, V., Schomaker, L. and Wiering, M. 2016. Comparative Study between Deep Learning and Bag of Visual Words for Wild-Animal Recognition. *IEEE Symposium Series on Computational Intelligence (SSCI)* (Dec. 2016), 1–8.

[15] Ruder, S. 2016. An Overview of Gradient Descent Optimization Algorithms. (Sep. 2016), 1–14.

[16] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. and Chen, L.C. 2018. MobileNetV2: Inverted Residuals and Linear Bottlenecks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Dec. 2018), 4510–4520.

[17] Sawada, Y. and Kozuka, K. 2016. Whole Layers Transfer Learning of Deep Neural Networks for a Small Scale Dataset. *International Journal of Machine Learning and Computing*. 6, 1 (2016), 27–31.

[18] Simonyan, K. and Zisserman, A. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. *3rd International Conference on Learning Representations (ICLR)*.

[19] Surinta, O., Karaaba, M.F., Schomaker, L.R.B. and Wiering, M.A. 2015. Recognition of handwritten characters using local gradient feature descriptors. *Engineering Applications of Artificial Intelligence*. 45, (Oct. 2015), 405–414.

[20] Szegedy, C., Ioffe, S., Vanhoucke, V. and Alemi, A. 2017. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *The Thirty-First AAAI Conference on Artificial Intelligence* (2017), 4278–4284.

[21] Wang, B. and Klabjan, D. 2017. Regularization for Unsupervised Deep Neural Nets. *The Thirty-First AAAI Conference on Artificial Intelligence* (Aug. 2017), 2681–2687.