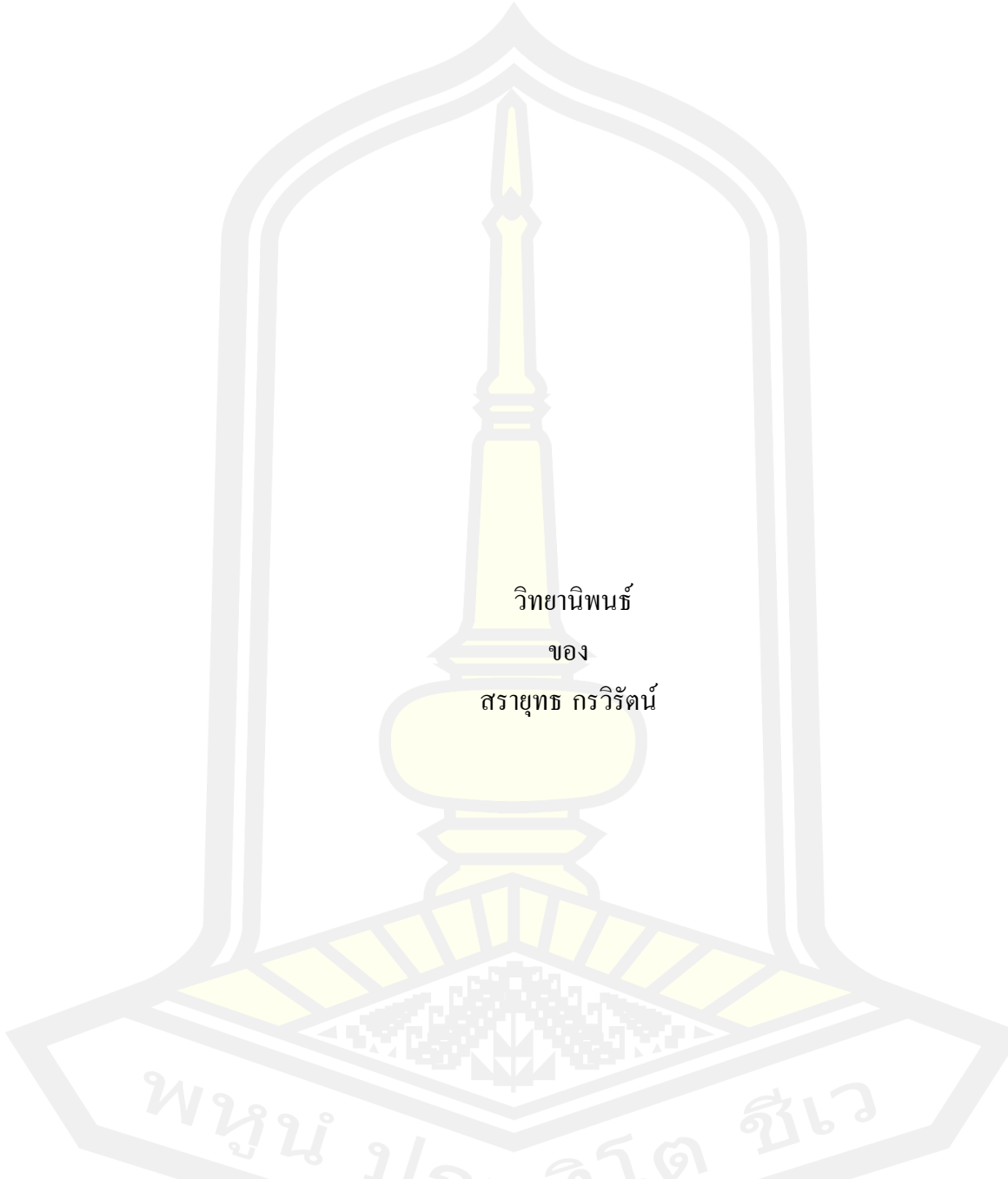
Historical Document Transcription using Deep Learning

Sarayut Gonwirat

A Thesis Submitted in Partial Fulfillment of Requirements for
degree of Doctor of Philosophy in Information Technology
January 2023

การถอดความเอกสารโบราณด้วยการเรียนรู้เชิงลึก



วิทยานิพนธ์

ของ

สรายุทธ กรวิรัตน์

เสนอต่อมหาวิทยาลัยมหาสารคาม เพื่อเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาปรัชญาดุษฎีบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ

มกราคม 2566

Historical Document Transcription using Deep Learning

Sarayut Gonwirat

A Thesis Submitted in Partial Fulfillment of Requirements

for Doctor of Philosophy (Information Technology)

January 2023

The examining committee has unanimously approved this Thesis, submitted by Mr. Sarayut Gonwirat , as a partial fulfillment of the requirements for the Doctor of Philosophy Information Technology at Mahasarakham University

Examining Committee

.....Chairman
(Prof. Rapeepan Pitakaso , Ph.D.)

.....Advisor
(Asst. Prof. Olarik Surinta , Ph.D.)

.....Committee
(Asst. Prof. Rapeeporn Chamchong , Ph.D.)

.....Committee
(Asst. Prof. Chatklaw Jareanpon , Ph.D.)

.....Committee
(Asst. Prof. Phatthanaphong Chompoowises , Ph.D.)

Mahasarakham University has granted approval to accept this Thesis as a partial fulfillment of the requirements for the Doctor of Philosophy Information Technology

.....	.....
(Assoc. Prof. Jantima Polpinij , Ph.D.)	(Assoc. Prof. Krit Chaimoon , Ph.D.)
Dean of The Faculty of Informatics	Dean of Graduate School

TITLE Historical Document Transcription using Deep Learning

AUTHOR Sarayut Gonwirat

ADVISORS Assistant Professor Olarik Surinta , Ph.D.

DEGREE Doctor of Philosophy **MAJOR** Information Technology

UNIVERSITY Mahasarakham University **YEAR** 2023

ABSTRACT

This thesis focuses on handwritten text recognition problems. The research aimed to approach deep learning methods to improve the performance of recognitions in a historical document.

Chapter 1 briefly introduces deep learning for handwritten text recognition systems and uses deep learning techniques for analyzing and recognizing a historical document, including research questions, the objectives of the dissertation and its contributions are described.

In Chapter 2, Two deep convolutional neural networks (CNNs): VGGNet and InceptionResNet, are proposed for handwritten character recognition. The proposed research investigated two learning strategies, including scratch and transfer learning, and compared them with traditional machine learning techniques of local descriptor and support vector machine. The results showed that VGGNet architecture with transfer learning can reduce learning time. Moreover, it also increased the efficiency of recognition.

Chapter 3 presents solutions to problems that can reduce handwritten character recognition performance, such as image degradation, light conditions, low-resolution images, and even the quality of the capture devices. We combine the deblur generative adversarial network architecture (DeblurGAN) with a CNN called DeblurGAN-CNN. The DeblurGAN-CNN could transform the noisy characters into new clean characters and recognize clean characters simultaneously. We have evaluated and compared the experimental results of the proposed DeblurGAN-CNN architectures with the existing methods on four handwritten character datasets: n-THI-C68, n-MNIST, THI-C68, and THCC-67. For the n-THI-C68 dataset.

Chapter 4 proposes the architecture of the CNN and recurrent neural network (RNN), called CRNN architecture, to predict the sequence pattern of the handwritten text images. We propose a novel cyclical data augmentation strategy called CycleAugment, to discover various local minima values and prevent overfitting. Each cycle rapidly decreased the training loss to reach a new local minima.

Chapter 5 comprises two main sections: - 1) answers to the research questions and 2) future work. This chapter briefly explains the proposed approaches and answers three main research questions in handwritten text recognition using deep learning techniques. Two main approaches are planned to be the focus of future work, as follows. We might need to synthesize the handwritten text images and use them as the training set. The GAN is the best choice to study and synthesize the training set. And to enhance deep learning performance, we plan to work on the ensemble CNNs technique and combine the DeblurGAN-CNN architecture as a part of the ensemble CNNs.

Keyword : Handwritten character recognition, Denoising image, Handwritten text recognition, Generative adversarial network, Convolutional neural network, DeblurGAN, convolutional recurent neural network, Cycle agumentation

ACKNOWLEDGEMENTS

This thesis was supported by the scholarships of Kalasin University and was financially supported by the Royal Golden Jubilee Ph.D. Program by the Thailand Research Fund (Grant No. PHD/0210/2561). The authors would like to thank the researchers and give special thanks to Asst. Prof. Dr. Olarik Surinta, my advisor, for his efforts toward completing this research.

Additionally, I would like to thank my family for the best encouragement and support all through my studies, my friends, lab mates, colleagues, and research team for a cherished time spent together in the lab and in social settings.

Sarayut Gonwirat

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

Chapter 1

Introduction

In ancient times, humans wrote a text or documents to keep their information, knowledge, history, and imagination. The text was recorded in materials, such as paper, books, palm leaves, wooden planks, or stones, etc. Nowadays, those historical documents are transformed into digital files. Most of them are archived by scanning as image files. However, it is difficult to retrieve information from images. Thus, many archives need to make index or metadata convenient for users. Due to the growing rate of historical collections, it is more challenging to prepare metadata by humans, and some documents are old languages that need specialists or historians to transcribe. For example, Thailand has some languages not usually used in recent writing, such as Thai Noi in the Northeast and Lanna in the North. To help extract those ancient documents, an automatic text understanding or recognition system is vital for historical documents in archives to reduce man work and ease of use.

To overcome machine understanding in historical documents, the research in this area is about document analysis and recognition, mainly focusing on automatic information extraction such as layout analysis, word localization, text recognition, text transcription, etc. As an illustration of application, Schomaker et al. (2009) developed the Monk system to support researchers in machine learning. This system consists of more than a thousand pages of digital images that are labeled index on pages, lines, words, or characters. The datasets were provided for text recognition, dating classification of manuscripts, and writer identification.

In modern applications, text recognition is important and was applied to industrial automation, robot navigation, and instant translation. Optical character recognition (OCR) is a fundamental machine learning problem in image recognition research in the classical text recognition problem. It can be divided into two main categories of dataset, including printed and handwritten text. Currently, the printed text is solved by machine learning techniques. In contrast, handwritten text recognition has been challenged by different personal styles, strokes, and cursive writing of multiple persons or even a single person. For handwritten text recognition, we have found various research in text documents such as scene text recognition or video subtitles. In addition, handwritten text recognition is proposed to solve in many languages in each of their counties, such as English, Chinese, Arabic, Indian, and Amharic (Sujatha and Bhaskari 2019; Yan and Xu 2020; Abdurahman, Sisay, and Fante 2021; Ameryan and Schomaker 2021; Butt et al. 2021; Singh, Sharma, and Chauhan 2021), and historical documents were usually found as handwritten text.

Handwritten character recognition can achieve high performance if character segmentation is robust. However, handwritten text recognition (HTR) methods mainly focus on word recognition and have become a more prominent research domain. Moreover, the another research fields are focused on the effects of handwritten text

recognition performance to be considered 1) the degradation of historical documents, 2) may be due to a lack of expert staff and the humidity from a storage location, digital transformation, 3) the blur and noisy document images that appeared when using low-quality equipment and taking the picture with a camera without adequate lighting, and 4) the limitation of dataset is an insufficient and uncovered dataset of handwritten character images in the training process. Those effects are still challenging to solve in historical documents.

Due to the rapid development of artificial intelligence in computer vision, deep learning techniques, including convolutional neural networks (CNNs), auto-encoder, generative adversarial networks (GANs), recurrent neural networks (RNNs), and vision transformers, are proposed to improve image recognition, image restoration, natural style transfer, or object detections, etc. For handwritten text recognition, LeCun, et al. (1998) proposed the first CNN to digital handwritten character recognition on the well-known dataset, namly MNIST. After that, CNN is dominant in general image classification. In sequence learning, RNNs are applied in speech recognition and various applications in natural language processing (NLP). Further, word image recognition has been solved by RNNs or a combined architecture of RNN and CNN, called a convolutional recurrent neural network (CRNN). To generate more sample images, GANs were proposed to synthesize handwritten text styles that generate image text for an insufficient dataset to generate more sample images. Moreover, GANs were applied in image restoration, such as deblurred images, super-high resolution, denoising images, etc. Furthermore, deep learning is still growing as a future in artificial intelligence and distribution to various areas such as self-driving cars, medical diagnosis, agriculture precision, manufacturing, etc.

As successful development of AI-based on deep learning, many deep learning architectures have been proposed to approach document analysis and recognition. This dissertation aims to solve handwritten text recognition in historical documents under the condition of various writing styles, cursive, and degraded documents, including noisy and low-quality document images. The research presented deep learning techniques for improving the effectiveness of recognition performances on Thai handwritten character and word recognition. The handwritten text recognition will be introduced as follows.

1.1 Handwritten Text Recognition in Historical Documents

Handwritten Text Recognition (HTR) using the deep learning method are a successful method to achieve high performance. Indeed, deep learning can reduce hand-crafted feature engineering with robust automatic features. However, the challenges of historical documents are low-quality images, various writing styles, and difficulty of character segmentation.

The following section describes in detail the text handwritten recognition system and the approach of deep learning techniques.

1.1.1 Handwritten Text Recognition System

Our research includes different topics consisting of text image restoration and generation. In general, HTR systems in Figure 1 consist of the processes of pre-processing, text segmentation, and recognition. These approaches are explained in the following sections.

1.1.1.1 Data Collection

This process involves transforming the original into a digital document. The document has been scanned and stored as an image dataset which can be recognized and performed in the following process.

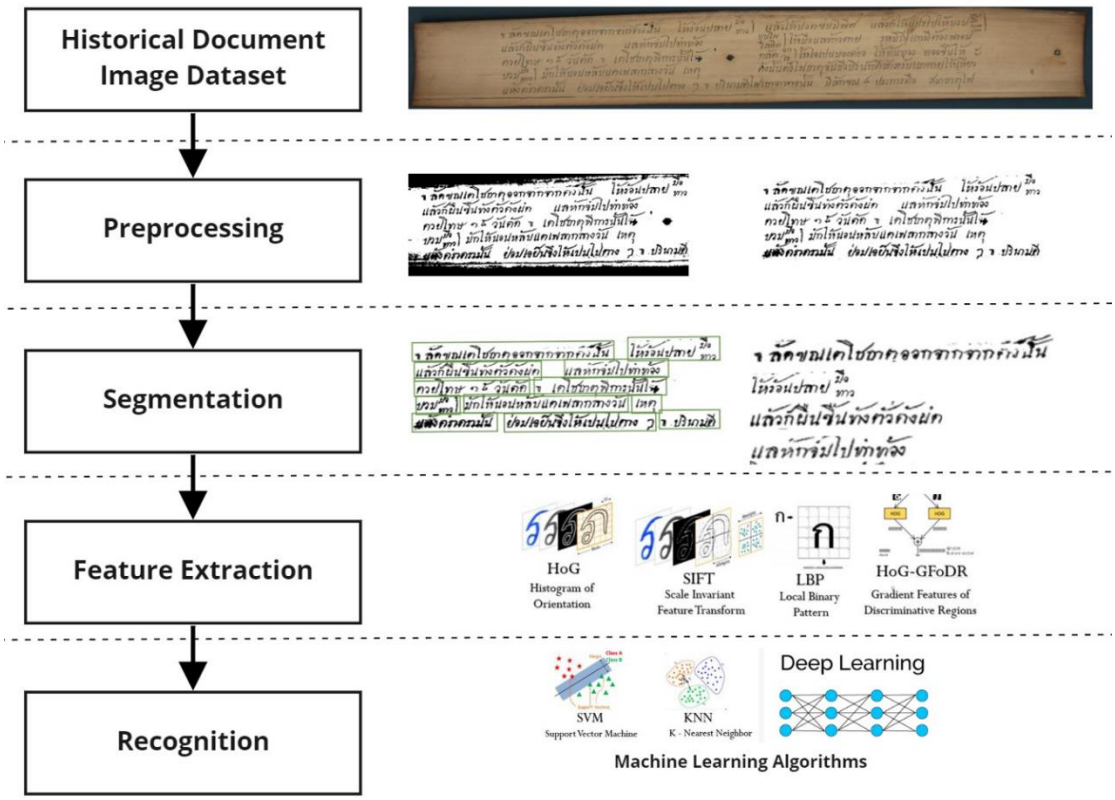


Figure 1 An Overview of handwritten text recognition.

1.1.1.2 Pre-processing

This process aims to enhance the quality of input images to clean and prepare text images to be easier recognized. There are operations includes:

1) *Skew Correction* obtains to correct the orientation of an image. It may be aligned at any angle, and then skew Correction applies to guarantee the image is forwarded to a subsequent process.
2) *Binarization* is applied for converting color images to binary images using local maxima and minima methods, Otsu's and adaptive thresholding, etc.

3) *Background Cleaning and Noise Removal*, old document images such as palm leaves can appear as noise background or, during a scan process, can be caused a shadow on an image. This process removes background and noise from various image sources to make clean and uniform input.

1.1.1.3 Segmentation

After the text image preprocessing, segmentation is a technique of breaking the image into subparts to further process. Document recognition can be divided into three segmentation levels: blocks or lines, words, and characters. The standard method is histogram projection which determines the number of foregrounds and background pixels to segment the image.

1.1.1.4 Feature Extraction

This process extracts features from an input image to make more discriminative with another images — traditional machine learning is mostly not feature learning. Feature extraction methods such as histogram of orientation, scale-invariant feature transform, and local binary pattern are proposed to increase the quality of recognition performance. Deep learning techniques, including CNN, RNN, and LSTM are used to extract features for recognition or query images

1.1.1.5 Recognition

The final process of the HTR system is the decision-making process. This process inputs segment image or image features in the machine learning model to produce the final output of the result text. Traditional machine learnings are support vector machines (SVM), K-nearest neighbor (KNN), or multilayer perceptron (MLP). Recently, deep learnings are CNN, RNN, LSTM, CRNN, and Transformer, which are applied in handwritten character recognition (HCR) and word recognition.

1.1.2 Deep Learning Techniques

Early research was based on handcrafted feature extraction, in which the HTR system needed to separate 2 processes of feature extraction and recognition by machine learning algorithms. In the development of deep learning, it can automatically feature learning, achieving high performance in a single model. There are many deep learning techniques applied in real word applications such as healthcare, automotive industry, smart city, social application, stock market analysis, etc. In handwritten text recognition, deep learnings were applied as the following:

1.1.2.1 Convolutional Neural Network

Early CNN was introduced by LeCun, et al. in 1998 and is used in computer vision, image recognition, segmentation, object detection, image restoration, etc. The general structure of the model shown in the Figure 2 consists of various layers and details as follows.

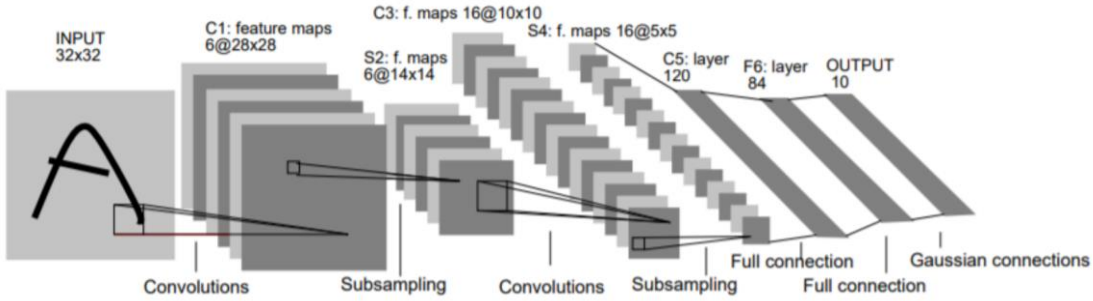


Figure 2 CNN Architecture (LeCun, et al. in 1998).

1) *Convolution Layer* is the main layer of CNN used to create a feature map with convolution operator (∗) as shown in Equation 1, where x_n^p is input with n channels and located at the layer p, filter kernel K with size m x n channels. It has an output or a feature map x_m^{p+1} , and m is the number of channels.

$$x_m^{p+1} = K_{m,n} * x_n^p \quad (1)$$

The feature map output obtained from each layer is partially passed either through a batch normalization (BN) layer (Ioffe & Szegedy, 2015) or a rectified linear units (ReLU) activation function (Nair & Hinton, 2010), as shown in Equation 2.

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

2) *Pooling Layer* is a spatial computation to help reduce the number of parameters in the network. This layer is used to find the maximum partition. In addition, the global average pooling (GAP) found in the network in network (NiN) research (Lin et al., 2014) reduced the number of dimensions of width and length (W x H) to single one value. The current CNN structure usually applied GAP instead of the pooling layer
3) *Connected Layer (FC)* connects all nodes from one layer to every node of the next layer and the last layer of CNN is used to recognize. The number of nodes equals the number of categories. The Softmax function was used to calculate the output as shown in Equation 3, where x_i is the characteristic vector and i is the order component of the vector x.

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

CNN has become highly influential in image recognition research since the AlexNet model was introduced by Krizhevsky et al. (2012) to learn the ImageNet dataset (Deng et al., 2009)(Deng et al. 2009). After that, other CNN structures have been improved and developed, including VGG (Simonyan and Zisserman 2015), which introduces convolution with a 3x3 kernel size only. The parameters used are

reduced. In contrast, GoogLeNet (Szegedy et al. 2015) proposes using various kernel sizes. InceptionV2 and V3 (Ioffe and Szegedy 2015; Szegedy et al. 2016, 2017) used Batch Normalization(Ioffe and Szegedy 2015), ResNet (He et al. 2016), and Inception-ResNet (Szegedy et al. 2017). Residual Connection has been used to increase the number of tiers to more than 100, and DenseNet (Huang, Liu, et al. 2017) has proposed a dense blocks model that combines the characteristics of the previous layer, giving that layer more information. Another approach to improvement is to reduce the computational scale. SqueezeNet (Hu et al., 2018) and InceptionV2 - V4 adopted a matrix factorization to reduce a number of parameter weights, MoblieNet (Howard et al. 2017) used depthwise separable convolution modules and MobileNetV2 (Sandler et al. 2018) add the inverted residual and linear bottleneck in place of the standard convolution model. Moreover, autonomous structuring by auto-machine learning, NASNet, was found in the network architecture search (NAS) research (Zoph and Le 2017). Reinforcement learning was used. (Reinforcement learning) and RNN to propose a model of the optimal structure and the AmoebaNet network, an evolutionary search algorithm. (Evolutionary Algorithm) and EfficientNet (Tan and Le 2019) looks to scale CNNs across the width. Length and solution size (Resolution) of the feature filter. NasNet has improved recognition performance over custom structured models. But it takes longer to calculate. Due to the number of structures searched, there are many possibilities.

1.1.2.2 Generative Adversarial Networks

GAN was proposed by Goodfellow et al. (2014) to be used for data generation that can generate data similar to the real sample dataset x (real data). The generator ($G(z;\ \theta_g)$) is responsible for generating comparative data distributed on probability p_g (fake data distribution) from random inputs on the latent space $p_z(z)$ (random distribution in latent space), and discriminator $D(x;\ \theta_d)$ is responsible for separating the real and fake data sets. Through the training process of the network, the GAN seeks to learn to achieve the maximum capacity of the separator that will recognize the actual sample dataset x from the constructor's output. The constructor attempts to create pseudo-data that can be deceived by the discriminator. As shown in Equation 4.

Many researchers have presented a structure that can improve the quality of images to be more apparent. It consists of 3 GAN networks: CycleGAN, SRGAN, and DeblurGAN. GAN has been used in image creation for various applications, including natural style transfer, image restoration, deblurring, and face generator.

$$\min_G \max_D V(D,\ G) = \boldsymbol{E}_{\boldsymbol{x\sim p_{data}(x)}}[\log D(x)] + \boldsymbol{E}_{\boldsymbol{z\sim p_z(z)}}[\log(1 - D(G(z)))] \qquad (4)$$

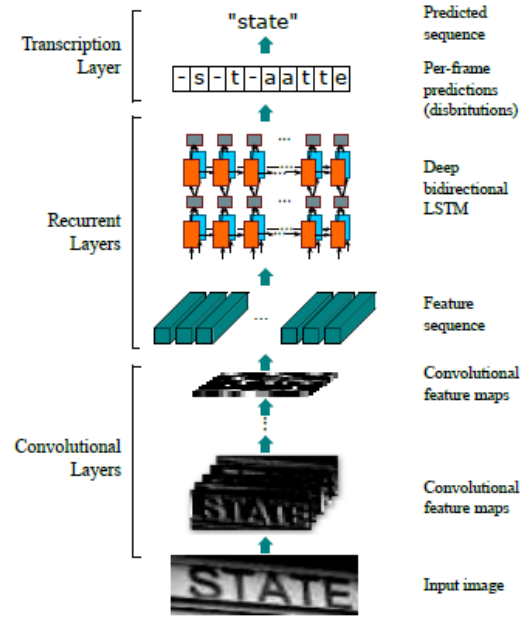


Figure 3 CRNN Architecture for word recognition (*Shi et al.*, 2017).

1.1.2.3 Convolutional Recurrent Neural Network

CRNN is a combination of CNN and RNN and has distinctive advantages over conventional neural network models. It can be directly learned from sequence labels (for instance, words), requiring no detailed annotations (for instance, characters). The network architecture of CRNN is shown in Figure 3.

1.2 Research Aim

This research aimed to approach deep learning methods to improve the performance of handwritten text recognition in a historical document.

1.3 Research Questions

The main research question that motivates this dissertation is: How can we enhance the performance of handwritten text recognition in historical documents using the deep learning method? This dissertation proposes to investigate and approach deep learning techniques to deal with the problems of handwritten text recognition as the following research questions:

RQ1: Character recognition is a fundamental problem in document analysis and recognition. In historical document images, handwritten characters are usually challenging to solve due to various personal writing and cursive styles. Previous works aim to extract features from local descriptors as hand-crafted feature and

recognize them by machine learning techniques such as SVM, KNN, and MLP. In contrast, we propose investigating CNN architectures that can automatically extract and recognize features. Is it possible to improve the recognition performance of handwritten characters? And which CNN architectures are suitable for this problem?

RQ2: Document degradations are caused by document aging effects and image acquisition with light conditions or a moving camera. These problems, called noisy character images, can decrease the recognition performance of handwritten characters. How can we improve the recognition rate of noisy characters? Can we assume that denoise GAN to clean noisy image provides better accuracy result of CNN? Furthermore, can a single DeblurGAN-CNN network enhance performance when recognizing different types of noisy characters?

RQ3: Thai historical documents are cursive writing style and difficult to segment to each character. Indeed, we focus on word or line recognition by sequence learning method suitable for handwritten documents. CRNN is a deep learning technique that is applied to various text recognition problems such as sense text recognition, video subtitle, and handwriting document. What is the best combination of CNN and RNN to construct robust CRNN in word or line recognition? Furthermore, the limitation of the dataset is insufficient handwritten text images for training. We propose a novel data augmentation technique for training CRNN; Is it possible to enhance the performance of Thai handwritten word recognition?

To answer all these research questions, Chapter 2 to Chapter 4 describe the research that succeeded. We will present concrete solutions to these research questions in Chapter 5.

1.4 Contributions

The contributions of the dissertation are approaching deep learning techniques to recognition of text handwritten in historical documents. The work reported in this dissertation involved experiments on handwritten character recognition with clear and noisy character datasets and handwritten word recognition. The contributions of the dissertation are as follows.

In chapter 2, we investigate the performance of CNNs on Thai handwritten character recognition. The architecture of CNN in this research is composed of VGGNet and Inception-ResNet, which do not need to extract features of special image characteristics because convolutional layers in deep CNN can automatically extract lower-level features. To evaluate the performance of CNNs, CNNs were compared in both learning styles, including scratch learning and transfer learning. We did not use data augmentation to increase training data for learning the deep CNN in both architectures due to researcher wanted to compare the experimental results with the siftD+SVM (Surinta et al. 2015) and HOGfoDRs methods (Inkeaw et al. 2019). The experiment found that VGGNet architecture with transfer learning was more

effective in recognition than other methods. Therefore, this method is suitable for solving the problem of character recognition in Thai handwritten. This chapter is based on the following publication. -

Gonwirat, S., & Surinta, O. (2020). Improving Recognition of Thai Handwritten Character with Deep Convolutional Neural Networks. The 3rd International Conference on Information Science and Systems (ICISS), pages 82–87. ACM.

Chapter 3 focuses on the effect of noisy character recognition. We contribute main works as follows; Firstly, we proposed a new standard noisy Thai handwritten character dataset, called the n-THI-C68 dataset, to challenge other researchers to reconstruct the sharp handwritten character images. The new noisy handwritten character images were synthesized by adding five noisy methods: low resolution, low contrast, additive white Gaussian noise, motion blur, and mixed noise. Secondly, we proposed the generative adversarial networks (GANs), namely DeblurGAN (Kupyn et al. 2018), combined with the convolutional neural network (CNN) architectures, called the DeblurGAN-CNN architecture, to reconstruct the quality handwritten character images from the noisy handwritten character images and enhance the accuracy of the handwritten character recognition.

Gonwirat, S., & Surinta, O. (2022). DeblurGAN-CNN: Effective Image Denoising and Recognition for Noisy Handwritten Characters. IEEE Access, 10, 90133-90148.

Finally, Chapter 4 presents the new data augmentation strategy, namely CycleAugment. The proposed data augmentation strategy mainly minimizes the validation loss and avoids overfitting. We achieve our goal with a simplistic strategy and implementation. Furthermore, we offer the CycleAugment strategy that provides the ability to train the CRNN model with and without applying data augmentation techniques simultaneously. Importantly, our CycleAugment strategy confirms that it can handle every CRNN architecture. We evaluate the efficiency of the CycleAugment strategy on several CRNN architectures for handwritten word recognition on Thai archive manuscripts. The content of this chapter is based on the following publication.-

Gonwirat, S., & Surinta, O. (2022). CycleAugment: Efficient Data Augmentation Strategy for Handwritten Text Recognition in Historical Document Images. Engineering and Applied Science Research, 49(4), pages 505-520.

Chapter 2

Improving Handwritten Character Recognition by Convolutional Neural Network

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 are also a problem. These similarities have led to recognition mistakes. This chapter 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 when tested with 10-fold cross-validation

2.1 Introduction

Character recognition is fundamental to research that can lead to document analysis, text transcription, or development of automatic reading systems (Marinai 2008). The recognition method can be beneficial in many fields, e.g. historical document recognition systems, text image retrieval, signature verification, and traffic-sign recognition.

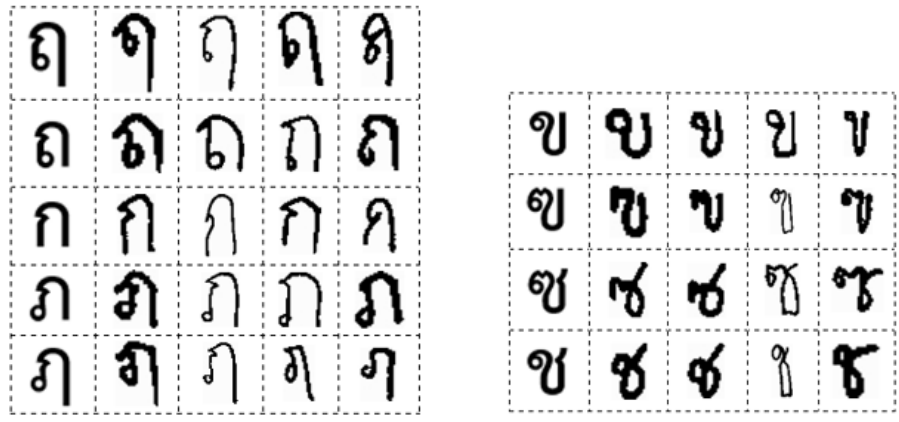
In general, the data used in recognition research about handwritten character recognition (HCR) includes digit, vowel, consonant, and special characters which depend on the writing style of each country (Kim and Xie 2015; LeCun, et al. 1998; Surinta, Karaaba, et al. 2015). The widespread traditional method is the feature extraction method, including histogram of oriented gradients (HOG), scale-invariant feature transform (SIFT) (Surinta et al. 2015), and Local Binary Pattern (LBP) (Joseph and Anantaprayoon 2018). Subsequently, the extracted data are made as an input for various types of machine learning, including K-Nearest Neighbors (KNN) (Surinta et al. 2015), Support Vector Machine (SVM) (Inkeaw et al. 2019; Surinta et al. 2015), Multi-layer Perceptron (MLP) etc.

The CNN method [10], which is a deep learning algorithm for fixing the problems in HCR (Kim and Xie 2015; LeCun, et al. 1998), 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 (Simonyan and Zisserman 2015) consists of 16 and 19 layers, ResNet (He et al. 2016) 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 (Hu et al. 2018) and global average pooling (GAP) layer (Inkeaw et al. 2019; Surinta et al. 2015). The regularization can also be used as an adjustment for weight (Wang and Klabjan 2017), dropout, and batch normalization (Ioffe and Szegedy 2015) 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 4(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 4(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 (Inkeaw et al. 2019; Surinta, Karaaba, et al. 2015), have used various methods to find unique characteristics. Surinta et al. (Surinta et al. 2015) used two local descriptor methods; SIFT Descriptor and 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 results show that siftD with SVM was the most effective method at 94.34% accuracy.



(a) (b)

Figure 4 Examples of similar character groups (a) characters with different tail traces and (b) characters with different indentation at head positions.

Inkeaw et al. (Inkeaw et al. 2019) 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 chapter 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 (Simonyan and Zisserman 2015) and Inception-ResNet (Szegedy et al. 2017), 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 (Surinta et al. 2015) and HOGfoDRs methods (Inkeaw et al. 2019). 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.

2.2 Related Works

2.2.1 Convolutional Neural Network

The convolutional neural network (CNN) presented by LeCun, et al. (1998) for English character recognition. CNN has become popular in image recognition after Krizhevsky et al. (2012) 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 (Huang et al. 2017), and MobileNet (Sandler et al. 2018). Each CNN had different architecture and different name, e.g. number of convolutional layers, inception module (Ioffe and Szegedy 2015; Szegedy et al. 2017), shortcut connection module (He et al. 2016; Sandler et al. 2017; Szegedy et al. 2018), and depthwise convolutional filters (Sandler et al. 2018). The basic structure of CNN architecture describes as follows;

2.2.1.1 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 5. 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.

$$X_{k,l,n}^{p+1} = \sum_{i,j,m} K_{i,j,m,n}\, x_{k+i-1,l+j-1,m}^{p} \quad (5)$$

Output or feature map from each layer was sent to the activation function in a Rectified Linear Units (ReLU): as shown in Equation 2 (Nair and Hinton 2010). Then, it was sent to batch normalization (BN) process (Ioffe and Szegedy 2015). 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.

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.

2.2.1.3 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).

2.2.2 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 (Ruder 2016) for weight parameters adjustment (shown in Equation 7).

$$\theta_{t+1} = \theta_t + v_{t+1} \quad (6)$$

$$v_{t+1} = \mu v_t - \alpha \nabla f(\theta_t) \quad (7)$$

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 8.

$$\alpha_{\mathrm{t}} = \frac{\alpha_0}{1+dt} \quad (8)$$

where α_{o} is the initial learning rate, d is learning rate decay.

2.2.3 CNN based Scratch and Transfer Learning

CNN learning method was divided into 2 processes; comprising learning from scratch and transfer learning. Learning from scratch (Okafor et al. 2016) 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 (Okafor et al. 2016; Sawada and Kozuka 2016) 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.

2.3 CNN Architectures

Since 2012, researchers have developed high effective CNN Architectures with structural adjustment methods, e.g. VGGNet (Simonyan and Zisserman 2015). In addition, the layers were increased up to 16 and 19 layers. GoogLeNet architecture (Ioffe and Szegedy 2015; Szegedy et al. 2017) 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 (He et al. 2016) 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 (Szegedy et al. 2017) and DenseNet (Huang, Liu, et al. 2017). In this paper, two CNN architectures were tested. These were VGGNet and Inception-ResNet.

2.3.1 VGGNet Architecture

In 2014, a research team sent VGGNet (Simonyan and Zisserman 2015) 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.

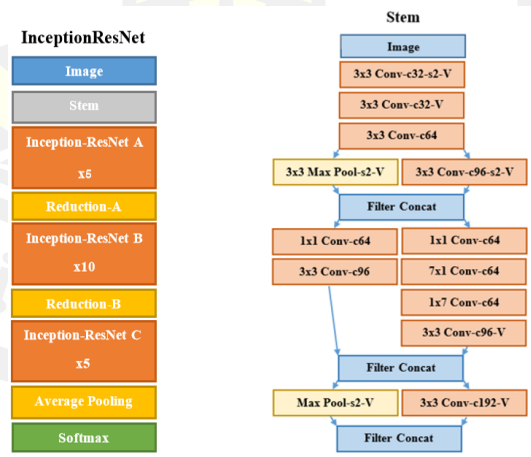
Table 1 shows VGGNet architecture with 16 and 19 layers. VGG16 consists of convolution layer (Conv) with 3x3 (Conv3), 13 filter layers, and 3 FC layers, total 16 layers. The amount of feature maps has increased to 64, 128, 256, 512, 512, and 512 layers consequently. Max Pooling Layers were added between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

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

Table 1 Configuration of the VGG16 and VGG19 architectures

2.3.2 Inception-ResNet-v2 architecture

Inception-ResNet-v2 (Szegedy et al. 2017) 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.



(a) (b)

Figure 5 Inception-ResNet-v2 architecture. (a) Core architecture and (b) detail of the Stem block.

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 5(a).

2.3.2.1 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 5(b)

2.3.2.2 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 6. The gap between Inception-ResNet blocks are separated by Reduction blocks due to parameter reduction.

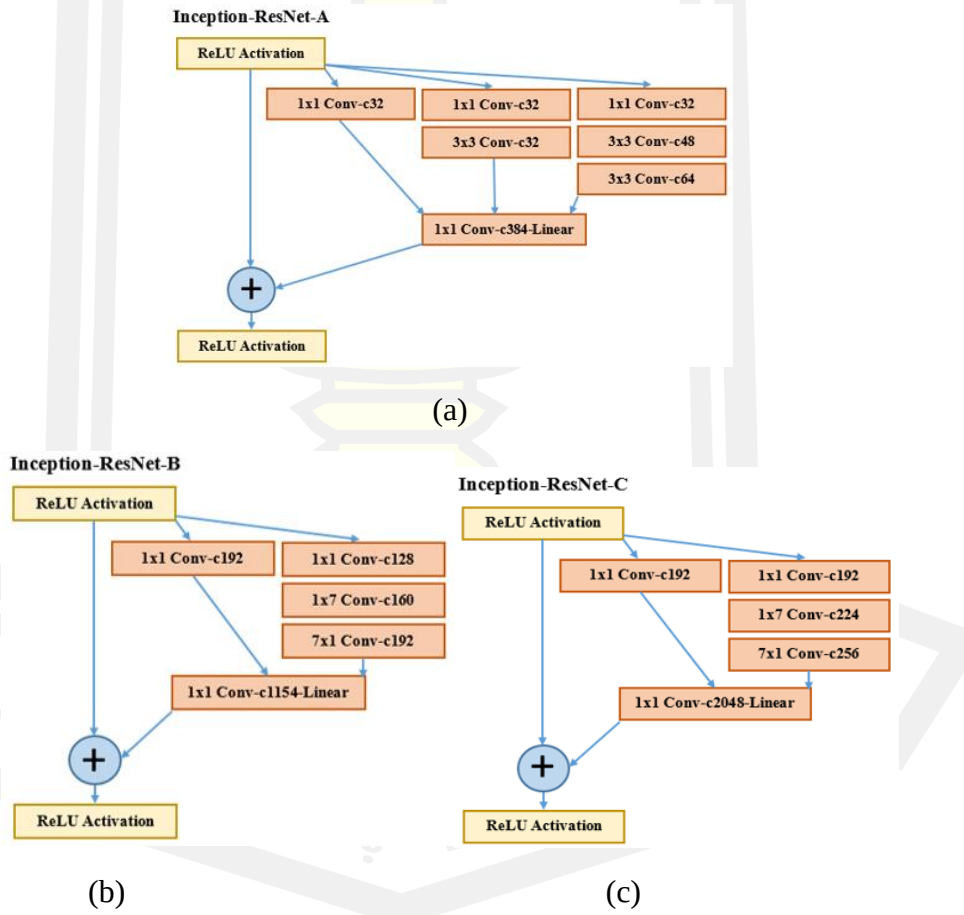
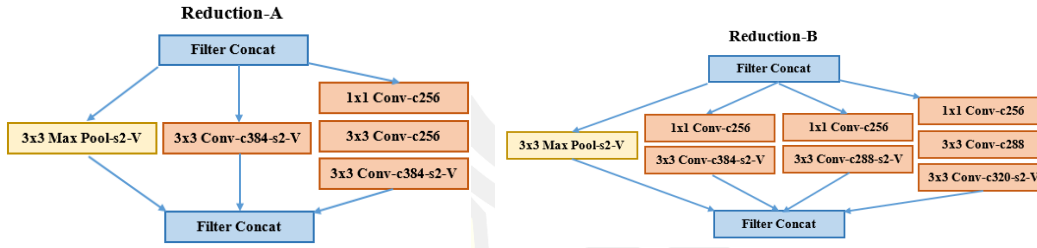


Figure 6 Architecture details of the Inception-ResNet. block (a) A, (b) B, and (c) C.



(a) (b)

Figure 7 The Reduction block (a) A and (b) B.

2.3.2.3 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 7.

2.4 Thai Handwritten Character Dataset

The Thai handwritten character dataset in this research is ALICE-THI dataset (Surinta et al. 2015), 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 8.

Surinta et al. (2015) used special features siftD take it to learn by SVM algorithm. The accuracy rate was 94.37%. Moreover, Inkeaw et al. (2019) 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.

Figure 8 Example of 68 Thai handwritten characters.

2.5 Experimental Result

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 (Simonyan and Zisserman 2015) and Inception-ResNet-v2 architectures to compare the performance between both deep CNN architectures. The effectiveness of the methods were compared with siftD-SVM (Surinta et al. 2015) and HOGFoDRs-SVM (Inkeaw et al. 2019) 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 (Deng et al. 2009). It was called Fine-tuned.

To ensure comparison equality, data augmentation was not used in (Inkeaw et al. 2019; Surinta, Karaaba, et al. 2015). This research also did not use data augmentation due to several studies e.g. (Okafor et al. 2016), 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.

Methods	Accuracy Rate (%)	
	10-cv	**5-cv**
SiftD-SVM (Surinta et al. 2015)	94.34	-
HOGFoDRs-SVM (Inkeaw et al. 2019)	-	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

Table 2 Performances of different models on THI-C68 dataset

Surprisingly, InceptionResNet-v2 architecture (Szegedy et al. 2017) 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, a smaller number of parameter, faster speed in the experiment, and highest accuracy rate. The most important is the model that achieved the highest accuracy rate.

2.6 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 increase 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.

Chapter 3

Denoising and Recognition Deep Neural Network for Noisy Handwritten Characters

Many problems can reduce handwritten character recognition performance, such as image degradation, light conditions, low-resolution images, and even the quality of the capture devices. However, in this research, we have focused on the noise in the character images that could decrease the accuracy of handwritten character recognition. Many types of noise penalties influence the recognition performance, for example, low resolution, Gaussian noise, low contrast, and blur. First, this research proposes a method that learns from the noisy handwritten character images and synthesizes clean character images using the robust deblur generative adversarial network (DeblurGAN). Second, we combine the DeblurGAN architecture with a convolutional neural network (CNN), called DeblurGAN-CNN. Subsequently, two state-of-the-art CNN architectures are combined with DeblurGAN, namely DeblurGAN-DenseNet121 and DeblurGAN-MobileNetV2, to address many noise problems and enhance the recognition performance of the handwritten character images. Finally, the DeblurGAN-CNN could transform the noisy characters to the new clean characters and recognize clean characters simultaneously. We have evaluated and compared the experimental results of the proposed DeblurGAN-CNN architectures with the existing methods on four handwritten character datasets: n-THI-C68, n-MNIST, THI-C68, and THCC-67. For the n-THI-C68 dataset.

3.1 Introduction

Character recognition is a sub-process of text recognition systems used to recognize handwritten and printed texts within document images, such as historical documents, memoranda, and archival material. Therefore, when the main objective is to focus on the effects of handwritten character recognition, the factors that affect are as follows. 1) Writing styles; the distinctions of writing in each era, the diversity of individual writing styles, and even writing types of equipment (Surinta et al. 2015; Alom et al. 2018). 2) Degradation of historical documents; this maybe due to a lack of expert staff and the humidity of a storage location. 3) Digital transformation; blurred and noisy document images were created when using low-quality equipment and taking the picture with a camera without adequate lighting. 4) Limitations of data; an insufficient and uncovered dataset of handwritten character images in the training process. These factors need to be considered when recognizing handwritten text images.

The factors mentioned above directly affect machine learning, leading to decreased recognition performance. In the case of noise when digitizing ancient documents, Su el at. (2019) experimented with noise generation using the differential

evolution method to determine the optimal position for digitization. Adding one pixel to the original image (called a one-pixel attack) logically is the trick that causes the convolutional neural network (CNN) models to be misrecognized. Their experiments showed that adding one pixel could harm the CNN model by increasing the recognition errors. Mei et al. (2019) demonstrated that blurred images affect the recognition rate. Subsequently, the DeepDeblur algorithm was invented to transform blurred into sharp images before sending the sharp images for recognition. Also, the sharp images caused the model to increase its recognition performance.

Recently, CNN has replaced traditional machine learning (LeCun, Bengio, and Hinton 2015) and is widely used in handwritten character recognition. Since the CNN method is an automatic algorithm that consists of feature extraction techniques and image recognition, it is currently used in character recognition in many languages, such as Latin, Arabic, Bangla, Korean, Chinese, and Thai (Surinta et al. 2015; Alom et al. 2018; Gonwirat and Surinta 2020; Eltay et al. 2022), resulting in increased character recognition efficiency. However, if the training images are low quality and noisy, they will significantly reduce recognition efficiency (Mei et al. 2019; Su et al. 2019).

Furthermore, deep learning techniques, including CNN, auto-encoder, and generative adversarial network (GAN), have also been proposed to improve image restoration and denoising. Dong et al. (2016) proposed the image restoration technique using the CNN technique. The objective of their study was to transform the low-resolution images into high-resolution images. They proposed the super-resolution CNN method, which is a lightweight deep learning architecture that quickly restores and reconstructs quality images. Zhang et al. (2017) presented feed-forward denoising CNNs, which integrate single residual learning into the CNN architecture for denoising images and to manipulate blind Gaussian noise without unknown noise levels. Further, Gondara et al. (2016) proposed a convolutional denoising autoencoder to denoise the signal from the medical images and Souibgui et al. (2022) proposed an encoder-decoder architecture based on vision transformers, called DocEnTr, to enhance degraded document images.

The GAN architecture is widely used in many domains, especially for image restoration and deblur (Goodfellow et al. 2014; Isola et al. 2017; Kupyn et al. 2018). The GAN architecture is designed as a generator that is capable of learning from many images and recreating a new image. The adversarial loss function in the GAN architecture is used to create a robust model that aims to create high-quality images during regeneration. DeblurGAN (Kupyn et al. 2018) was first employed by using the learning process of the WGAN-GP (Gulrajani et al. 2017) and used perceptual loss (Johnson, Alahi, and Fei-Fei 2016), allowing the model to deblur images in the form of blind motion blur that can be caused by camera movement during a photograph. Consequently, GAN is designed to solve the problems of document images, such as cleaning noisy backgrounds, deblurring text in the documents, and

regeneration of damaged characters into the complete characters (Sharma et al. 2018; Bhunia et al. 2019; Khamekhem Jemni et al. 2022; Souibgui and Kessentini 2022).

Contributions: This research presents the DeblurGAN-CNN architecture that aims to solve the recognition problems of noisy handwritten character images. The proposed DeblurGAN-CNN architecture improved the image quality and resulted in higher performance of handwritten character recognition on various handwritten character and noisy character datasets. The contributions of our research are the following.

1) This paper proposes a new standard noisy Thai handwritten character dataset, called the n-THI-C68 dataset, to challenge other researchers to reconstruct sharp and clean handwritten characters. The noisy handwritten character images were synthesized by adding five noisy methods: low resolution, low contrast, additive white Gaussian noise, motion blur, and mixed noise. The n-THI-C68 dataset includes 68 classes and contains 11,592 character images in the training set and 14,290 character images in the test set.

2) We propose the deblur generative adversarial networks (GANs) combined with the convolutional neural network (CNN) architectures, called the DeblurGAN-CNN architecture, to reconstruct high-quality handwritten characters from noisy handwritten characters and simultaneously enhance the accuracy of the handwritten character recognition systems. In the DeblurGAN-CNN architecture, DeblurGAN is proposed to learn from the noisy images and regenerate the new sharp and clean handwritten character images. Hence, the reconstructed handwritten character images are assigned to the CNN architecture for recognition.

3.2 Related Work

3.2.1 Generative Adversarial Network

The generative adversarial network (GAN) was first presented by Goodfellow et al. (Goodfellow et al. 2014). GAN is an unsupervised learning model that automatically learns from the regularities of input images and is then capable of creating a new image that is similar to the original image. Therefore, GAN has been applied in a wide range of applications, such as natural transfer style, image super-resolution, face generation, image restoration, and even image deblurring (Kupyn et al. 2018; Wang et al. 2018; Karnewar and Wang 2020; Wang el at. 2021).

Since GANs have generative ability and style transformation, they were applied in the data augmentation technique to improve recognition performance for document images. Fogel et al. (2020) proposed ScrabbleGAN, which is semi-supervised learning by using unlabeled and labeled samples during the training process, to synthesize different Latin and French handwritten text styles. In addition, Eltay et al. (2022) proposed adaptive data augmentation based on the ScrabbleGAN architecture

to recognize Arabic handwritten text. The adaptive method generated more balanced characters in training samples.

Moreover, many issues in documents, such as blurred image, noisy background, salt-and-pepper, and faded text, lead to the document being unreadable to humans, significantly decreasing the recognition performance of the text algorithms (Sharma el at. 2018; Bhunia et al. 2019; Khamekhem et al. 2022; Souibgui and Kessentini 2022).. To solve these problems, Bhunia et al. (Bhunia et al. 2019) proposed two networks, including texture augmentation and binarization networks, to binarize the degraded document images. First, the texture augmentation network was designed to create multiple textual contents with diverse noisy textures to increase the size of the document binarization dataset. Second, the binarization network generated new images, which are the clean binary document images. Sharma et al. (2018) used CycleGAN to remove the noise from the documents resulting in cleaned documents. The CycleGAN model was employed to map noise to clean documents and clean to noisy documents using the cycle consistency loss function. Their experiment showed that the CycleGAN provided acceptable results. In terms of document enhancement, Souibgui and Kessentini (2022) applied conditional GAN, which is a single GAN network, to restore various problems of mixed document degradations, including tasks of document clean up, binarization, deblurring, and watermark removal.

Furthermore, Wu et al. (2020) applied Wasserstein loss to the CycleGAN that improved the CycleGAN algorithm to deblur text images into clear text images. Also, Zhao et al. (2020) used the GAN model to optimize the distortion of input images before feeding the rectified images to the text recognizer.

Since the first GAN architecture was presented in 2014 [12] to regenerate a new image similar to the original image, many GAN architectures have been proposed to solve the problems, for example, noisy images, degraded documents, and blur text, in the domain of document images. We then have the concept of using the GAN architecture to denoise the handwritten character images before recognizing them using the CNN architecture.

3.2.2 Convolutional Neural Network

CNN architectures achieved high efficiency on image classification problems, Su et al. (2019) demonstrated an image generation technique that only added one pixel into the target image based on the differential evolution technique. With only one attack pixel, the accuracy performance significantly decreased. For handwritten character recognition, many noisy methods were applied to the character images, such as motion blur, low contrast, and additive Gaussian white noise (AGWN) (Basu et al. 2015), to demonstrate that the noise images could significantly reduce the recognition performance. Consequently, to increase the recognition efficiency, a synthesized image technique was introduced to remove noise before sending images to recognition.

CNN architectures have been proposed for image classification purposes. The CNN architectures combine two main tasks (feature extraction and machine learning) into one architecture to specifically reduce the complex feature extraction processes. Many state-of-the-art CNN architectures have been proposed and have become successful in many domains. For example, AlexNet, GoogLeNet, VGGNets, MobileNets, ResNet, DenseNet, NASNet, and EfficientNet. However, the latest CNN architectures operate with more deep layers, convolution operations (i.e., 1D, 2D, 3D convolution (Ji et al. 2013; Khan et al. 2020), and depthwise separable convolution), and extra layers (i.e., global average pooling, inception module, reduction cell) (Zoph et al. 2018) to compute the robust spatial features from the image. Therefore, the researcher could propose new CNN architecture, invent new operations, and combine them with the existing CNN architectures.

From related work above, we found that the GAN architecture could be used to solve the problems of noisy images, while various CNN architectures could propose to recognize the noisy handwritten character images. The proposed denoising and recognition framework is described in-depth in the following section.

3.3 Denoising and Recognition Framework

Due to the performance of the handwritten character recognition is always affected by noise. We then proposed the DeblurGAN-CNN architecture to address the noisy problems. Although, many robust CNN architectures achieved high accuracy in every domain, even on handwritten character images. However, the accuracy unexpectedly decreases when affected by many types of noise, such as blur, low resolution, and low contrast. In this research, we first studied the effect of the noisy character images that harm the performance of handwritten character recognition. Second, the data augmentation techniques were applied while training the CNN model to increase new patterns of the handwritten character images. The data augmentation methods could generalize the CNN model when the noise is not adequately high. Hence, the performance decreased after adding a high noise level. Third, we discovered that the DeblurGAN could transform the noise into new clean handwritten characters. Finally, DeblurGAN architecture and the robust CNN architecture are combined to enhance the recognition performance of the handwritten character images, called DeblurGAN-CNN.

There are several methods for improving image quality, for example, super-resolution, image restoration, and deblurring images. However, some noise appears in the handwritten character images while transforming the document papers into digital format. Consequently, we considered two GAN architectures (DeblurGAN and CycleGAN) to address our problems because these two GAN architectures are designed for deblurring images. However, the CycleGAN is mainly used for a style transfer that transforms from one style to another style. In comparison, many noises occur in the handwritten character images, which means CycleGAN is not appropriate for these problems. Furthermore, we used the DeblurGAN architecture that could deal with many-to-one style transfer.

In this paper, we proposed the DeblurGAN-CNN framework that combines two state-of-the-art deep learning architectures to denoise and recognizes the noisy handwritten characters into one architecture. The proposed framework contains a generator of generative adversarial network (GAN) and convolutional neural network (CNN) architectures, as shown in Figure 9.

In the following subsections, the details of the DeblurGAN-CNN framework are described. 1) DeblurGAN is employed as a denoising network. 2) DenseNet121 is the convolutional neural network architecture performed as a recognition network. 3) We describe the DeblurGAN-CNN architecture and training strategy that is used for training the proposed framework.

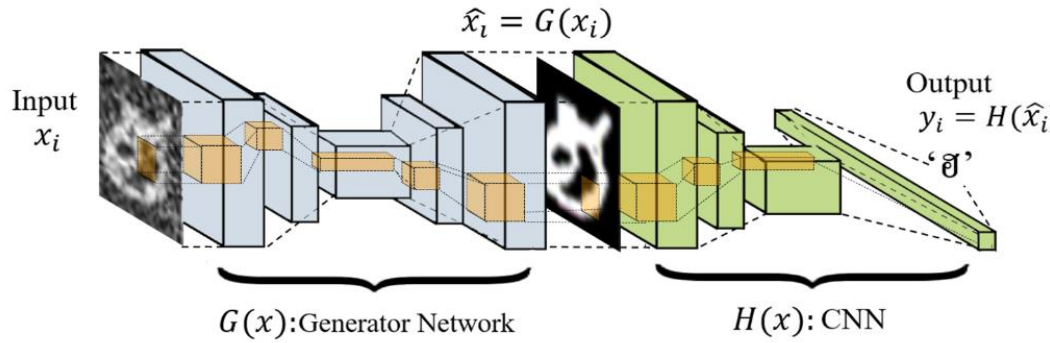


Figure 9 Illustration of the DeblurGAN-CNN architecture.

3.3.1 DeblurGAN

Kupyn et al. (2018) proposed the GAN architecture to automatically deblur blurred images from any unknown blur function, called DeblurGAN, which can synthesize sharp images (I_S) from blurred images (I_B). The DeblurGAN uses the generator (G_{θ_G}) and the discriminator (D_{θ_D}) to distinguish between real and generated images.

The generator architecture of the DeblurGAN is shown in Figure 10. The beginning part of the network consists of three convolutional blocks that are designed to downsample the feature maps. The middle of the network is a sequence of nine residual blocks. In the last part of the network, the transposed convolution blocks are constructed to upsample feature maps to the original size as an input image. Moreover, the global skip connection is also proposed for this architecture by adding input to the output image. The global skip connection makes the network converge faster and yields better output results.

In the DeblurGAN, the PatchGAN architecture (Isola et al. 2017) is used as the discriminator. The PatchGAN architecture has downscale convolutional layers

followed by instance normalization and leaky rectified linear unit (LeakyReLU) with α=0.2.

Consequently, as shown in Equation 9, the loss function is presented in the DeblurGAN that includes adversarial ($\mathcal{L}_{GAN}$) and content loss ($\mathcal{L}_X$) that is weighted by λ, where λ is a parameter that controls the relative of two objectives: adversarial and content loss. The WGAN-GP (Gulrajani et al. 2017), which is the critic function to determine the completeness of the generator result, is used as the adversarial loss, as shown in Equation 10. Also, the content loss is the perceptual loss (Johnson el at. 2016) to compare the style-transfer, called reconstructed image, with the original image using the L2 loss function.

$$\mathcal{L} = \underbrace{\underbrace{\mathcal{L}_{GAN}}_{adversarial\ loss} + \underbrace{\lambda \mathcal{L}_X}_{content\ loss}}_{total\ loss} \tag{9}$$

$$\mathcal{L}_{GAN} = \sum_{n=1}^{N} -D_{\theta_D}\left(G_{\theta_G}(I^B)\right) \tag{10}$$

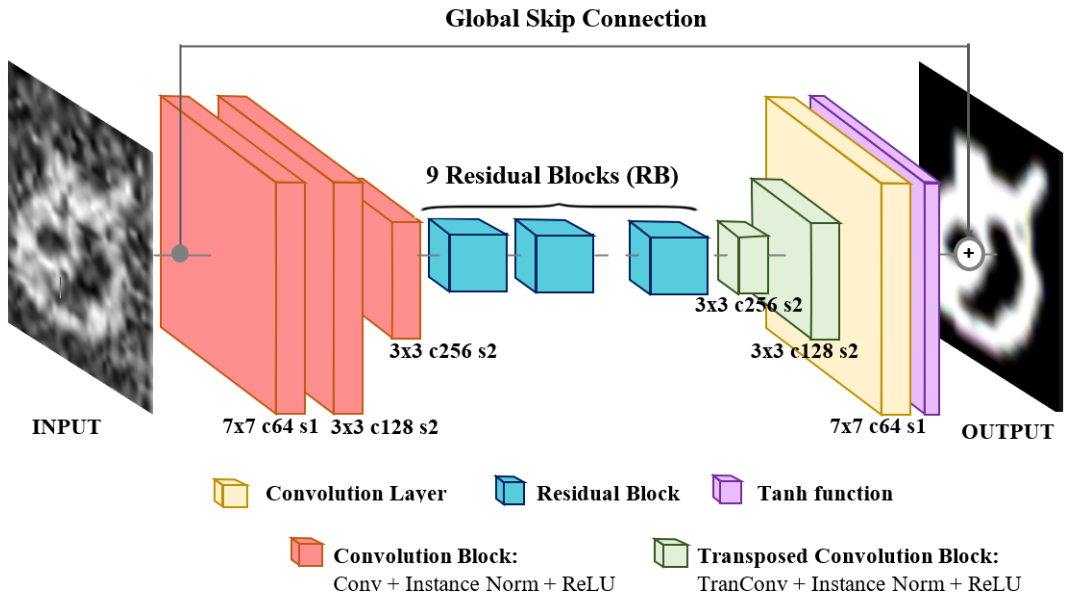


Figure 10 Illustration of the DeblurGAN generator architecture.

3.3.2 DenseNet

In the early architecture, a residual connection using element-wise input (x) with an output building block (F(x)) (He et al. 2016) was proposed, called ResNet. The benefit of the ResNet architecture was that the network could construct with deep convolutional layers and still obtain better results in terms of speed and performance. However, the DenseNet architecture (Huang et al. 2017) was designed to include the maximum information flow by concatenating all feature maps (x_n^p) from the previous

convolutional layers, called a dense block. DenseNet was proposed to deal with the reuse of the features, reduce the architecture parameters, and eliminate gradient problems. The equation of the DenseNet is shown in Equation 11.

$$x_n^p = H^p\left(\left[x_n^0, x_n^1, x_n^2, \dots, x_n^{p-1}\right]\right), \tag{11}$$

where $\boldsymbol{H^p}(\cdot)$ is the composite function of a layer (p), including batch normalization (BN), rectified linear unit (ReLU), and convolutional (Conv) layer. The function parameter $\left[x_n^0, x_n^1, x_n^2, \dots, x_n^{p-1}\right]$ is a concatenation of previous layers from the first layer (x_n^0) to last layer (x_n^{p-1}).

An overview of the DenseNet is shown in Figure 11(a). The DenseNet architecture consists of three main parts.: 1) A convolutional layer with a kernel size of 7×7. The convolutional block includes BN, ReLU, and Conv layers, with a stride of 2 and followed by a 3×3 max pooling layer with a stride of 2. 2) Four dense blocks and transition layers. 3) The global average pooling (GAP) and classification layers with a softmax function.

Details of the DenseNet architecture, are as shown in Figure 11(b). The dense block is concatenated with the output of bottleneck layers, which is expanded N times, proposed to decrease the parameters of the architecture. Each bottleneck layer consists of 1×1 Conv and 3×3 Conv layers, as shown in Figure 11(c). The transition layer (see Figure 3(d)) is proposed to reduce the feature map width and height by 2 ×2 average pooling with a stride of 2 and θ parameter applies to compress the network where a range of a parameter is $0<\theta\leq 1$.

In this chapter, we proposed to use DenseNet121 since it is the smallest size appropriate for handwritten character recognition.

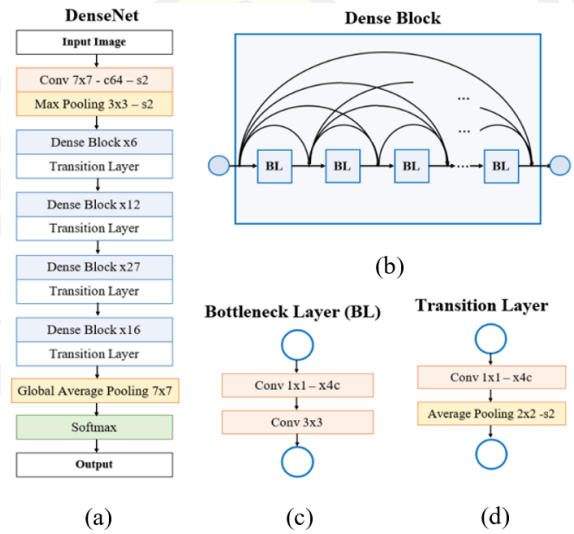


Figure 11 Illustration of the DenseNet121 architecture, including (a) core block, (b) dense block, (c) bottleneck layer, and (d) transition layer.

3.3.3 DeblurGAN-CNN Setting and Training

In this section, we provide the construction and training strategy of the proposed framework, as shown in Algorithm 1. Also, the details of the setting and training strategy of the DeblurGAN-CNN framework are described in the following.

Algorithm 1. Construction and training of the DeblurGAN-CNN framework

Input: Training set including pair of sharp and noisy character images $(x_i^{sharp}, x_i^{noisy})$ and label y_i, where $i = 1,2, \dots, n$,
training epochs of DeblurGAN: M,
training epochs of CNN: P.
Define: (X^D, Y^D) is training set of data augmentation technique $\{(x_i^{sharp}, \mathrm{y}_i)\} \cup \{(x_i^{noisy}, \mathrm{y}_i)\}$.

Step 1) Create DeblurGAN network including generator network $G(x)$ and discriminator $D(x)$.

Step 2) Train DeblurGAN using M epochs with dataset of pair set $\{(x_i^{sharp}, x_i^{noisy})\}$ and save the best model based on the loss function in Equation (1).

Step 3) Create CNN of pretrained weight from the ImageNet dataset.

Step 4) Train CNN using P epochs with the dataset (X^D, Y^D) and save the best model based on the loss function in Equation (4).

Step 5) Construct a DeblurGAN-CNN network as the following:
- Load the $G(x)$ network in the step 2).
- Load the CNN network in the step 4).
- Combine $G(x)$ and CNN with the intermediate layer.

Step 6) Fine tune the DeblurGAN-CNN network training using P epochs with the dataset (X^D, Y^D) and the loss function in Equation (4). The training steps consist of two steps as the following:
- Freeze the part of $G(x)$ in the network and train using $P/2$ epochs.
- Unfreeze and train all layers in the network using $P/2$ epochs.

Output: The DeblurGAN-CNN network

3.3.3.1 DeblurGAN Training

DeblurGAN was designed for deblurring images. However, in our problems, DeblurGAN was applied to reconstruct the sharp handwritten character images from the various noisy styles, such as low contrast, motion blur, and white Gaussian noise.

To train the DeblurGAN architecture, the dataset then includes the pairs of noisy and sharp handwritten character images, $(x_i^{noisy}, x_i^{sharp})$, where $i = 1,2, \dots, n$. In the DeblurGAN training process, the generator network receives a noisy image (x_i^{noisy}) as input and adjusts the weights to reconstruct the output as a sharp image (x_i^{sharp}). We evaluate the quality of the reconstructed handwritten character images using the discriminator and the loss function as shown in Equation 9.

3.3.3.2 CNN Training

We employed the CNN architectures to train on a handwritten character dataset that consisted of the pairs (x_i, y_i), where $i = 1,2, \dots, n$, x_i is handwritten character of character i and y_i is label of character i. To improve the efficiency performance, we proposed the transfer learning method (Gonwirat and Surinta 2020) with convolutional kernels of prior knowledge for faster convergence in a few epochs. The pre-trained CNN model was modified in the classification layer and then fine-tuned in the network. Furthermore, we trained the CNN models with the data augmentation techniques with noisy handwritten character images (x_i^{noisy}), which is synthesized from the original sharp images (x_i^{sharp}), where $x_i^{noisy} = f^{noisy}(x_i^{sharp})$ and $f^{noisy}(x)$ is the generator function of a synthesized noisy image. We trained the CNN model to classify images using categorical cross-entropy loss function as shown in Equation 12.

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} log\ (p_{CNN}\ (x_i; \theta)), \tag{12}$$

where N is the number of training images and $\boldsymbol{p_{CNN}}(x_i; \theta)$ is the probability distribution of CNN output, where x_i is an input image and θ is weight parameters.

3.3.3.3 DeblurGAN-CNN Construction

The DeblurGAN-CNN network connects the DeblurGAN generator and CNN, as shown in Figure 1. This proposed network benefits from the generator producing a sharp output image from various noisy images before recognizing it by the CNN model. The output of the generator $(G(x))$ is called the intermediate output $(\hat{x}_i)$, where $\hat{x}_i = G(x_i)$ is computed using the tanh function. Hence, the output image values are in the range of -1 and 1. Subsequently, we add the intermediate layer to convert the value to the range of 0 and 1, the same as the input of the CNN which the adjusting function is $f(x) = (x + 1)/2$ where x is intermediate output.

3.3.3.4 DebulrGAN-CNN Fine-Truning

The DeblurGAN-CNN network is still an incomplete merge network since a part of CNN has inexperienced generator output. Thus, fine-tuning the DeblurGAN-CNN network is an approach to improvement. In the first step, we only trained the CNN by freezing the DeblurGAN generator for stable network training and retraining the output as sharp images. In the second step, we trained the DeblurGAN-CNN

network with unfrozen whole layers. The proposed DeblurGAN-CNN network was trained with a few training epochs. We trained only ten epochs in each frozen step and each unfrozen step.

3.4 Handwritten Character Datasets

In this section, we briefly describe the handwritten character datasets used in the experiments, including two Thai handwritten character datasets: THCC-67 (Sae-Tang and Methasate 2004) and THI-C68 (Surinta et al. 2015), and two noisy handwritten character datasets: n-MNIST (Basu et al. 2015) and n-THI-C68. An overview of the handwritten character datasets is shown in Table 3.

Datasets	TYPES AND LANGUAGES	NUMBER OF CLASSES	TRAINING SETS	Test Sets
THCC-67 (Sae-Tang and Methasate 2004)	Char, Thai	67	-	9,012
THI-68 (Surinta et al. 2015)	Char, Thai	68	11,592	2,898
n-THI-68	Char, Thai	68	11,592	14,290
n-NMINST (Basu et al. 2015)	Digit, Arabic	10	180,000	30,000

Table 3 Overview of the handwritten character datasets.

(a) (b)

Figure 12 Examples of Thai handwritten character datasets: (a) THCC-67 and (b) THI-C68.

3.4.1 The NECTEC Thai Handwritten Character Corpus(THCC-67)

The National Electronic and Computer Technology Center (NECTEC) presented a Thai handwritten character corpus (THCC) of consonants, vowels, and tones that contains 67 classes, called THCC-67. The THCC-67 dataset has 9,012 characters that were rescaled to 32×32 pixels. In this research, we used it as an independent test. The THCC-67 dataset is shown in Figure 12(a).

3.4.2 The ALICE Offline Thai Handwritten Character Dataset (THI-68)

The THI-C68 dataset containing 28 classes was proposed by Surinta et al. (2015). The THI-C68 dataset was collected from 150 university students aged 20-23 years old. Students wrote the Thai characters on a form with a white background that was scanned with a resolution of 200 dpi. Image transformation was used to rescale the aspect ratio to avoid distortion and images were stored in grayscale format. The THI-C68 dataset has 14,490 character images containing consonants, vowels, and tones. An example of the THI-C68 is shown in Figure 12(b).

3.4.3 Noisy THI-C68 (N-THI-68)

In this research, we propose a new noisy Thai handwritten character dataset, called noisy THI-C68 (n-THI-C68). We synthesized new noisy character images using five different noisy techniques: low resolution, additive white Gaussian noise (AWGN), low contrast, motion blur, and mixed noise.

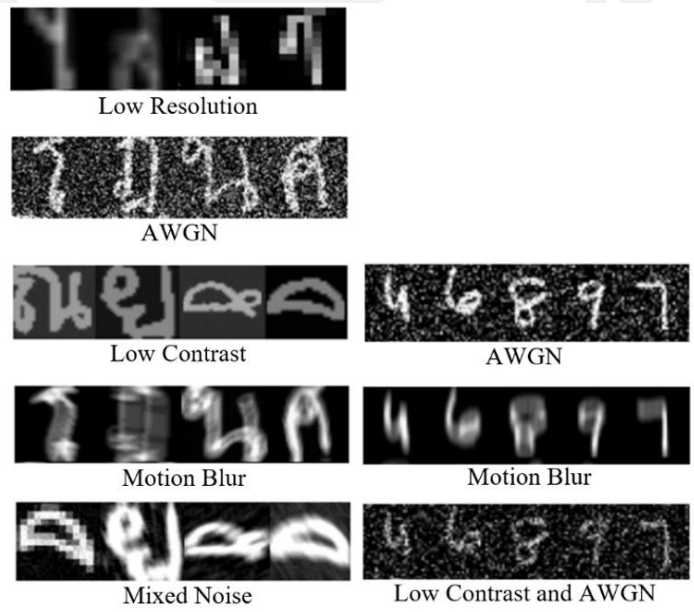


Figure 13 Examples of noisy handwritten character datasets: (a) n-THI-C68 that applied 1) low resolution, 2) AWGN, 3) low contrast, 4) motion blur, and 5) mixed noise and (b) n-MNIST that applied 1) AWGN, 2) Motion blur, and 3) low contrast and AWGN.

We randomly selected one noisy technique to synthesize each character image according to the THI-C68 dataset with 11,592 training images and 2,898 test images. We obtained 11,592 noisy character images for the training set that were randomly applied with noisy techniques with various adjustment values. For the test set, we increased the size from 2,898 character images up to 14,290 noisy character images by randomly applying five noisy techniques to the original character images.

As shown in Figure 13(a), noisy Thai handwritten character images were synthesized as follows. 1) Low resolution with a low level at 8-12 pixels. 2) AWGN with increasing noise with a peak signal to noise ratio (PSNR) of 9.5. 3) Low contrast with reduced color gradient in range of 0.15-0.5 based on the original images. 4) Motion blur with two blur methods: directional motion blur (Basu et al. 2015; Karki et al. 2018) and random motion blur (Boracchi and Foi 2011). 5) Mixed noise between four noisy methods.

3.4.4 Noisy MNIST (n-MNIST)

Karki et al. (2018) proposed the noisy MNIST (n-MNIST), which is the extended version of the MNIST dataset (LeCun, et al. 1998) that applied three noisy methods: AWGN, motion blur, and combinations between reduced contrast and AWGN. The n-MNIST dataset contains 10 classes (0-9) and has 180,000 training samples and 30,000 test samples due to applying three noisy techniques to the original images.

Figure 13(b) shows noisy digits were applied as follows. 1) AWGN using increase noise with RSNR of 9.5. 2) Motion blur using linear motion filter with a size of 5 pixels and rotation with 15 degrees, and a combination between reduced contrast and AWGN with a PSNR of 12.

3.5 Experiment Results

In this section, we evaluated the performance of the proposed DeblurGAN-CNN architecture on the handwritten character datasets and noisy handwritten character datasets. We then investigated the effective recognition of CNNs and the quality of image restoration by the generative adversarial networks (GAN). In this study, we trained the CNN and GAN models on Linux operating systems with Nvidia GeForce GTX1080ti 8G GPU, Intel(R) Core i5-7400 Processor 3.00GHz CPU, 32GB DDR4 RAM.

3.5.1 Evaluation of The CNN Architectures on THI-C68 Dataset

3.5.1.1 COMPARISON of state-of-the-art CNNs

We evaluated four CNN architectures: VGG19, InceptionResNet, MobileNetV2, and DenseNet121 on the Thai handwritten character dataset to find the best CNN architecture. We divided the THI-C68 dataset into a training set and test set with 80% and 20% ratios, with 13,041 training images and 1,449 test images. Hence,

the training set was evaluated using 5-fold cross-validation. The test set was an independent holdout set for final evaluation.

Furthermore, we focused on three training methods: 1) scratch learning (SL), 2) transfer learning (TL), and transfer learning with noisy data augmentation techniques (TL-nDA).

We proposed four noisy data augmentations: low resolution, AWGN, motion blur, and mixed noise, which were generated as a training set of the n-THI-C68 dataset.

The hyperparameters in CNN models were defined as follows: training epochs = 100 epochs, batch size = 32, stochastic gradient descent (SGD) optimizer, learning rate = 0.001, decay rate = 0.0001, momentum = 0.9, and image size = 128×128 pixels which is the smallest input of the InceptionResNet architecture. In transfer learning, we also used the pre-trained CNN model that learned on the ImageNet Dataset (Deng et al. 2009).

The accuracy results of CNN architectures are shown in Table 4. The accuracy performance of the CNNs was above 97% accuracy. The VGG19 architecture achieved the lowest performance on the THI-C68 dataset with an accuracy of 96.93% when training from scratch. On the other hand, the DenseNet121 architecture achieved the best performance in all learning methods with an accuracy of 99.48% when using transfer learning.

Furthermore, we demonstrate that noisy data can decrease the recognition performance of the CNN architectures. This experiment then applied four noisy data augmentation techniques while training the CNN model using the transfer learning method. It clearly showed that the accuracy of DenseNet121 was slightly decreased from 99.48% to 99.28% when training with noisy images. Subsequently, we proposed the DeblurGAN-CNN architecture to address the problems of noisy images. The result of the DeblurGANs is shown in the Section 3.5.2.

CNN Models	Learning Methods					
	SL		TL		TL-nDA	
	5-cv	Test	5-cv	Test	5-cv	Test
VGG19	96.51±0.76	96.93	**99.34±0.23**	98.81	92.72±7.39	98.45
InceptionResNet	**98.63±0.31**	98.15	99.05±0.19	98.61	92.79±7.40	98.38
MobileNetV2	97.10±0.78	97.10	99.13±0.21	98.96	93.97±6.51	98.93
DenseNet121	98.61±0.32	**98.41**	99.27±0.11	**99.48**	**95.41±5.06**	**99.28**

Table 4 Recognition performances (mean validation accuracy: 5-cv, standard deviation, and test accuracy) of four CNN models: VGG19, InceptionResNet, MobileNetV2, and DenseNet121, using different learning methods (SL, TL, and TL-nDA) on the THI-C68 dataset.

3.5.1.2 COMPARISON of the CNNs and other Studies

According to previous experiments, we selected two CNN architectures, DenseNet121 and MobileNetV2. In this study, two CNN architectures were used and hand-crafted feature extraction combined with machine learning, namely SiftD-SVM (Surinta, Karaaba, et al. 2015) and HOGFoDRs-SVM (Inkeaw et al. 2019), were evaluated and compared on the THI-C68 dataset.

To consider a fair comparison between CNN architectures and previous studies, we provided two shuffled random subsets of the THI-C68 dataset according to the experiments of Surinta et al. (Surinta, Karaaba, et al. 2015) and Inkeaw et al. (Inkeaw et al. 2019). The first subset (Set-I) had 11,592 training samples and 2,898 test samples. The second subset (Set-II) had 13,041 training samples and 1,449 test samples. Note that, Set-I and Set-II were compared with the HOGFoDRS-SVM and the SiftD-SVM methods.

The results reported in Table 5 show that the DenseNet121 architecture with transfer learning (DenseNet121-TL) outperformed every CNN architecture on both sets with 5-fold cross-validation. Consequently, DenseNet121-TL outperformed the HOGFoDRs-SVM method by 0.51% on Set-I and outperformed the SiftD-SVM method by 0.37%. Also, MobileNetV2 with transfer learning (MobileNetV2-TL) achieved the highest performance on the independent test set of Set-II with 99.31% accuracy. MobileNetV2-TL significantly outperformed the SiftD-SVM method by 4.97%.

Methods	Set-I		Set-II	
	5-cv	Test	10-cv	Test
SiftD-SVM (Surinta et al. 2015)	-	-	98.93 ± 0.03	94.34
HOGFoDRs-SVM (Inkeaw et al. 2019)	98.76	-	-	-
MobileNetV2-TL	99.13 ± 0.21	98.96	99.16 ± 0.31	**99.31**
DenseNet-TL	**99.27 ± 0.11**	**99.48**	**99.30 ± 0.36**	99.03
MobileNetV2-TL-nDA	93.97 ± 6.51	98.93	94.69 ± 7.62	99.10
DenseNet-TL-nDA	95.41 ± 5.06	99.28	95.44 ± 6.88	99.17

Table 5 The performance (mean validation accuracy: 5-cv, standard deviation, and test accuracy) comparison of the CNN models using different learning methods with other studies on the THI-C68 dataset.

Table 5 The performance (mean validation accuracy: 5-cv, standard deviation, and test accuracy) comparison of the CNN models using different learning methods with other studies on the THI-C68 dataset

From the results above, the CNN architectures with transfer learning impact improving the performance of handwritten character recognition. Consequently, the

CNN models achieved better accuracy than the hand-crafted features (Inkeaw et al. 2019; Surinta et al. 2015) on the THI-C68 dataset.

3.5.2 Denoising Performance of DeblurGAN on The n-THI-C68 Dataset

In this experiment, the input images were the noisy images of the n-THI-C68 dataset with 128×128 pixels. We first reconstructed the denoise character images with 128×128 pixels using Wasserstein and content loss functions. The hyperparameters of DeblurGAN were applied as follows: the optimization algorithm is Adam, learning rate = 0.0001, momentum = 0.9 and 0.999, training epochs = 200, and batch size = 32.

To study the reconstruction quality of the denoise images, we evaluated the DeblurGAN architecture with two well-known image quality metrics called the peak signal to noise ratio (PSNR) and the structural similarity index (SSIM) on the n-THI-C68 dataset. The noise images with different noise methods and reconstructed images are shown in Figure 14. We reported the PSNR and SSIM values obtained when evaluating the different noise methods. High PSNR and SSIM values represent better accuracy and reconstruction of the image, respectively. We achieved the best PSNR and SSIM when using DeblurGAN to reconstruct the character images from noisy images of the low contrast, low resolution, and AWGN, respectively. However, motion blur and mixed noise were the most difficult to reconstruct.

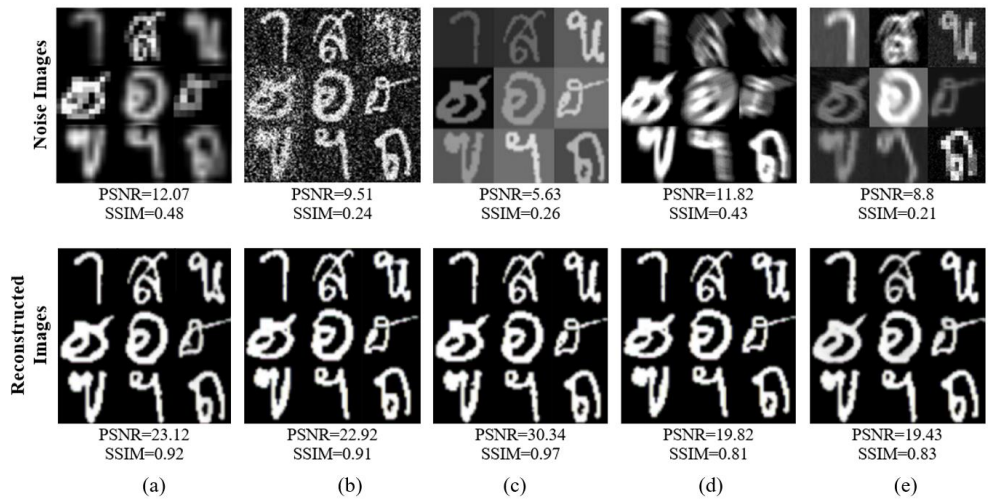


Figure 14 Illustration of the noisy images of (a) low resolution, (b) AWGN, (c) low contrast, (d) motion blur, and (e) mixed noise, as shown in the first row and reconstructed images using DeblurGAN architecture, as shown in the second row. Note that the high PSNR value presents better performance accuracy, and the high SSIM value presents the most similar character images between the reconstructed and original images.

The DeblurGAN architecture adds the residual blocks and global skip connection in the generator, making the DeblurGAN only learn a residual correction to transform the noisy images. The DeblurGAN could be more generalized in reconstructing the denoise images generated by multiple generations or from the

unknown kernel. Importantly, the DeblurGAN (Kupyn et al. 2018) uses the WGAN-GP and perceptual loss when reconstructing denoise images, while the traditional neural networks use L1 and L2 optimization algorithms when reconstructing denoise images.

3.5.3 Denoising Performance of DeblurGAN on the n-THI-C68 Dataset

This section presents the DeblurGAN-CNN architectures to perform on the n-THI-C68 dataset. In response to the experimental results, as shown in Section A, we selected two CNN architectures, DenseNet121 and MobileNetV2, as the CNN models. Hence, we connected DeblurGAN with CNN architecture, called DeblurGAN-DenseNet121 and DeblurGAN-MobileNetV2. Consequently, we compared the DeblurGAN-CNN architectures with the traditional CNN architectures to recognize the noisy character images, as shown in Table IV.

Noise Methods	CNN Architectures				DeblurGAN-CNN Architectures	
	MobileNetV2-TL	DenseNet121-TL	MobileNetV2-TL-nDA	DenseNet121-TL-nDA	DeblurGAN-MobileNetV2	DeblurGAN-DenseNet121
Low Resolution	77.24	49.90	93.02	93.96	98.48	98.52
AWGN	27.92	16.63	96.72	98.21	98.72	99.03
Low Contrast	13.80	31.30	95.62	93.51	99.28	99.41
Motion Blur	45.89	49.59	91.75	93.06	97.96	97.69
Mixed Noise	30.78	25.02	93.93	92.89	97.90	98.00
Overall	39.13	34.49	94.21	94.33	98.47	98.53

Table 6 The performance of the CNN architectures and DeblurGAN-CNN architectures on the n-THI-C68 dataset.

Methods	Memory (MB)	Parameters (M)	Train Times (hour:min)	Test Times (sec/image)	Accuracy (% diff)
VGG19	544	142.2	2:15	0.0034	-4.39
InceptionResNet	210	54.4	2:04	0.0043	-4.42
MobileNetV2	19.2	2.35	0:46	0.0006	-4.32
DenseNet121	57.8	7.02	1:17	0.0023	-4.20
DeblurGAN-MobileNetV2	62.9	13.35	1:27	0.0021	-0.06
DeblurGAN-DenseNet121	101.5	18.51	2:04	0.0032	0.00
DeblurGAN (standalone)	43.7	11.34	0:32	0.0016	-

Table 7 The configuration detail, computation times and differential accuracy of different CNNs and DeblurGAN-CNNs

Table 6 shows that the CNN architecture achieved low accuracy when using MobileNetV2-TL. It attained 77.24% accuracy when recognizing the noisy images with low resolution. The worst performance of only 13.80% accuracy was achieved when recognizing low-contrast images. However, we found that when training the

CNN model using transfer learning with noisy data augmentation techniques (TL-nDA), the accuracy increased from only 13.80% to 95.62% when using MobileNetV2-TL-nDA. The overall performance accuracy of MobileNetV2-TL-nDA and DenseNet121-TL-nDA was 94.21% and 94.33% respectively.

The results show that the DeblurGAN-CNN architectures could address the problems of noisy character images by achieving higher performance above 97% accuracy on all noise methods. Subsequently, the DeblurGAN-DenseNet121 achieved 98.53% accuracy and slightly outperformed the DeblurGAN-MobileNetV2 that achieved an accuracy of 98.47%. Moreover, the DeblurGAN-CNN architectures significantly outperformed the DenseNet121-TL-nDA and MobileNetV2-TL-nDA (The result was significant at $p < .05$). The misclassified characters are shown in Figure 15.

We concluded that only training the CNN models using the transfer learning with noisy data augmentation techniques could achieve accuracy above 90% on the n-THI-C68 dataset, although, very high accuracy is required in the handwritten character tasks to reduce the error while using the output data. Importantly, we recommend using the DeblurGAN-CNN architectures as this study yielded promising and outstanding results. The results show that the DeblurGAN-CNN architectures could address the problems of the noisy character images by achieving higher performance above 97% accuracy on all noise methods. Subsequently, the DeblurGAN-DenseNet121 achieved 98.53% accuracy and slightly outperformed the DeblurGAN-MobileNetV2 that achieved an accuracy of 98.47%. Moreover, the DeblurGAN-CNN architectures significantly outperformed the DenseNet121-TL-nDA and MobileNetV2-TL-nDA (The result was significant at $p < .05$). The misclassified characters are shown in Figure 15.

The proposed model was evaluated on a noisy dataset compared to the four CNN architectures and has achieved better recognition performance with an approximately 4% increase in accuracy. Moreover, Table 7 summarizes different compared models, including memory sizes, number of parameters, training and testing times, and the difference in accuracy from our best model, DeblurGAN-DenseNet121. In terms of time, DeblurGAN- DenseNet121 had slightly more training time than VGG19 and about the same amount of time as InceptionResNetV2. However, it requires 2.7 and 1.6 times more than MobileNetV2 and DenseNet121, respectively, which are lightweight CNN architectures.

Another proposed model, the DeblurGAN-MobileNetV2, was better speed training and was 1.9 and 1.1 times faster than MobileNetV2 and DenseNet121, respectively, However, it is still efficient in recognition performance by over 4%. In addition, Our proposed methods increase the number of parameters, since DeblurGAN has 11.34M and was training time to learn denoise images about 32 min.

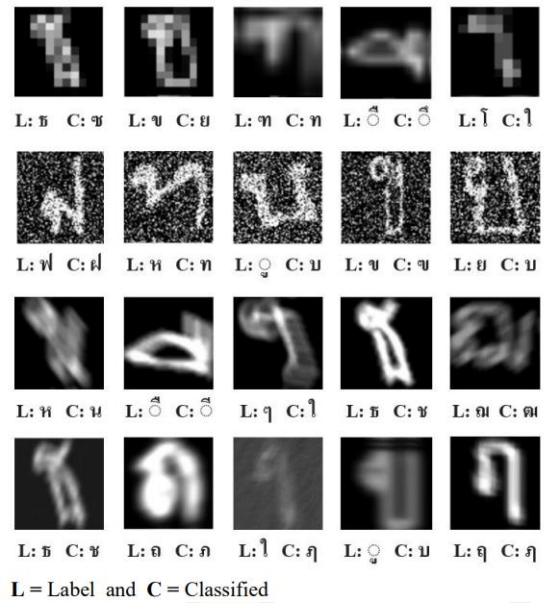


Figure 15 Illustration of misclassified characters on the test set of the n-THI-C68 dataset using DeblurGAN-CNN.

3.5.4 Comparison of the DeblurGAN-CNN Architecture and Other Approaches

We selected two DeblurGAN-CNN architectures: DeblurGAN-MobileNetV2 and DeblurGAN-DenseNet121, to evaluate generalization ability on the other noisy datasets n-MNIST and THCC-67. Comparisons of results on the n-MNIST and THCC-67 datasets with the GAN-CNNs and other approaches are presented in Table 8 and Table 9.

Table 8 compares the results between the proposed DeblurGAN-CNN architectures and other approaches on the n-MNIST dataset. As a result, the accuracy of the DeblurGAN-MobileNetV2 slightly outperformed the DeblurGAN-DenseNet121. The DeblurGAN-MobileNetV2 achieved the best accuracy on the n-MNIST dataset using AWGN and AWGN+Contrast noise methods.

The experimental results on the n-MNIST dataset showed that the optimal CNN-Hopfield network achieved an accuracy of 99.18%, 99.74%, and 97.53% when the AWGN, motion blur, and AWGN+Contrast noises were applied, respectively.

On the other hand, the DeblurGAN-MobileNetV2 achieved 98.93%, 99.36%, and 97.59% accuracies when applying the AWGN, motion blur, and AWGN+Contrast noises, respectively. Further, the DeblurGAN-MobileNetV2 architecture outperformed the optimal CNN–Hopfield network on the n-MNIST dataset when applying AWGN+Contrast noise.

Methods	Noise Methods		
	AWGN	Motion Blur	AWGN+ Contrast
PQ-DBN (Basu et al. 2015)	90.07	97.40	92.16
Dropconnect DBN (Karki et al. 2018)	97.57	97.20	96.93
PixelCNN PQ-DBN (Karki et al. 2018)	97.62	97.20	95.04
PCGAN-CHAR (Liu et al. 2019)	98.43	99.20	97.25
Optimal CNN-Hopfield Network (Keddous and Nakib 2022)	**99.18**	**99.74**	97.53
DeblurGAN-MobileNetV2 (Proposed method)	98.93	99.36	**97.59**
DeblurGAN-DenseNet (Proposed method)	98.89	99.40	97.51

Table 8 The performance comparison of DeblurGAN-CNN architectures with other approaches on the n-MNIST dataset.

Methods	Accuracy
HOGFoDRs-SVM (Inkeaw et al. 2019)	70.74
DeblurGAN-MobileNetV2 (Proposed method)	80.63
DeblurGAN- DenseNet121 (Proposed method)	**80.68**

Table 9 The performance comparison of DeblurGAN-CNN architectures with the HOGFoDRs-SVM method on the THCC-67 dataset.

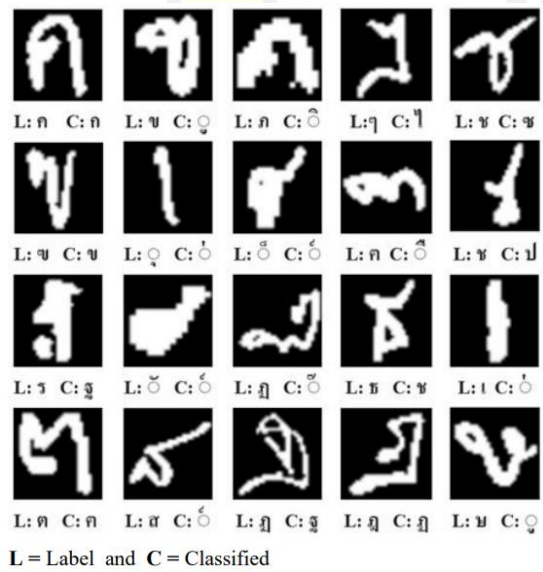


Figure 16 Illustration of the misclassified characters on the THCC67 dataset using DeblurGAN-DenseNet121

3.6 Discussion

We observed the training loss between the DenseNet121-TL-nDA and DeblurGAN-DenseNet121, as shown in Figures 17(a) and 17(b). The improvement of validation loss is shown in Figure 17(c). It can be seen that the training loss of the DeblurGAN-DenseNet121 is relatively low in the early epochs due to the transferring

of pre-trained weights. The training loss of the DeblurGAN-DenseNet121 is always lower than the DenseNet121-TL-nDA.

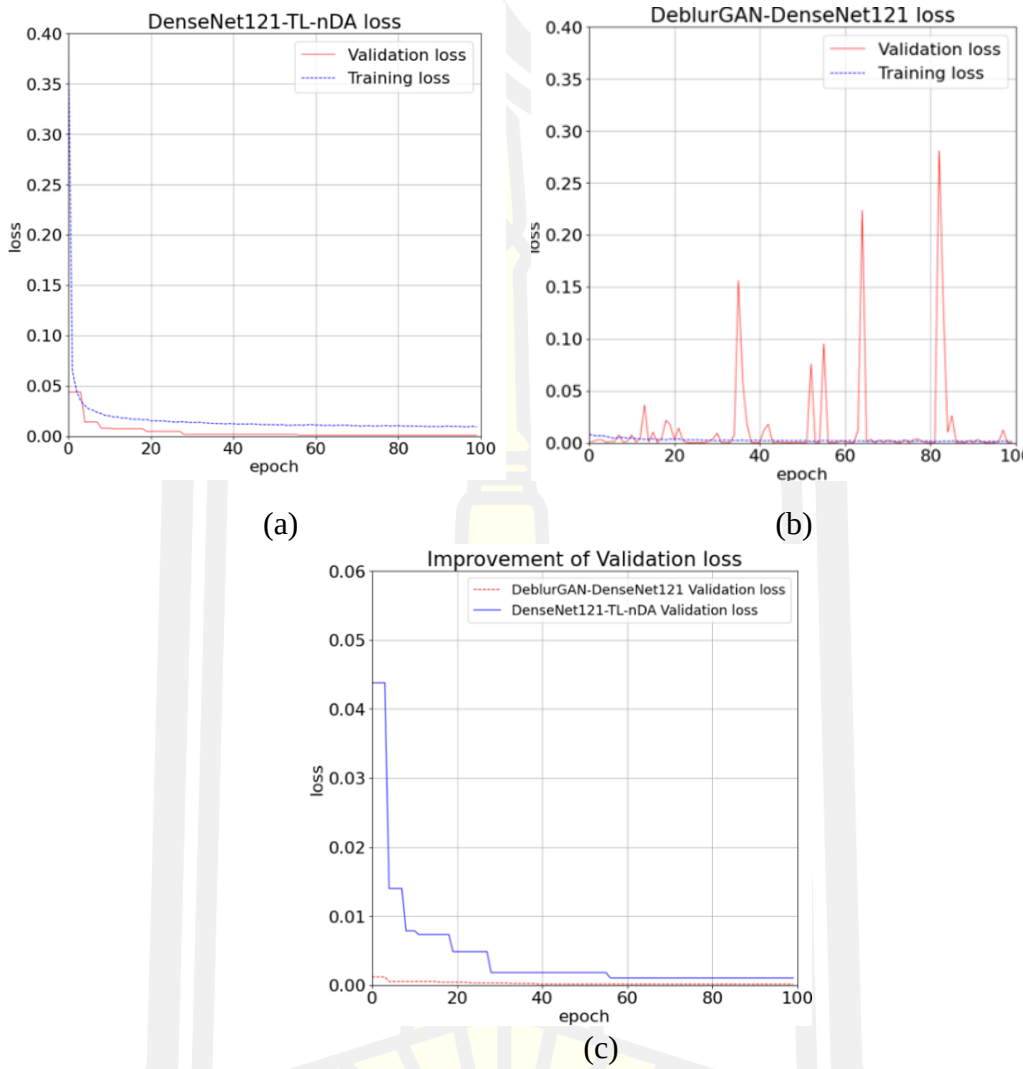


Figure 17 Illustration of the validation and training loss (a) DenseNet121-TL-nDA (b) DeblurGAN-DenseNet121 and (c) comparison of improving in validation loss.

As shown in Figure 18, we found that the DenseNet121 model with TL (DenseNet121-TL) achieved unsatisfactory performance when evaluated on the noisy images. The accuracy of DenseNet121-TL quickly dropped when the PSNR value was increased. The result shows that DenseNet121-TL-nDA obtained much better performance than DenseNet121-TL. However, the accuracy of DenseNet121-TL-nDA was quickly decreased when the PSNR value was higher than 20. Furthermore, the DeblurGAN-DenseNet121, when training using TL-nDA methods, achieved high accuracy even when the PSNR value was increased more to than 26, with an accuracy above 90%.

We also discussed in-depth the proposed DeblurGAN-CNN architecture and the optimal CNN-Hopfield network on the n-MNIST dataset in terms of accuracy. Therefore, the optimal CNN-Hopfield network (Keddous and Nakib 2022) outperformed our proposed architecture because the optimal CNN-Hopfield network is an ensemble method that combines many CNN outputs to achieve better recognition. The ensemble method has been reported to guarantees better accuracy in much published research (Gonwirat and Surinta 2021; Guo et al. 2019; Noppitak and Surinta 2022). On the other hand, the DeblurGAN-CNN architecture is a deep learning architecture that combines GAN and CNN architectures. So, only one output is recognized from the proposed architecture. Consequently, the optimal CNN-Hopfield network achieved an accuracy of 62%, 92%, and 97.52% when recognized using one, two, and three CNN models. In comparison, our proposed method achieved an accuracy of 98.93% using only one model and given an accuracy higher than 6% compared to the optimal CNN-Hopfield network that uses three CNN models.

Furthermore, finding texts that appear in natural scene images are challenging. To solve the challenge, object and scene text detection in the wild should be first applied to obtain the region of interest, which is the area of texts. Second, we could employ the DeblurGAN-CNN method to denoise and recognize the text in the natural scene images. This solution could be enhanced the recognition performance. We will concentrate on finding and recognizing text that appears in natural scene images for future work.

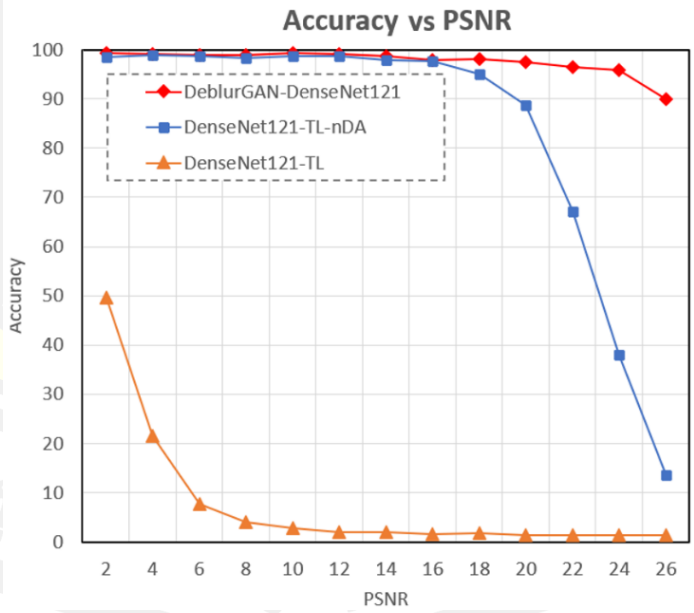


Figure 18 The effectiveness of different denoise architectures proposed to recognize the noisy character images on the n-THI-C68 dataset.

3.7 Conclusion

The performance of the handwritten character recognition systems decreases in consequence of many problems, such as handwriting styles, degradation of the documents, and noise appearance while transforming documents into a digital format. This research mainly focused on the denoise and recognition of noisy handwritten character images. Consequently, the robust generative adversarial network (GAN) combined with the convolutional neural network (CNN) architecture, called DeblurGAN-CNN, was proposed to synthesize new clean handwritten characters from noisy handwritten characters and recognition with improved handwritten character performance. For the CNN architecture, we combined two state-of-the-art CNNs: MobileNetV2 and DenseNet121, with the DeblurGAN, called DeblurGANMobileNetV2 and DeblurGAN-DenseNet121. The DeblurGAN-CNN architectures were trained using the transfer learning technique and applying the noisy data augmentation techniques to create a robust model. The most beneficial aspect of the DeblurGAN-CNN models was that they could learn and generalize from many noisy methods, including low resolution, additive white Gaussian noise (AWGN), low contrast, motion blur, and mixed noise. To evaluate the denoise model, the DeblurGAN produced significant output that achieved a high peak signal to noise ratio (PSNR) and structural similarity index (SSIM) values. As a result, the DeblurGAN architecture could remove various noises from the noisy handwritten character images. For the accuracy performance, the results show that the DeblurGANCNN architectures generated strong handwritten character images and achieved the highest performance on the n-MNIST and n-THI-C68 datasets when compared with other existing methods. Also, both DeblurGAN-DenseNet121 and DeblurGAN-MobileNetV2 presented significant performance and outperformed the HOGFoDRs-SVM on the THI-C68 and THCC-67 datasets. The DeblurGAN-CNN architectures achieved an accuracy above 98%, 97.59%, and 80.68% on the n-THI-C68, n-MNIST, and THCC-67 datasets. Subsequently, the DeblurGAN-CNN architectures, which used the DenseNet121 and MobileNetV2 as the CNN architectures, achieved high handwritten character recognition performance with and without noisy handwritten characters.

Chapter 4

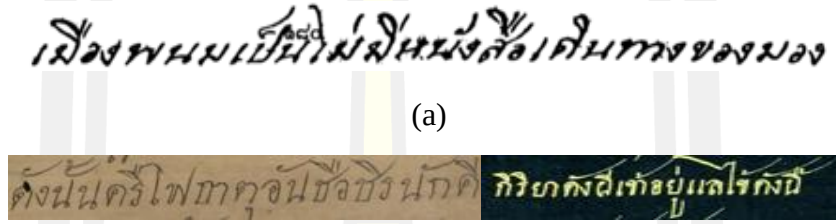
Efficient Data Augmentation Strategy on CRNN for Handwritten Text Recognition

Predicting the sequence pattern of the handwritten text images is a challenging problem due to the various writing style, insufficient training data, and even background and noise appearing in the text images. The architecture of the combination between convolutional neural network (CNN) and recurrent neural network (RNN), called CRNN architecture, is the most successful sequence learning method for handwritten text recognition systems. For handwritten text recognition in historical Thai document images, in this paper, we first trained nine different CRNN architectures with both training from scratch and transfer learning techniques to find out the most powerful technique. We discovered that the transfer learning technique does not significantly outperform scratch learning. Second, we examined training the CRNN model by applying the basic transformation data augmentation techniques: shifting, rotation, and shearing. Indeed, the data augmentation techniques provided more accurate performance than without applying data augmentation techniques. However, it did not show significant results. The original training strategy aims to find the global minima value and not always solve the overfitting problems. Third, we proposed a cyclical data augmentation strategy, called CycleAugment, to discover many local minima values and prevent overfitting. In each cycle, it rapidly decreased the training loss to reach the local minima. The CycleAugment strategy allowed the CRNN model to learn the input images with and without applying data augmentation techniques to learn from many input patterns. Hence, the CycleAugment strategy consistently achieves the best performance when compared with other strategies. Finally, we prevented image distortion by applying a simple technique to the short word images and achieved better performance on the historical Thai document image dataset.

4.1 Introduction

The offline text recognition system is a vision-based application that automates extracting information from handwritten and printed manuscripts and transforms images into digitally readable text that could be editable and comfortable to store and retrieve. Earlier, various research focused on character recognition which recognized isolated characters (Surinta et al. 2015; Inkeaw et al. 2019; Kavitha and Srimathi 2019; Wang et al. 2019). However, a few research concentrates on the recognition of handwritten text. This is because it takes more effort to segment handwritten text into individual characters (Lue et al. 2010; Choudhary et al. 2013; Inkeaw et al. 2018). Due to messy handwriting, various writing styles, and cursive texts, as shown in Figure 19, it is difficult to solve by segmenting characters and then

recognizing them by traditional optical character recognition (OCR). Character sequence learning is more suitable for word recognition (Giménez and Juan 2009; Bluche et al. 2013). Hence, an effective feature-based sliding window and sequence learning methods are applied to recognize each character and then transcript to words (Wang et al. 2011; Lee et al. 2014; Mishra et al. 2016). However, handwritten text recognition (HTR) methods mainly focus on word recognition and have become a more famous research domain nowadays.



(a)

(b) (c)

Figure 19 Examples of historical Thai handwritten texts from (a) Thai archive, (b) Phra Narai Medicine, and (c) King Rama V, Volume 1, Medicine Manuscripts

Deep learning methods have become the principal method in various computer vision applications, such as object detection, object recognition, speech recognition, and natural language processing. Further, convolutional neural networks (CNN) architectures, one of the deep learning methods, are widely proposed for feature extraction and image classification. The CNN also proposed to address the challenge of word recognition (Ameryan and Schomaker 2021; Chen et al. 2021; Singh et al. 2021). In addition, CNN and recurrent neural networks (RNNs), which are the famous sequence learner architectures, were proposed to recognize the printed and handwritten words (Ameryan and Schomaker 2021; Chen et al. 2021) and achieved a high accuracy performance. Consequently, state-of-the-art in handwritten character recognition is a combination of CNN and RNN, called convolutional recurrent network (CRNN). The CRNN also proposed solving problems in many text recognition fields such as scene text and video subtitle recognition.

Moreover, handwritten text recognition has been applied in many languages, such as English, Chinese, Arabic, Indian, and Amharic (Sujatha and Bhaskari 2019; Yan and Xu 2020; Abdurahman et al. 2021; Ameryan and Schomaker 2021; Butt et al. 2021; Singh et al. 2021). Particularly, historical manuscripts are faced with cursive writing, noisy background, and differing word spelling from ancient and insufficient lexicon for transcription. The challenge of Thai handwritten character recognition is that the Thai language does not have an exact rule to split the sentences and no space between words. For explicit prediction, it is demanding to segment sentences into tokenized words. Further, a few research studies and a comprehensive investigation on Thai word recognition.

The main contribution of this paper is to present the new data augmentation strategy, namely CycleAugment. The proposed data augmentation strategy mainly focuses on minimizing the validation loss and avoiding overfitting. We achieve our goal with a simplistic strategy and implementation. Our research is motivated by Huang et al. (Huang et al. 2017), who proposed the cyclic cosine annealing method that calculated the learning rate in every epoch and then started the new learning rate at the beginning of a new cycle.

Furthermore, training the CRNN model usually allows choosing only to train the CRNN model with or without applying data augmentation techniques. We offer the CycleAugment strategy that provides to train the CRNN model with and without applying data augmentation techniques simultaneously. Importantly, our CycleAugment strategy confirms that it can handle every CRNN architecture.

We evaluate the efficiency of the CycleAugment strategy on several CRNN architectures for handwritten word recognition on Thai archive manuscripts. To show the importance of the CycleAugment strategy, we have compared them to the original data augmentation strategy. The results show that the CycleAugment strategy significantly decreases the character error rate (CER). The CycleAugment strategy achieves the CER value of 5.43 and the original data augmentation strategy obtains the CER value of 7.31 on the Thai archive manuscript.

The remainder of this paper is organized as follows. The related work is briefly described in Section 2. Section 3 deeply explains the proposed CRNN architecture and proposed CycleAugment strategy. Section 4, present the Thai historical document dataset, training strategy, and experimental evaluation. The discussion is presented in Section 5. Finally, the last section gives the conclusion and future direction

4.2 Related work

In this section, we survey the HTR task based on deep learning techniques. We also study the transfer learning and data augmentation techniques that improve the performance of deep learning.

4.2.1 Handwritten text recognition

This Researchers have been proposing text recognition systems for several applications, such as scene text recognition (Shi et al. 2017; Shi et al. 2018; Luo et al. 2019; Chen et al. 2021), video subtitle recognition (Xu et al. 2018; Yan and Xu 2020), and handwritten text recognition in many languages (Abdurahman et al. 2021; Ameryan and Schomaker 2021; Butt et al. 2021). Currently, most of the proposed HTR methods are based on the CNNs and RNNs architectures.

For HTR, (Abdurahman et al. (2021) proposed a convolutional recurrent neural network architecture, called AHWR-Net, to recognize Amharic words. The

AHWR-Net architecture was divided into feature extraction, sequence modeling, and classification. First, they created a CNN model and compared their proposed CNN model with state-of-the-art CNN models: DenseNet-121, ResNet-50, and VGG-19. These CNN models were also proposed to extract the feature from the Amharic word images. Second, the RNN architecture was proposed as the sequence model to train spatial features extracted from the previous step. Finally, the probability distributions, which was the output of the RNN method, were classified using a connectionist temporal classification algorithm (CTC). In addition, Butt et al. (2021) built a robust Arabic text recognition system using the CNN-RNN attention model from natural scene images. Their Arabic text recognition system addressed the challenge with the texts in different sizes, fonts, colors, orientation, and brightness.

Furthermore, Ameryan and Schomaker (2021) proposed a high-performance word classification using homogeneous CNN and long short-term memory (LSTM) networks. First, for the CNN model, they created five CNN layers. Each CNN layer contained a convolutional layer, normalization method, nonlinear rectified linear unit (ReLU), and max-pooling layer. Second, for the LSTM, the bidirectional-LSTMs with three layers were used. Third, the CTC decoding was attached as the output of their network. Finally, the invented ensemble system was proposed, which included five networks. The outputs of each network were sent to vote using the plurality vote method.

4.2.2 Thai handwritten text recognition

There is a small amount of research that focuses on Thai handwritten text recognition. In 2019, Chamchong et al. proposed hybrid deep neural networks that combined three convolutional layers and two bidirectional gated recurrent unit (BiGRU) layers, namely 3CNN+BiGRU. The 3CNN+BiGRU was followed by softmax and CTC loss functions. For a computational time, both bidirectional LSTM (BiLSTM) and BiGRU were compared. The result of the computational time showed that the BiGRU is faster than BiLSTM. Therefore, the 3CNN+BiGRU showed the best character error rate (CER) of 12.1% on the Thai archive dataset when time step and RNN size were set as 32 and 128.

In 2020, Srinilta and Chatpoch proposed a deep learning method for multi-task learning on three scripts: Thai, Devanagari, and Latin. The deep learning method was divided into three main layers: CNN, RNN, and CTC. First, the CNN layer was trained based on ResNet50 architecture for multi-task learning, followed by the BiGRU layer. Second, the output of the BiGRU layer in the first step was trained using different BiGRU layers and then followed by the CTC layer. For example, the BiGRU-Thai layer was aimed to train and recognize only Thai scripts.

In 2021, Chamchong et al. created four CNN layers: convolutional, ReLU activation, max-pooling, and dropout layers. Then, two BiGRU and dropout layers were added to the last CNN layer. In their method, the dropout layers were set as 0.2. Furthermore, to decrease the training loss value, they compared based on two

optimization algorithms: SGD and RMSprop. The experimental results showed that the RMSprop optimization outperformed the SGD optimization algorithm on the standard Thai handwritten dataset.

4.2.3 Improve the deep learning performance with transfer learning and data augmentation techniques

In deep learning, achieving high performance is generally accompanied by various convolutional layers and training images (Wu et al. 2017) The very deep convolutional layers and limited training samples often attend to the overfitting problem (Thanapol et al. 2020). It directly affects the deep learning model that makes it hard to generalize new samples. Transfer learning, data augmentation, dropout, and reducing the complexity of the deep learning architecture are suggested to address the overfitting problem (Pawara et al. 2017; Enkvetchakul and Surinta 2021; Gonwirat and Surinta 2020).

Gonwirat & Surinta (Chapter 2) trained the CNN models (including Inception-ResNetV2 and VGG19 architectures) based on two training methods: scratch learning and transfer learning. The comparison results showed that the transfer achieved high accuracy when evaluated using 5-fold cross-validation (5-cv) and 10-cv methods. As a result, the VGG19 architecture, which included 19 layers and was designed as a stacked network, outperformed Inception-ResNetV2 when training with transfer learning on the THI-C68 dataset. It was also improving the recognition speed.

Pawara et al. (2017) applied six data augmentation techniques: rotation, blur, scaling, contrast, illumination, and projective to the original image. In their method, first, the training examples increased 10 to 25 times larger than the original data. Second, they trained the CNN models with original and augmentation images, called offline training strategy. Moreover, Enkvetchakul and Surinta (2021) trained the CNN models using three training strategies: offline, online, and mixed training. Six data augmentation techniques were applied with an online training strategy: width and height shift, rotation, zoom, brightness, cutout, and mixup while training the CNN model. The online training strategy was much faster than training with offline strategy. Because it did not increase the number of training examples, however, it transformed the original image using augmentation techniques while training. Hence, the CNN models could learn from the new images in each epoch.

4.3. The convolutional recurrent neural network

In this section, we present the convolutional recurrent neural network (CRNN) framework for Thai handwritten text recognition of historical document images with a new data augmentation strategy. Firstly, convolutional neural networks (CNNs) are described. Secondly, two recurrent neural networks (RNNs) (long short-term memory and gated recurrent unit) are briefly detailed. Thirdly, detail of the connectionist temporal classification (CTC) decoding is presented for the evaluation metric. Finally,

the proposed cyclical data augmentation strategy, namely CycleAugment, is presented. The proposed framework is explained as follows.

4.3.1 Overview of the CRNN architecture

The CRNN network is illustrated in Figure 20. The CRNN has only one input. Our framework also supports both images of a group of words and short words as for the input. In the CNN architecture, we propose eight different CNN architectures to find the best base CNN model. For the RNN network, we propose two layers of bidirectional RNN networks and connect them to the CNN architecture. Hence, the outputs of the bidirectional RNN network are then classified using the softmax function. The output of the CRNN is a matrix containing character probabilities for each time step. Further, the CTC decoding is attached at the last layer to decode the probability of characters to make the final text output. Our framework can predict a maximum of 94 members in total, including characters, numbers, and blank (space). The configurations of all CRNN architectures are shown in Table 10.

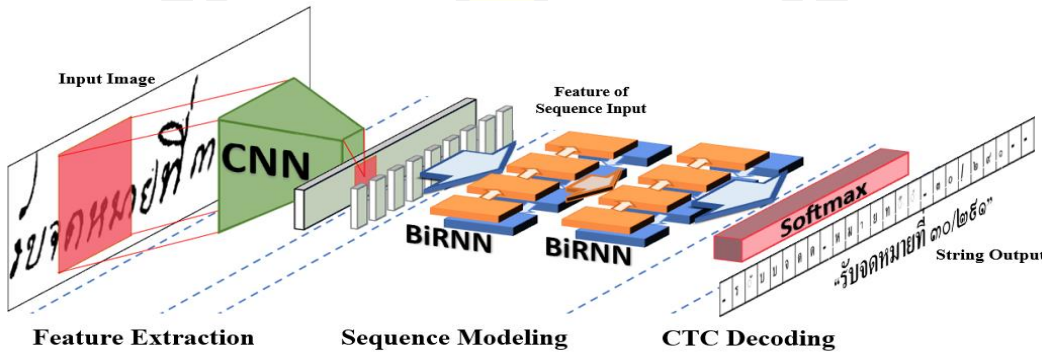


Figure 20 Overview framework of convolutional recurrent neural networks

CCNet	mCCNet-64	mCCNet-512	mVGG16	mVGG19	mResNet50	mDenseNet-121	mMobileNet-V2	mEfficientNet-B1
6 weighted layers	7 weighted layers	7 weighted layers	14 weighted layers	16 weighted layers	26 weighted layers	43 weighted layers	23 weighted layers	29 weighted layers
Input image (64, 504, 1)					Input image (64, 504, 3)			
Conv3-16 Maxpool2-s2	Conv3-16 Maxpool2-s2	Conv3-16 Maxpool2-s2	Conv3-16 Conv3-16 Maxpool2-s2	Conv3-64 Conv3-64 Maxpool2-s2	Conv7-64 Maxpool3-s2	Conv7-64 Maxpool3-s2	Conv3-32-s2 DwConv3-32	Conv3-32-s2 DwConv3-32
Conv3-32 Maxpool2-s2	Conv3-32 Maxpool2-s2	Conv3-32 Maxpool2-s2	Conv3-128 Conv3-128 Maxpool2-s2	Conv3-128 Conv3-128 Maxpool2-s2	$\begin{bmatrix}\text{Conv1}-64\\ \text{Conv3}-64\\ \text{Conv1}-256\end{bmatrix}$ x3	$\begin{bmatrix}\text{Conv1}-128\\ \text{Conv3}-32\end{bmatrix}$ x6 Conv1-128 Avgpool2-s2	Conv1-16 Conv1-96 DwConv3-96	Conv1-16 Conv1-96 DwConv3-96 SE
Conv3-32 Maxpool2-s2	Conv3-32 Maxpool2-s2	Conv3-32 Maxpool2-s2	Conv3-256 Conv3-256 Conv3-256 Maxpool2-s2	Conv3-256 Conv3-256 Conv3-256 Conv3-256 Maxpool2-s2	$\begin{bmatrix}\text{Conv1}-128\\ \text{Conv3}-128\\ \text{Conv1}-512\end{bmatrix}$ x4	$\begin{bmatrix}\text{Conv1}-128\\ \text{Conv3}-32\end{bmatrix}$ x12 Conv1-512	$\begin{bmatrix}\text{Conv1}-24\\ \text{Conv1}-144\\ \text{DwConv3}-144\end{bmatrix}$ x2	$\begin{bmatrix}\text{Conv1}-24\\ \text{Conv1}-144\\ \text{DwConv3}-144\\ \text{SE}\end{bmatrix}$ x2
-	-	-	Conv3-512 Conv3-512 Conv3-512	Conv3-512 Conv3-512 Conv3-512 Conv3-512 Conv3-512	-	-	$\begin{bmatrix}\text{Conv1}-32\\ \text{Conv1}-192\\ \text{DwConv3}-192\end{bmatrix}$ x3	$\begin{bmatrix}\text{Conv1}-40\\ \text{Conv1}-240\\ \text{DwConv5}-240\\ \text{SE}\end{bmatrix}$ x2
-	Conv1-64	Conv1-512	Conv1x1-512					
Global average pooling								
Bidirectional RNN-(N)								
Bidirectional RNN-(N)								
FC, Softmax (94)								
CTC decoding								

Table 10 Configuration details of CRNN architectures

Furthermore, we propose the cyclical data augmentation strategy (CycleAugment). The CycleAugment strategy provides the CRNN model to train handwritten text images concurrently with and without applying data augmentation techniques. The CycleAugment is a powerful strategy for obtaining various local optimal loss values in each cycle until they reach a minimum value at the end of training.

4.3.2 Convolutional neural network

In recent years, many CNN architectures have been proposed to enhance the performance of image classification. This section describes the configuration of the different CNN architectures evaluated in this paper to solve handwritten text recognition. We first explained the CNN architecture proposed by Chamchong et al. (2019), especially for handwritten text recognition, called CCNet. Second, we modified CCNet (mCCNet) by adding a 1x1 convolutional layer (Conv1) that mainly reduced the parameters of model CC. Finally, we modified state-of-the-art CNN architectures, including VGG16 and VGG19, ResNet50, DenseNet121, MobileNetV2, and EfficientNetB1.

4.3.2.1 CCNet

Chamchong et al. (2019) proposed simple CNN that includes three blocks of convolutional and max-pooling layers. Each block contained a convolutional layer with 3x3 kernel sizes (Conv3), a max-pooling layer using 2x2 kernel sizes (Maxpool2), and a stride of 2 (s2), in order. The convolutional layers in the first, second, and third blocks were 16, 32, and 32 feature maps. As well as the ReLU activation function and batch normalization (BN) were added to the last block.

4.3.2.2 Modified CCNets

We modified CCNet (mCCNet) by attaching Conv1x1 to reduce the CNN parameters and introduce new nonlinearity into the network. For the mCCNet-64 and mCCNet-512 models, we implemented the Conv1 layer with feature map sizes of 64 and 512, respectively.

4.3.2.3 Modified VGGs

The modified VGG16 (mVGG16) and VGG19 (mVGG19) were built based on the VGG16 and VGG19 (Simonyan and Zisserman 2015), respectively. These networks comprised a convolutional layer with 3x3 kernel sizes and followed by a max-pooling layer using 2x2 kernel sizes. However, the mVGG16 and mVGG19 were cut at the end of the fourth block and we also replaced them with Conv1 of 512 feature maps.

4.3.2.4 Modified ResNet50

He et al. (2016) proposed deep residual learning to construct a deeper network without facing the gradient vanishing problem, called ResNet. The ResNet50 included five convolutional blocks that the output of each convolutional block decreased half size when compared to the input. For the modified ResNet50 (mResNet50), we removed convolutional blocks 4 and 5 out from the network. Also, we added Conv1 with 512 feature maps at the end of convolutional block 3.

4.3.2.5 Modified DenseNet121

The DenseNet architecture (Huang et al. 2017) was proposed to collect knowledge from all previous layers and pass them to the next layer using the densely connected operation. Also, it required high computational time. The DenseNet121 contained two major layers: dense block and transition layer. The main network consisted of a convolutional layer, max-pooling layer, dense block, three times transition layer and dense block, and classification layer. The output of each layer decreased by half size, the same as the ResNet. For the modified DenseNet121 (mDenseNet121), however, we removed layers from the second transition layer. Hence, we attached Conv1x1 with 512 feature maps.

4.3.2.6 Modified MobileNetV2

The MobileNetV2 was proposed by Sandler et al. (2018) to reduce weighted parameters of a lightweight network using depthwise separable convolutional (DwConv) layers and inverted residuals of the bottleneck block. The main network comprised two block types: residual block with a stride of 1 and block with a stride of 2 to reduce the dimensionality of the feature map. The activation function used in the MobileNetV2 was the ReLU with the maximum value of 6 (ReLU6). The MobileNetV2 network consisted of DwConv, convolutional layers, seven bottleneck blocks with different repeated times, 1x1 convolutional layer, and global average pooling (GAP) layer. Since the output of layers is decreased by half size. For modified MobileNetV2 (mMobileNetV2), we removed the fourth bottleneck block and replaced them with Conv1 with 512 feature maps.

4.3.2.7 Modified EfficientNetB1

Tan and Le (2019) designed EfficientNets to search hyperparameters of the CNN architectures, including width scaling, depth scaling, resolution scaling, and compound scaling. In addition, the squeeze-and-excitation (SE) optimization was attached to the bottleneck block of EfficientNet to construct an informative channel feature by summation with GAP. Then find correlation features by reducing to small dimensions and transforming them to the original dimension. The EfficientNet was created based on the MobileNetV2, but it varies with resolutions, channels, and repeated times. For modified EfficientNetB1 (mEfficientNetB1), it is similar to mMobileNetV2, which removed the fourth bottleneck block and replaced Conv1 with 512 feature maps.

4.3.3 Recurrent neural network

Recurrent neural network (RNN) was a successful architecture that proposed to create a robust model from sequential data, such as speech (Alex el at. 2014), video (Donahue et al. 2017), and brain signals (Alhagry et al. 2017). RNN was designed by combining feedback loop connections that allow the output from the previous states to be applied as inputs of the current state. The feedback loop was performed in the hidden layers. However, RNN also had the limitation that constructs the output from only the previous context. The RNN is computed according to the following Equations.

$$y^t = f\left(Vh^t + b_y\right) \tag{13}$$

$$h^t = \sigma(Wx^t + Uh^{t-1} + b_h) \tag{14}$$

where y^t is the output of the RNN, h^t is the hidden state of the recurrent cell at time step (t) which is calculated by current input (x^t) and the previous hidden state (h^{t-1}). To calculate the h^t, RNN can be able to learn by adjusting the weighted parameters, including weighted matrices $(W, U$ and $V)$ and bias $(b_h$ and $b_y)$. Additionally, the output function $f(x)$ is an activation function and the sigmoid function $\sigma(x)$ is applied for hidden states.

4.3.3.1 Bidirectional recurrent neural network

Bidirectional recurrent neural network (BiRNN) was proposed to understand sequence data better than RNN architecture. It contained the backward $(\overleftarrow{h^t})$ and forward $(\overrightarrow{h^t})$ states connected to the same output layer that helped the network effectively increase the information context. The BiRNN architecture is shown in Figure 21 and is calculated as follows.

$$\vec{h}^t = \sigma\left(\vec{W}x^t + \vec{U}\vec{h}^{t-1} + \vec{b}_h\right) \tag{15}$$

$$\overleftarrow{h^t} = \sigma\left(\overleftarrow{W}x^t + \overleftarrow{U}\overleftarrow{h^{t-1}} + \overleftarrow{b}_h\right) \tag{16}$$

$$y^t = f\left(\overleftarrow{V}\overleftarrow{h^t} + \vec{V}\vec{h}^t + b_y\right) \tag{17}$$

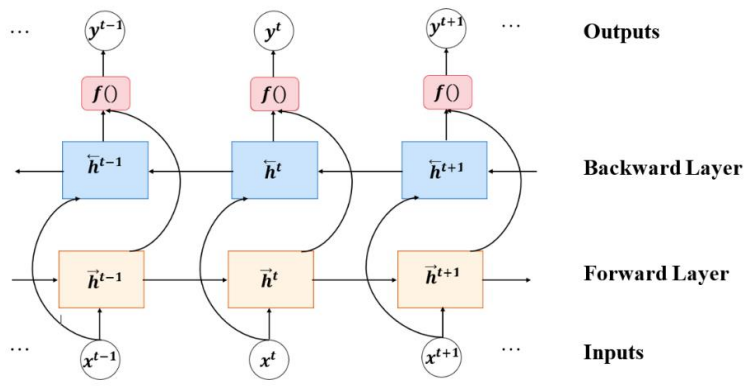


Figure 21 Illustration of bidirectional recurrent neural network.

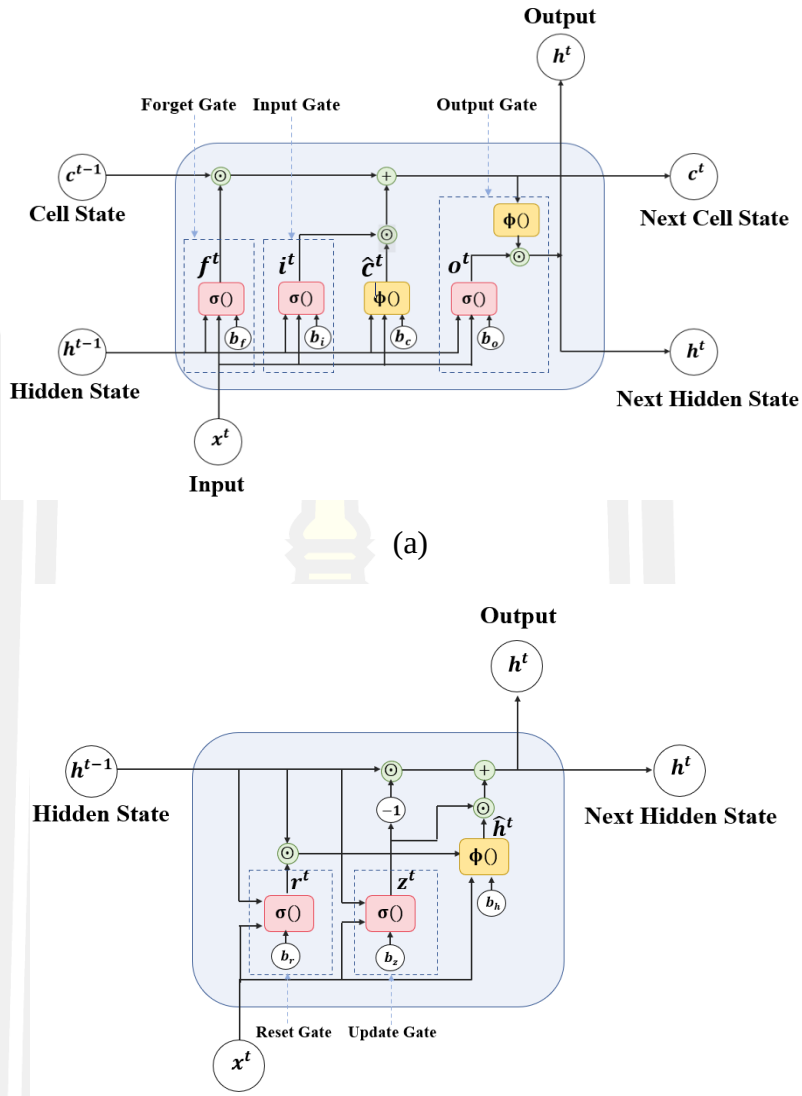


Figure 22 Illustration of the recurrent neural networks. (a) Long short-term memory and (b) gated recurrent unit.

4.3.3.2 Long Short-Term Memory

The Long Short-Term Memory (LSTM) was invented by Hochreiter and Schmidhuber (1997). It proposed to address the limitation of the RNN architecture that could not learn a long sequence and solve the vanishing and exploding gradient problems. The gates were designedto increase the memory capacity of the cell, including the forget gate, input gate, and output gate, as shown in Figure 22(a). The output o^t of the LSTM is computed using Equation 18 which has three input vectors, including input state x^t, previous hidden state h^{t-1}, and adding new cell state c^t. The output of the hidden state is calculated by the element-wise product ($\odot$) between the

current output of LSTM and the hyperbolic tangent function $\phi(\mathrm{x})$ of the cell state (c^t) by Equation 19. The LSTM is computed from the following Equation

$$o^t = \sigma(W_o x^t + U_o h^{t-1} + V_o c^t + b_o) \quad (18)$$

$$h^t = o^t \odot \phi(c^t) \quad (19)$$

where c^t is the cell state at time step (t) computed by Equation (8) which is designed to update new cell state by filtering of previous cell state (c^{t-1}) and cell candidate $\hat{c}^t$. The filtering operation is the sum of two element-wise products of forget gate and previous cell state, and input gate and cell candidate.

$$c^t = f^t \odot c^{t-1} + i^t \odot \hat{c}^t \quad (20)$$

where forget gate (f^t), input gate (i^t) and cell candidate ($\hat{c}^t$) are computed from the following Equation.

$$f^t = \sigma(W_f x^t + U_f h^{t-1} + V_f c^{t-1} + b_f) \quad (21)$$

$$i^t = \sigma(W_i x^t + U_i h^{t-1} + V_i c^{t-1} + b_i) \quad (22)$$

$$\hat{c}^t = \phi(W_c x^t + U_c h^{t-1} + V_c c^{t-1} + b_c) \quad (23)$$

In the LSTM, W_o, W_f, W_i, W_c, U_o, U_f, U_i, U_c, V_y, V_f, V_i, and V_c are weight matrices and b_o, b_f, b_i, and b_c are bias parameters which are required to adjust during training.

4.3.3.3 Gate Recurrent Unit

The Gate Recurrent Unit (GRU) was proposed by Cho et al. (2014) and designed to reduce the complexity of the LSTM architecture. The GRU architecture is shown in Figure 22(b). In the GRU, the input and forget gates were replaced with reset gate. The GRU introduces the reset gate r^t computed from the following Equation.

$$r^t = \sigma(W_r x^t + U_r h^{t-1} + b_r) \quad (24)$$

where the hidden state h^t is computed using (13).

$$h^t = (1 - z^t) \odot h^{t-1} + z^t \odot \hat{h}^t \quad (25)$$

where the update gate (z^t) and the candidate of hidden state ($\hat{h}^t$) are computed from the following Equation.

$$z^t = \sigma(W_z x^t + U_z h^{t-1} + b_z) \quad (26)$$

$$\hat{h}^t = \phi(W_h x^t + U_h(r^t \odot h^{t-1}) + b_h) \quad (27)$$

where W_r, W_z , W_h, U_r, U_z, and U_h are weight matrices and b_r, b_z, and b_h are bias parameters which are required to adjust during training.

BiRNN with high-capacity memory is shown to be more efficient to learn a sequence of context information than a single RNN layer. The BiRNN was also proposed to understand better sequence data that link the content from the backward state and link to the forward state.

In our proposed CRNN networks, the two bidirectional RNN layers were stacked on state-of-the-art CNN architectures. Two specials bidirectional RNNs, including BiLSTM and BiGRU, with diverse hidden unit sizes, were investigated.

4.3.4 Connectionist temporal classification

Connectionist Temporal Classification (CTC) is a conditional probability proposed to support RNN architecture that tackles various sequence problems (Alex Graves et al. 2006), such as speed recognition and handwritten text recognition. The output of the CTC algorithm is the sequence probabilities that are decoded from the RNN output at each time step. The condition probability $p(l|x)$ is calculated by the sum of probabilities of all possible paths, as described in Equation 28.

$$p(l|x) = \sum_{\pi \in G^{-1}(l)} p(\pi|x) \tag{28}$$

where $p(\pi|x)$ is probability of possible path (π), when x is sequence input that is predicted to sequence label (l), $x = x_1, x_2, \dots, x_T$, T is the length of the sequence, is member of L. In the handwritten text recognition problem, L is defined as 94 alphabets as shown in Table 11, and G is mapping function for transforming to l. For example, $G(\text{“}-\ -\ \text{H H}\ -\ \text{e l l}\ -\ \text{l}\ -\ \text{o o}\ -\ -\text{”}) =$ “Hello” or $G(\text{“}-\ -\ \text{ส ส}\ -\ \text{วั ส ส}\ -\ \text{ด ี}\ -\text{”}) =$ “สวัสดี” (in Thai). The $p(\pi|x)$ is computed as follows Equation.

$$p(\pi|x) = \prod_{t=1}^{T} x_{\pi}^{t} \tag{29}$$

where x_{π}^{t} is probability of having label π_t of time (t). CTC loss function is applied for training set (D) of pair input image and sequence label $(x_{i,}, l_i) \in D$. The CTC loss function is defined as follows Equation.

$$CTC\ loss = -\sum_{(x_{i,}, l_i) \in D} log\ \left(p(l_i|x_i)\right) \tag{30}$$

4.3.5 The proposed cyclical data augmentation strategy

In this section, we proposed a cyclical learning method, a novel data augmentation strategy called CycleAugment, that cooperates between training the CRNN model with applying data augmentation technique and without applying data augmentation technique. The transformation data augmentation technique and the CycleAugment strategy are described as follows.

4.3.5.1 Transformation data augmentation technique

We applied the basic transformation data augmentation techniques that include random shifting, rotation, and shearing, called $DA(w, h, r, s)$, where w and h are parameters of the maximum random percent of width and height shifting, respectively, r is an orientation rotation in the range of 0 to 360 degrees, and s is

orientation shearing in the range of 0 to 360 degrees. In our experiments, the default parameters of the $DA()$ were defined as w_{max} = 0.15, h_{max}=0.2 , r_{max}=5, and s_{max}=5.

In addition, we computed the scaling factor (α) to the DA(). The scaling factor affected the image directly by increasing and decreasing the image transformation.

4.3.5.2 CycleAugment strategy

Data augmentation is a fundamental process that can reduce overfitting but optimizing the data augmentation parameters is necessary to minimize the loss during training effectively. Therefore, we proposed the new cyclical data augmentation strategy, called the CycleAugment, effectively to minimize the validation loss and handle the overfitting problem. It was motivated by Huang et al. (Huang, Li, et al. 2017). In their method, the cyclic learning rate, which is the cycle of the adaptive learning rate, starts at the maximum number in each cycle and then decreases until the minimum number. Instead of adjusting the learning rate, our proposed method presented another approach to improve the performance of the data augmentation technique. It can achieve higher efficiency by taking the form of a cycle of adaptive data augmentation.

Algorithm 2. CycleAugment strategy for training CRNN

Input: model (m), number of max epochs (T), number of cycles (N)
$M = \lfloor \frac{T}{N} \rfloor$ is the number of epochs per cycle.
for epoch t = 1 **to** T **do**:
    if $mod(t, M) > \frac{M}{2}$: // determine the half of the cycle.
        Training model m without data augmentation
    else:
        $\alpha_t = a(t)$ // calculate a scaling factor (α) using Equation (19)
        Adjust the scaling factor (α_t) of w, h, r, and s
        Training model m with data augmentation $DA(w, h, r, s)$
end for
Output: trained model m

In the CycleAugment strategy, we first train the CRNN model by applying the data augmentation techniques in the first half of the cycle. Hence, the training and validation losses become high at the beginning and climb down to the local minima

value in each cycle. Second, we performed training without applying the data augmentation technique in the second haft of the cycle to minimize the loss in the local space with low data variants. As cyclic learning rate, a high learning rate at the start of the new cycle makes a high gradient for climbing out from the current local minima. As a result, the training and validation losses move up again and are ready to find new local minima. For our work, we focused on applying the data augmentation technique to increase the loss and the chance to escape the local minima. Finally, we repeat this step many times until the last cycle. We used the scaling factor as the linear decrement that starts from the maximum and decreases to the minimum values in each cycle. The equation and algorithm of the CycleAugment strategy are described in Equation 31 and Algorithm 2.

$$a(t) = 2 * (\alpha_{max} - \alpha_{min})(\frac{T-t}{T}) \quad (31)$$

where α is the scaling factor, t is the epoch, T is the maximum epoch.

4.4 Experimental results

4.4.1 Thai archive manuscript dataset

The handwritten text manuscripts used in our experiments are Thai archive manuscripts (Chamchong el at. 2019) collected from Thailand's national library. It was written approximately in 1902 AD using 94 Thai alphabets and contained 140 manuscripts. The 94 alphabets are shown in Table 11. A sample of the Thai archive manuscripts is shown in Fig 23. In this dataset, the handwritten text images were extracted from 140 manuscripts and contained 3,446-word images.

ที่ ๑๔/๔๔

Ground Truth: ที่ ๑๔/๔๔

พระที่นั่งวิมานเมฆ

Ground Truth: พระที่นั่งวิมานเมฆ

วันที่ ๕

Ground Truth: วันที่ ๕

เมษายน

Ground Truth: เมษายน

รัตนโกสินทรศก๓๕๑๒๑

Ground truth: รัตนโกสินทรศก๓๕๑๒๑

(a) (b)

Figure 23 Illustrated of Thai archive manuscript dataset. Examples (a) of the Thai archive manuscript and (b) word images and ground truths.

Types	Number of members	Members									
Consonants	44	ก	ข	ฃ	ค	ฅ	ฆ	ง	จ	ฉ	ช
		ซ	ฌ	ญ	ฎ	ฏ	ฐ	ฑ	ฒ	ณ	ด
		ต	ถ	ท	ธ	น	บ	ป	ผ	ฝ	พ
		ฟ	ภ	ม	ย	ร	ล	ว	ศ	ษ	ส
		ห	ฬ	อ	ฮ						
Vowel	19	◌ั	ะ	า	◌ำ	◌ิ	◌ี	◌ึ	◌ื	◌ุ	◌ู
		เ	แ	โ	ใ	ไ	ๅ	ฤ	ฦ	◌ๅ	
Tones	4	◌่	◌้	◌๊	◌๋						
Special symbols	17	ฯ	ๆ	็	์	"	(	)	+	,	
		-	.	/	x	_	–	(blank)			
Numeral	10	๐	๑	๒	๓	๔	๕	๖	๗	๘	๙
Total	**94**										

Table 11 The categories of Thai characters and other symbols.

4.4.2 Training strategy

4.4.2.1 Optimization algorithms

In the deep learning algorithms, various optimization algorithms were proposed to calculate the gradients of the error function while the network back-propagation. We evaluate three optimization algorithms, including stochastic gradient descent (SGD), Adam, and RMSprop, to converge the deep learning model and reduce the loss value while training with the same learning rate of 0.001, suitable for handwritten text recognition problems. For other parameters, here, we use the parameters momentum = 0.9 and decay rate = 0.001 as for SGD optimizer, discounting factor (γ) = 0.9 as for RMSprop optimizer, and the first estimate (β_1) = 0.9, the second estimate (β_2) and epsilon (ε) = 1e-07, for Adam optimizer.

In the experiments, we found that the Adam optimizer outperformed other optimizers. Further, all the experimental results shown in the following section evaluate based on the Adam optimizer.

4.4.2.2 Transfer Learning

Training the CNN model usually starts with random parameters, called scratch learning, and adjusts the weighted parameters by error gradient. Hence, the weighted parameters can extract high discriminative features from input images. Furthermore, transfer learning is derived from weighted parameters that learn from a large image dataset, called a pre-trained model. The pre-trained model contains prior knowledge of convolution filters. We can remove some top layers in the pre-trained model and attach a few new layers to the last layer of the pre-trained model.

For the transfer learning, we could train six CNN architectures: VGG16, VGG19, ResNet50, DenseNet121, MobileNetV2, and EfficientNetB1, using the pre-trained models, except only CCNet, because they did not provide the pre-trained model. In the RNN architectures, however, the random weighted parameters consist of Conv1x1 of 512 feature maps, global average pooling layer, BiRNN, and dense layer.

4.4.3 Quantitative evaluation

In this section, we evaluate CRNN architectures on the Thai archive manuscript dataset using character-level error rate (CER) as the evaluation metric. We also compare nine state-of-the-art CRNN models regarding the number of parameters and training time. Moreover, we evaluate the new data augmentation strategy (CycleAugment) and compare our CycleAugment strategy with the original data augmentation strategy. Both strategies apply data augmentation techniques based on transformation techniques, including random shifting, rotation, and shearing. In addition, we evaluate the CRNN models that train from scratch and use the transfer learning technique to understand wherewith the transfer learning technique affects the CRNN models.

The performance of the handwritten text recognition is evaluated based on the CER. CER is calculated as the minimal Levenshtein distance, which is the number of single-character modifications that change the predictive text from the ground truth transcription of the word (Théodore Bluche 2015). There are three operations of the CER metric, including insertion, deletion, and substitution. The CER is calculated by the following Equation:

$$CER = \frac{I+S+D}{N} \tag{32}$$

where I is the number of character insertions, S is the number of character substitutions, D is the number of character deletions, and N is the total number of characters in the target text.

4.4.4 Performance of different combination of CRNNs

To evaluate the performance of CRNN architectures, we resized all images to 64x496 pixels and used them as the input to the CRNN architectures. We trained all the CRNN models using the Keras framework with TensorFlow backend and trained on Google cloud with NVIDIA Tesla P100 GPU with 16GB of RAM.

For the training process, we divided the Thai archive manuscript dataset with the ratio of 70:10:20 for training, validation, and test, respectively. The nine CRNN networks (see Table 10) combined with two types of BiRNNs: BiLSTM and BiGRU. The number of RNN sizes with 128, 256, and 512 neurons were evaluated.

The CRNN networks were trained with the following parameters: 200 epochs, batch size of 32, Adam optimizer, the learning rate of 0.001, the first- and second-moment estimate values of 0.9 and 0.999, and epsilon of 1e-07.

Models	No. of Parameters			Training Time (hh:mm)			Character Error Rate (%)		
	RNN Sizes								
	128	256	128	128	256	128	128	256	128
CCNet-BiGRU (Chamchong, Gao, and McDonnell 2019)	0.49M	1.75M	6.62M	00:26	00:27	00:30	13.32	14.54	14.43
CCNet-BiLSTM (Chamchong, Gao, and McDonnell 2019)	0.64M	2.30M	8.78M	00:26	00:27	00:34	14.54	14.81	15.29
mCCNet-64-BiGRU	0.50M	1.75M	6.62M	00:26	00:27	00:33	15.19	16.23	16.48
mCCNet-64-BiLSTM	0.64M	2.31M	8.78M	00:26	00:27	00:30	16.09	16.64	14.36
mCCNet-512-BiGRU	0.87M	2.47M	8.03M	00:26	00:27	00:30	14.48	16.15	15.70
mCCNet-512-BiLSTM	1.13M	3.26M	10.65M	00:26	00:26	00:33	14.26	12.69	11.35
mVGG16-BiGRU	8.72M	10.32M	15.87M	00:50	00:54	01:00	11.41	11.37	14.05
mVGG16-BiLSTM	8.98M	11.10M	18.49M	00:50	00:54	01:00	9.04	12.03	14.01
mVGG19-BiGRU	11.67M	13.27M	18.82M	00:54	00:57	01:00	13.32	20.01	19.56
mVGG19-BiLSTM	11.97M	14.05M	21.44M	00:57	01:00	01:07	12.30	15.01	15.10
mResNet50-BiGRU	2.54M	4.14M	9.70M	00:37	00:40	00:43	8.40	8.22	10.77
mResNet50-BiLSTM	2.80M	4.92M	12.31M	00:37	00:40	00:47	11.16	7.29	8.21
mDenseNet121-BiGRU	2.39M	3.99M	9.55M	00:40	00:43	00:50	10.08	8.07	**7.44**
mDenseNet121-BiLSTM	2.65M	4.78M	12.17M	00:40	00:43	00:50	**7.72**	**7.13**	7.65
mMobileNetV2-BiGRU	0.98M	2.58M	8.14M	00:40	00:43	00:50	13.95	15.08	13.04
mMobileNetV2-BiLSTM	1.24M	3.36M	10.76M	00:40	00:43	00:50	10.91	9.13	9.73
mEfficientNetB1-BiGRU	1.06M	2.66M	8.22M	01:04	01:07	01:14	47.41	45.94	41.17
mEfficientNetB1-BiLSTM	1.32M	3.45M	10.84M	01:04	01:07	01:14	27.61	54.30	20.47

Table 12 Comparison of the parameters and computational time between different backbones CNNs and RNN sizes

Table 12 shows the comparison of the number of parameters and computation time between different CRNN backbones. The results showed that the CCNet-BiGRU proposed by Chamchong et al. (2019) provides 0.49M with the fewest parameters. It also spent less computation with only 26 minutes. Because CCNetBiGRU had only six weighted layers. In comparison, other CRNN architectures had more than 20 weighted layers, except only mVGG16 and mVGG16 had 14 and 16 weighted layers.

When comparing the number of parameters and time spent while training the CRNN model between BiGRU and BiLSTM, we found that the BiGRU always provides fewer parameters than the BiLSTM. Therefore, the time spent while training the CRNN model between BiGRU and BiLSTM showed the same approximate time.

In terms of the handwritten text recognition, Table 12 presents the CER value (%) in different RNN sizes (128, 256, and 512). The lowest CER value represents the best performance. In our experiments, the mDenseNet121-BiLSTM with the RNN size of 256 showed the fewest CER value of 7.13%. On the other hand, the mEfficientNetB1-BiGRU and -BiLSTM performed the worst with a CER value above 40% with BiGRU.

In the following experiments, we will continue experiments based on the best performance in each CNN architecture.

4.4.5 Performance of CRNN with CycleAugment strategy

In this experiment, we tested our CycleAugment strategy with all CRNN architectures to show that the proposed CycleAugment strategy obtains robust performance when training with every CRNN architecture. To discover the best CycleAugment strategy, we trained all CRNN models with 200 epochs in total and with 200 epochs, a network training five cycles (number of epochs per cycle M =200/5). A network training using transformation data augmentation technique in 20 epochs (M/2) and then switching to train the model without using data augmentation techniques in the following 20 epochs. Hence, continue the loop until the last epochs.

Table 13 presents the performance of the CycleAugment strategy. We achieved worthwhile performance when using CycleAugment with N=5. As a result, the CER value of the mEfficientNetB1-BiLSTM enormously decreased from 27.6% to only 7.74%. Consequently, the mResNet50-BiLSTM was the best CRNN model that achieved a 5.47% CER value using the CycleAugment strategy.

Models (RNN size)	Character Error Rate (%)					
	Number of Cycles (N)					
	N=1	N=2	N=3	N=4	N=5	N=6
CCNet-BiGRU (128) (Chamchong, Gao, and McDonnell 2019)	11.40	10.12	10.89	10.14	9.46	10.26
mCCNet-64-BiLSTM (512)	10.47	13.07	13.03	12.57	13.55	12.41
mCCNet-512-BiLSTM (512)	12.97	11.17	9.74	9.21	9.66	9.14
mVGG16-BiLSTM (128)	9.50	**5.70**	6.20	6.59	6.29	6.03
mVGG19-BiGRU (128)	9.87	9.26	9.52	7.84	7.51	7.18
mResNet50-BiLSTM (256)	**5.79**	5.97	**5.64**	**5.54**	**5.47**	**5.71**
mDenseNet121-BiLSTM (256)	6.02	5.97	6.46	6.29	5.64	6.29
mMobileNetV2-BiLSTM (256)	7.75	9.35	7.95	7.73	7.64	8.36
mEfficientNetB1-BiLSTM (128)	31.46	19.83	18.52	8.17	7.74	7.30

Table 13 Performance of different number of cycles in CycleAugment strategy Models (RNN size)

Model (RNN size)	Character Error Rate (%)					
	No Augmentation		Data Augmentation		CycleAugment	
	5-cv	Test	5-cv	Test	5-cv	Test
CCNet-BiGRU (128) (Chamchong el at. 2019)	13.79 ± 0.58	13.32	13.99 ± 0.97	12.29	10.33 ± 0.67	9.46
mCCNet-64-BiLSTM (512)	18.26 ± 1.36	14.36	17.70 ± 1.71	13.78	13.34 ± 1.07	13.55
mCCNet-512-BiLSTM (512)	11.93 ± 0.76	11.35	11.55 ± 0.72	11.18	9.73 ± 0.64	9.66
mVGG16-BiLSTM (128)	8.94 ± 0.78	9.04	10.37 ± 0.74	11.71	6.77 ± 0.40	6.29
mVGG19-BiGRU (128)	13.34 ± 2.15	10.25	12.74 ± 0.56	10.10	7.62 ± 0.27	7.51
mResNet50-BiLSTM (256)	9.54 ± 1.39	7.29	7.89 ± 0.43	7.85	6.65 ± 0.40	**5.47**
mDenseNet121-BiLSTM (256)	**7.15 ± 0.57**	**7.13**	**7.57 ± 0.60**	**7.44**	**6.55 ± 0.75**	5.64
mMobileNetV2-BiLSTM (256)	10.83 ± 0.87	9.13	11.64 ± 0.88	10.47	7.93 ± 0.52	7.64
mEfficientNetB1-BiLSTM (128)	29.75 ± 2.17	27.61	29.01 ± 2.28	26.19	12.19 ± 0.78	7.74

Table 14 Performance of scratch learning different data augmentation strategies

Model (RNN size)	Character Error Rate (%)					
	No Augmentation		Data Augmentation		CycleAugment	
	5-cv	Test	5-cv	Test	5-cv	Test
mVGG16-BiLSTM (128)	9.00 ± 0.83	7.63	10.24 ± 0.60	7.31	6.88 ± 0.32	**5.43**
mVGG19-BiGRU (128)	13.61 ± 2.39	15.05	14.60 ± 1.45	14.89	7.64 ± 0.52	7.37
mResNet50-BiLSTM (256)	8.19 ± 0.73	**7.15**	7.67 ± 0.51	**7.54**	6.17 ± 0.53	5.69
mDenseNet121-BiLSTM (256)	**7.04 ± 0.40**	7.18	**7.62 ± 0.40**	7.48	**5.82 ± 0.40**	5.69
mMobileNetV2-BiLSTM (256)	10.72 ± 0.85	9.62	11.12 ± 1.17	10.18	7.88 ± 0.58	7.44
mEfficientNetB1-BiLSTM (128)	24.34 ± 3.25	27.68	24.38 ± 2.67	28.29	10.77 ± 1.12	7.60

Table 15 Performance of transfer learning different data augmentation strategies

Furthermore, to demonstrate that our CycleAugment strategy outperforms the original data augmentation strategy, we train CRNN architecture with two different data augmentation strategies using a 5-fold cross-validation technique (5-cv). We also trained the CRNN model using the learning from scratch and transfer learning techniques. All the experimental results are shown in the following section.

We compare scratch learning and transfer learning, as shown in Table 14 and Table 15, with different training strategies, including training without applying data

augmentation, training with applying transformation data augmentation techniques with scaling factor α = 1, and training with applying CycleAugment strategy with the number of cycles N= 5, α_min = 0.5, and α_max= 2.5.

As a result, the CycleAugment strategy outperformed other data augmentation strategies on both scratch learning and transfer learning. For scratch learning, the CycleAugment achieved the best CER value of 5.47% on the test set when training with mResNet50-BiLSTM (256). For transfer learning, we found that the mVGG16-BiLSTM (128) outperformed all the CRNN architectures with the CER value of 5.43% on the test set.

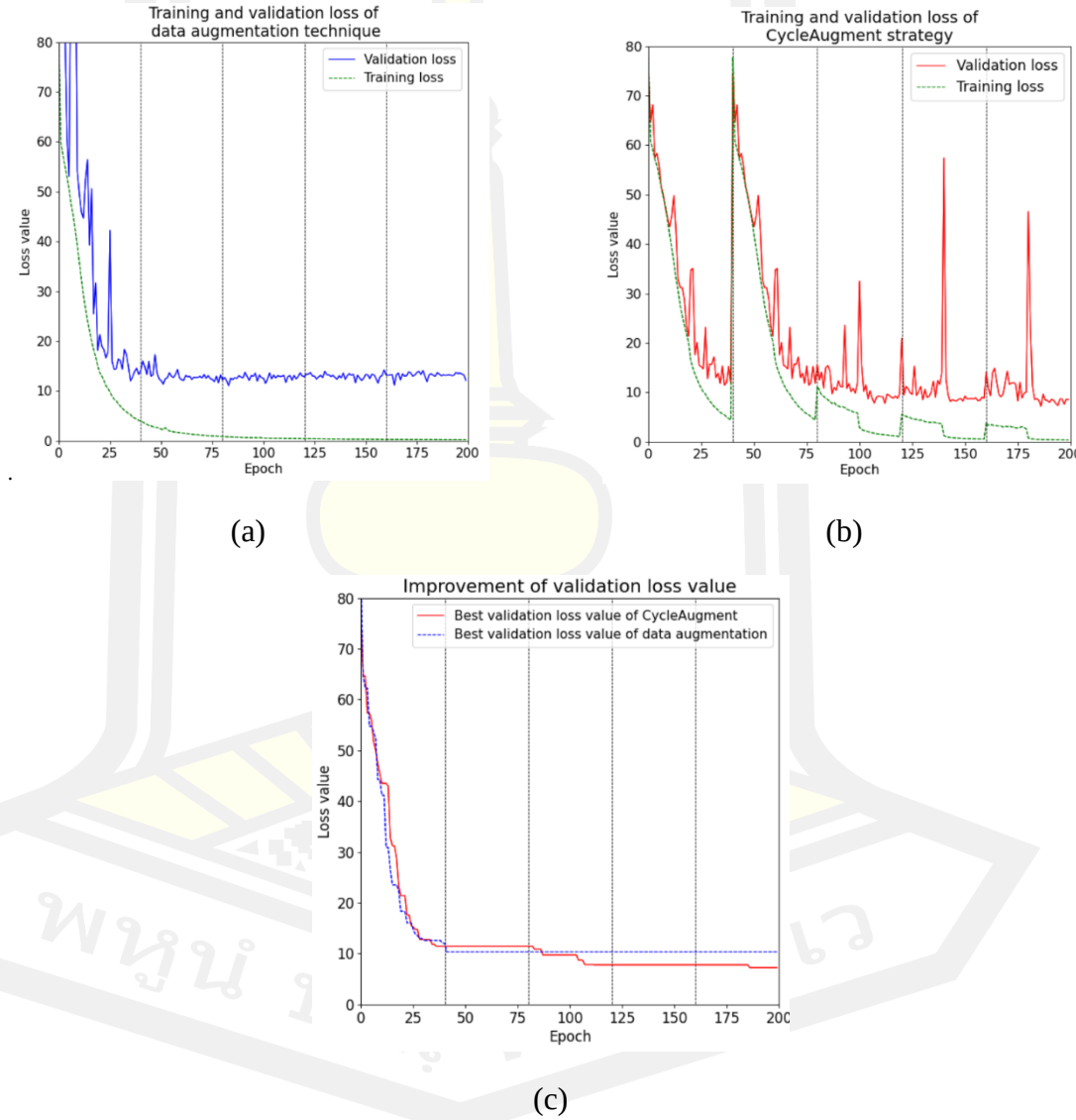
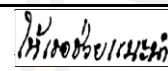
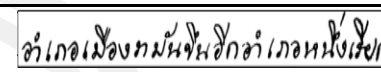


Figure 24 Illustration of the training loss and validation loss values of (a) original data augmentation technique, (b) CycleAugment strategy, and (c) best loss value

Consequently, we evaluated the CRNN architectures using the 5-fold cross-validation (5-cv). We found that both scratch and transfer learning provided nearly

similar CER values because we could transfer a few parameters of the CNN pre-trained models. For example, only 1 million parameters could transfer from the pre-trained ResNet50 model, while the total parameters of the mResNet50-BiLSTM (256) are 4 million.

Input Image			
Ground truth		ให้เธอช่วยแนะนำ	อำเภอเมืองรามันขึนอีกอำเภอหนึ่งเรียก
No data augmentation	mVGG16-BiLSTM (128)	ให้เธอช่วยแนะนำ	อำเภอเมืองรามันจินอีกอำเภอหน่งเรียก
	mVGG19-BiGRU (128)	ให้เธอช่วยแนะหนำก	อำเกอเมืองรามันจินอีกอำเภาอหนี่ยงรียก
	mResNet50-BiLSTM (256)	ให้เธอช่วยแนะนำ	อำเภอเมืองรามันจ้นอีกอำเภอหนึ่งเรียก
	mDenseNet121-BiLSTM (256)	ให้เธอช่วยแนะนำ	อำเภอเมืองรามันจิ่นอีกอำเภอหนึ่งเรียก
Data augmentation	mVGG16-BiLSTM (128)	ให้เธอช่วยแนะนำ	อำเถอเมืองรามันขินอีกอำเภอหนึ่งเรียก
	mVGG19-BiGRU (128)	ให้เธอช้วยแนนนำ	อำเภอเมืองรามันินจีดซำเราจหนึ่งมียกด
	mResNet50-BiLSTM (256)	ให้เธอช่วยแนะนำ	อำเภอเมืองถามันขึนอีกอำเรภอหนึ่งเรียก
	mDenseNet121-BiLSTM (256)	ให้เธอช่วยแนะนำ	อำเถอเมืองรามันจีนอีกอำเภอหนึ่งเรียก
CycleAugment	mVGG16-BiLSTM (128)	ให้เธอช่วยแนะนำ	อำเภอเมืองรามันขึนอีกอำเภอหนึ่งเรียก
	mVGG19-BiGRU (128)	โ้เอช่อยแนะนำ	อำเภวเมืองทมันจีนอีกตำเภอหนึ่งเือยก
	mResNet50-BiLSTM (256)	ให้เธอช่วยแนะนำ	อำเภอเมืองรามันขึนอีกอำเภอหนึ่งเรียก
	mDenseNet121-BiLSTM (256)	ให้เธอช่วยแนะนำ	อำเภอเมืองทามันขี่นอีกอำเภอหนึ่งเรียก

*Note that green text represents the correct text recognition without error (CER value = 0%), blue characters with an underline represent error characters, and black characters are the correct character recognition.

Table 16 Results of handwritten text recognition using different CRNN models Input Image

In Figure 24, we illustrate the loss values of two data augmentation strategies. The training and validation loss of the original data augmentation strategy was presented in Figure 24(a). The training loss values reduced smoothly and closed to zero, but the validation loss was not performing below 10. If we train the model more than 200 epochs, the validation loss may be increased. On the other hand, loss values of the CycleAugment strategy, as shown in Figure 24(b), were rapidly decreased in

the first cycle and then grew up at the beginning of the next cycle. Because, firstly, the CRNN model learns from without applying the data augmentation technique, so the CRNN model tries to converge that particular pattern. Secondly, the CRNN attempts to fit the model with the new input data when applying the data augmentation techniques. Since the model never trained with the new data, that is why the loss value grew up and quickly decreased again. We then showed the performance of the CycleAugment strategy compared to the original data augmentation strategy, as shown in Figure 24(c). Furthermore, when we train more epochs, the loss value still could slowly decrease, while the loss value of the original data augmentation strategy stopped decreasing from around epoch 40. It demonstrates that the CycleAugment strategy could benefit from learning with different patterns and avoiding overfitting problems.

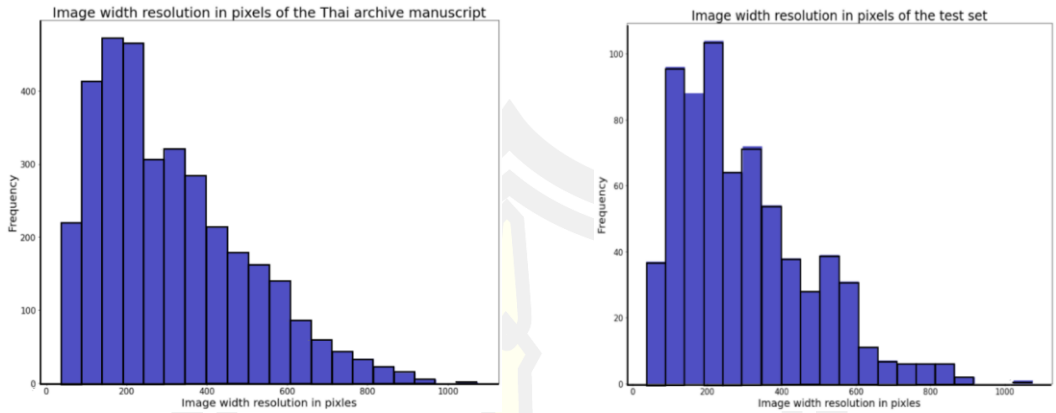
Examples of handwritten text recognition using CRNN models. When the green text means the CRNN model recognizes and obtains correct output with the whole words. The blue characters with underlining are misclassified characters, as shown in Table 16.

4.4.6 Performance on short word recognition

In previous experiments, we focused on the performance of the handwritten text dataset consisting of various distributions of word length. Figure 25 presents the histogram of image width resolution in pixels on (a) whole dataset and (b) only test set. The image width is distributed in the range of 36 to 1,075 pixels. Due to various ranges of the image width, we are curious whether the short word images (image width between 36 to 186 pixels) affect the recognition performance.

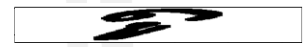
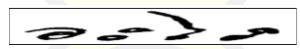
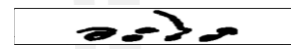
We evaluated the performance of the short word by selecting the short word images from the test set. First, we resized the short word into 64x496 pixels and then recognized the short word images. Second, we resized the short word images into 64x346 pixels and then added white space on the left (75 pixels) and right (75 pixels) sides to prevent the distortion of the texts in the image. Hence, the input image is equal to 64x496 pixels.

We examined the short word performance using mResNet50-BiLSTM (256) model. Firstly, we evaluated the performance of the short word images (see Table 17 in the second column) and achieved the CER value of 8.07%. Secondly, the short word images were adjusted by adding the white space (see Table 17 in the third column). The experimental results showed that adding the white space before sending to predict by the CRNN model presented better performance than resizing the image. It achieved a CER value of 7.04%. Consequently, we found that image distortion could harm the handwritten text recognition system. It is necessary to rescale the short word images and combine a space into the images before recognizing them.



(a) (b)

Figure 25 Illustrated histograms of the image width resolution in pixels. (a) The Thai archive manuscript and (b) test set of the Thai archive manuscript.

Original Image Resolution	**Testing Image Resolution**	**Adjusting Image Resolution**
		
Ground truth = ดู	Prediction = ดา	Prediction = ด
		
Ground truth = ๓	Prediction = กา	Prediction = ๓
		
Ground truth = อะไร	Prediction = อฮโร	Prediction = อ๐ไร
		
Ground truth = เมื่อ	Prediction = เมือ	Prediction = เมื่อ
All short word test images	CER = 8.07%	CER = 7.04%

*Note that green text represents the correct text recognition without error (CER value = 0%), blue characters with an underline represent error characters, and black characters are the correct character recognition.

Table 17 Examples of short word recognition when resizing images into 64x496 pixels (second column) and adding white space to prevent image distortion (third column)

4.5 Discussion

4.5.1 CycleAugment strategy

As it is known, deep learning requires data augmentation techniques to improve performance and avoid overfitting problems. To create the robust CRNN model, we then applied the data augmentation technique. The experimental results showed that the data augmentation techniques did not always confirm the best performance. Consequently, the CRNN model will find only the global minima value when training the CRNN with the original data augmentation strategy. The training loss never more increases, as shown in Figure 24(a). Indeed, it increases the chance of encountering overfitting problems.

We then proposed the new cyclical learning method, namely the CycleAugment strategy. The proposed strategy can effectively improve the performance of handwritten text recognition by escaping the trapping in global minima and overfitting problems. The CycleAugment strategy increases the chances of discovering local minima in each cycle by switching between two training states with and without applying data augmentation while training the CRNN model, as shown in Figure 24(b). The CRNN model adapted to the local minima because the weight of the CRNN architecture is adjusted using a high error gradient value obtained from variation of the input images.

Moreover, the geometric transformation data augmentation process requires an appropriate scaling factor to adjust the effect of rotation, shift, scaling, and shear. If the scaling factor is large, it will affect the convergence time of training (training losses decrease slowly or never), but using a minimal value, the result will still be overfitting. Our CycleAugment proposes this adaptive scaling factor by cyclic scheduling to reset and reduce the scaling factor. It can disturb the weight of deep learning, as shown in Figure 24(b). The increase in the loss that CycleAugment offers to avoid overfitting better than traditional data augmentation methods. Additionally, each reset cycle causes the network to find a new local minima and offers an opportunity to optimize the model in the subsequent cycles.

In terms of computational times, the training time in each epoch of CycleAugment is not increased compared to traditional data augmentation. Our experiments in this research set an equal training number to 200 epochs for all methods. Therefore, there is no incremental time difference between the CycleAugment method and the traditional data augmentation method. However, considering Figure 24(a), the number of traditional data augmentation epochs and the validation loss started to converge around 50 epochs. If it applies an early stop for training, the number of epochs might stop training at about 75 epochs. Thus, it can reduce training time by 2.67 times compared to CycleAugment. However, the accuracy performance is still less than the proposed methods by about 2%.

4.5.2 Effective of transfer learning technique

We have learned from much research that the transfer learning technique consistently performed better than scratch learning (Pawara et al. 2017; Gonwirat and Surinta 2020; Enkvetchakul and Surinta 2021). Therefore, we evaluated the performance of the scratch and transfer learning, as shown in Table 14 and 15. The experimental results were quite surprised that the transfer learning performance did not significantly outperform the scratch learning. However, in the CRNN architecture, we discovered that the transfer learning did not show outstanding results because the number of transfer parameters from the pre-trained CNN model is more limited than the parameters in the RNN architecture. We have to train the RNN model with huge parameters that did not transfer from the pre-trained model. The parameters of the RNN architecture are larger, approximately four times more than the CNN architecture.

4.5.3 Improvement of short word recognition

We also observed that short word images directly decrease the performance of the handwritten text recognition system, as shown in Table 17. We found that the short word images are always distorted when resizing to the fixed input of the CRNN architecture. Hence, we employed the most straightforward technique that avoids distortion of text information in short word images. The simple technique is to adjust the short word images by adding white space on both sides of the image. The performance was presented when applying our proposed method.

4.6. Conclusion

In recent years, a few research aimed to address the challenge of the Thai handwritten text recognition system. This study discovered a robust CRNN architecture, a sequence learning approach that achieves high accuracy on the Thai handwritten text recognition system. For training the CRNN model, the original data augmentation strategy is proposed. The CRNN model was trained by applying the transformation data augmentation techniques from the first training epoch until the last epoch. With this training strategy, the CRNN model slowly obtained the global minima value. The model can face overfitting problems because the training loss decreases to the lowest value. However, the validation loss sometimes does not converge to the lowest value. However, we invented a cyclical data augmentation strategy called CycleAugment, to avoid finding the global minima and control overfitting problems. In our strategy, the whole training epochs are divided into cycles. In each cycle, we assign the CRNN model to discover the local minimal value. Hence, it repeatedly starts at high loss value by learning new patterns from the training images when beginning a new cycle. As a result, the weight model is adapted by a high gradient value. The benefit of our proposed CycleAugment strategy is that the CRNN model can learn from both with and without applying data augmentation techniques.

In the experiments, we evaluated nine CRNN architectures to recognize handwritten text on the Thai archive manuscript dataset. The result showed that the mDenseNet121-BiLSTM(256) outperformed all the CRNN architectures. First, we performed the CRNN architectures using scratch and transfer learning. It is quite surprising that transfer learning does not show a significant performance when compared with scratch learning. Second, we trained the CRNN models with three different data augmentation strategies: without data augmentation, with data augmentation, and CycleAugment. The proposed CycleAugment strategy achieved the best performance when combined with all CRNN models. Finally, we are concerned about the performance of the CRNN model when predicting the short word images. The text information inside the short word images is regularly distorted when transformed into the input of the CRNN model with the same size as the long word images. We proposed the simple technique is of adding white space on both sides of the short word images. We achieved a better result with the simple technique.

Chapter 5

Discussion

The objective of this thesis is to propose deep learning approaches to address the problems of handwritten text recognition in historical documents. Firstly, we proposed to investigate different CNN architectures to improve the accuracy rate of character handwritten. Secondly, we enhance the quality of degraded images to increase the performance of handwritten character recognition. Thirdly, character segmentation is a difficult problem since Thai historical document has connected and cursive writing text. To overcome this problem, we approach CRNN to learn the sequence characters of words without segmenting them into a single character.

We will briefly describe and discuss the challenges of Thai handwritten text recognition in modern and historical documents using a deep learning approach.

Chapter 2 showed that, due to the challenges of Thai handwritten character recognition, each person's writing style emphasizes weight while writing the alphabet (e.g., curve, head, loop, and curl), and some characters are like other characters. We proposed approaching the CNN models, including VGG19 and Inception-ResNet, to solve these challenges. With comparison methods, the feature extractions consist of siftD and HOGfoDRs methods are recognized by SVM. In addition, we transferred pertained weight parameters of CNNs to overcome better training time of network convergence in the short epochs and have archived better performance comparison with scratch learning.

In the experiment, the dataset was Thai handwritten characters, namely THI-C68, which has 14,490 characters in 68 classes, and we divided the dataset into a training set and test set with 80% and 20% ratios, with 13,041 training images and 1,449 test images. The experimental results showed that the VGG19 architecture with transfer learning improved accuracy by approximately 1-2% compared to the training from the scratch method. The VGGNet with a scratch learning process compared to 10-fold cross-validation achieved an accuracy rate of 97.93%, which is 3% higher than siftD-SVM and VGGNet-Transfer (98.81%) and achieved an insignificantly higher effective rate than HOGFoDRs-SVM (98.76%).

In Chapter 3, we mainly concentrated on robust the performance of handwritten character recognition in degradation documents or noisy characters. Firstly, we proposed to use GAN, namely DeblurGAN, to reconstruct the noisy characters. Secondly, we proposed the DeblurGAN-CNN for denoising and recognition in a single network. To construct the network, we propose the setting and training scheme including pretraining DeblurGAN and CNN, connecting them, and fine-tuning the network. To find the best combination of the network, we investigate four state-of-the-art CNN architectures, consisting of VGG19, Inception-ResNet, MobileNetV2, and DenseNet121. We use data augmentation by generating noisy

characters and transfer learning to improve the quality of training. Furthermore, we presented a new noisy Thai handwritten character dataset, called noisy THI-C68. The dataset was generated by synthesizing noisy character images using five different noisy techniques: low resolution, AWGN, low contrast, motion blur, and mixed noise.

To evaluate the novel architecture, firstly, we experimented to find which CNN architectures are suitable for handwritten characters without or with noisy effects. Secondly, we evaluate DeblurGAN to clean noisy character images. Finally, we evaluated our proposed, DeblurGAN-CNN, on the performance of handwritten character recognition on two noisy handwritten datasets: n-THI-C68 and n-MNIST, and two handwritten character datasets: THI-C68 and THCC-67. We found that MobileNetV2 and DenseNet121 are competitive in achieving high accuracy.

In Chapter 4, we have attempted to address the challenge of the Thai handwritten text recognition system in terms of word or line recognition. This study discovered a robust CRNN architecture, a sequence learning approach that achieves high accuracy on the Thai handwritten text recognition system. For training the CRNN model, the original data augmentation strategy is proposed. The CRNN model was trained by applying the transformation data augmentation technique from the first training epoch until the last. The CRNN model slowly obtained the global minima value with this training strategy. The model can face overfitting problems because the training loss decreases to the lowest value. However, the validation loss sometimes does not converge to the lowest value.

Nonetheless, we invented a cyclical data augmentation strategy called CycleAugment, to avoid finding the global minima and control overfitting problems. In our strategy, all training epochs are divided into cycles. We assign the CRNN model in each cycle to discover the local minima. Hence, it repeatedly starts at a high loss value by learning new patterns from the training images when beginning a new cycle. As a result, the weight model is adapted by a high gradient value. The benefit of our proposed CycleAugment strategy is that the CRNN model can learn from both with and without applying data augmentation techniques.

In experiments, we evaluate the CRNN with Thai handwritten text recognition on the dataset about the Thai archive manuscript consisting of 3,446 word images. Firstly, we investigate the best combination of backbone CNNs and BiRNNs. Finally, we are concerned about the performance of the CRNN model when predicting the short word images. The text information inside the short word images is regularly distorted when transformed into the input of the CRNN model with the same size as the long word images. We proposed the simple technique of adding white space on both sides of the short word images. We achieved a better result with the simple technique.

In this dissertation, three robust approaches to improve the performance of handwritten text and character recognition are proposed, including the VGG with

transfer learning, the DeburGAN-CNN network for problems handwritten character recognition, and CRNN architectures to archive word handwritten recognition.

5.1 Answers to The Research Questions

According to the research questions (RQ) in Chapter 1, we explain the improvement of handwritten text recognition in historical documents with three solutions. In this section, we briefly answer each research question.

RQ1: Character recognition is a fundamental problem in document analysis and recognition. In historical document images, handwritten characters are usually found and challenging to solve due to various personal writing and cursive style. Previous works aim to extract features from local descriptors as a hand-crafted feature and recognize them by machine learning techniques such as support vector machines (SVM), K-nearest neighbor (KNN), or multilayer perceptron (MLP). In contrast, we propose to investigate CNN architectures that can automatically extract features and recognize them. Is it possible to improve the recognition performance of handwritten characters? And which CNN architectures are suitable for this problem?

We focus on state-of-the-art CNN models consisting of VGGNet (Simonyan and Zisserman 2015) and Inception- ResNet-v2 architectures to compare the performance between deep CNN architectures and proposed two learning methods, including scratch and transfer learning. In addition, we compare CNNs with siftD-SVM (Surinta, Karaaba, et al. 2015) and HOGFoDRs-SVM (Inkeaw et al. 2019) to evaluate effectiveness.

To answer RQ1, we experimented with handwritten character recognition of Thai characters with the ALICE-THI dataset by choosing a specific test on the THI-C68 dataset and investigating our proposed two CNN architectures with scratch and transfer learning. It found that the proposed two CNNs are better results than previous works of the local descriptors and are recognized by machine learning techniques. Transfer learning significantly increased the accuracy rate for CNN architecture by 1-2% compared to the training from the scratch method. Transfer learning is a way to reduce learning time and increase recognition efficiency. The research has shown that VGGNet-19 architecture with transfer learning is suitable and has been designed to be stacked together to make it easier to learn from the network and increase the recognition speed.

Consequently, we can improve the performance of Thai handwritten characters with the highest accuracy rate.

RQ2: Document degradations are caused by document aging effects and image acquisition with light conditions or a moving camera. These problems, called noisy character images, can decrease the recognition performance of handwritten characters. How can we improve the recognition rate of noisy characters? Can we assume that denoise GAN to clean noisy image provides better accuracy result of

CNN? Furthermore, can a single DeblurGAN-CNN network enhance performance when recognizing different types of noisy characters?

We propose to the robust generative adversarial network (GAN) combined with the convolutional neural network (CNN) architecture, called DeblurGAN-CNN, was proposed to synthesize new clean handwritten characters from noisy handwritten characters and recognition with increased performance. The DeblurGAN-CNN architectures were trained by applying the noisy data augmentation techniques and transfer learning to create a robust model.

To answer RQ2, we first evaluated the DeblurGAN architecture with two well-known image quality metrics, the peak signal-to-noise ratio (PSNR) and the structural similarity index (SSIM) on the n-THI-C68 dataset. We found that the PSNR and SSIM values were obtained when evaluating the different noise methods. The high PSNR and SSIM values represent better accuracy and reconstruction image, respectively.

To improve recognition performance, we presented the DeblurGAN-CNN models on two noisy handwritten datasets: n-THI-C68 and n-MNIST, and two handwritten character datasets: THI-C68 and THCC-67. The results showed that the DeblurGAN-CNN architectures generated strong handwritten character images and achieved the highest performance on the n-MNIST and n-THI-C68 datasets compared with other existing methods.

RQ3: Thai historical documents are cursive writing style and difficult to segment to each character. Indeed, we focus on a word or line recognition by sequence learning method suitable for handwritten documents. The CRNN is a deep learning technique applied to various text recognition problems such as sense text recognition, video subtitle, and handwriting document. What is the best combination of CNN and RNN to construct robust CRNN in word or line recognition? Furthermore, the limitation of the dataset is insufficient handwritten text images for training. We propose a novel data augmentation technique for training CRNN; Is it possible to enhance the performance of Thai handwritten word recognition?

To answer the last question in RQ3, we evaluated nine CRNN architectures to recognize handwritten text on the Thai archive manuscript dataset. The result showed that the mDenseNet121-BiLSTM(256) outperformed all the CRNN architectures. First, we performed the CRNN architectures using scratch and transfer learning. It is quite surprising that transfer learning did not show a significant performance compared to scratch learning. Second, we trained the CRNN models with three different data augmentation strategies: no data augmentation, data augmentation, and CycleAugment. The proposed CycleAugment strategy achieved the best performance when approached with all CRNN models.

5.2 Future Work

This dissertation presents recent deep learning techniques in handwritten text recognition, including CNN for character recognition, DeblurGAN to reconstruct the handwritten character of various noisy types, and CRNN in historical word recognition.

Due to insufficient handwritten text images for training the CRNN model, the model might not give generalizations. We might need to synthesize the handwritten text images and use them as the training set. The generative adversarial network (GAN)) (Fogel et al. 2020) is the best choice to study and synthesize the training set. In sequential learning, we must investigate an attention-based model (Shi et al. 2018; Luo et al. 2019; Atienza 2021) and word beam search (Ameryan and Schomaker 2021) to better predict handwritten text images.

Since the CycleAugment sets a fixed number of cycles, the number of epochs per cycle follows to fix. The inappropriate number of cycles can affect the network's losses. For example, if the number of cycles is quite frequent (the number of epochs per cycle is tiny), the network might be unable to train for convergence or be slower. In the first cycle in Figure 24(b), the validation loss is still not entirely decreasing to the bottom, which might affect the convergence of the network, called the immaturity state. Therefore, in further research, instead of using a fixed number of epochs per cycle, which might cause an early reset. We need to investigate that allows the network to reach a maturity state. The adaptive methods adjust an appropriate number of epochs which will expect the network to be better perform.

Moreover, each reset of the cycle results grows up to a high validation loss. It indicates that it has a high effect and possibly slows convergence. Instead of cyclical, simulated annealing can be proposed to disturb a slight change of network losses and control the effect not to be the high difference during all training epochs. It is another way to escape from local minima, which might reduce the convergence time and make it more efficient.

To enhance deep learning performance, we plan to work on the ensemble CNNs technique and combine the DeblurGAN-CNN architecture as a part of the ensemble CNNs technique (Guo et al. 2019; Gonwirat and Surinta 2021) to achieve much higher accuracy. Another direction for future work is creating new DeblurGAN-CNN architecture by searching for efficient architecture with a lightweight model. Finally, we will embed DeblurGAN-CNN with the recurrent neural networks (RNNs) (Ameryan and Schomaker 2021) or vision transformers (Dosovitskiy et al. 2021; Souibgui et al. 2022) to recognize word and sentence images.

Finally, researchers will design deep architectures that reduce the number of parameters and learning time. However, the quality must still be equivalent to or better performance than with the previous architecture and it will be tested with handwritten characters in other languages similar to Thai, such as Thai Noi in Northeastern, Lanna in North of Thailand, or Khmer.

REFERENCES

Abdurahman, Fetulhak, Eyob Sisay, and Kinde Anlay Fante. 2021. “AHWR-Net: Offline Handwritten Amharic Word Recognition Using Convolutional Recurrent Neural Network.” *SN Applied Sciences* 3(8): 1–11. https://doi.org/10.1007/s42452-021-04742-x.

Alhagry, Salma, Aly Aly Fahmy, and Reda A El-Khoribi. 2017. “Emotion Recognition Based on EEG Using LSTM Recurrent Neural Network.” *International Journal of Advanced Computer Science and Applications* 8(10): 355–58. http://dx.doi.org/10.14569/IJACSA.2017.081046.

Alom, Md Zahangir et al. 2018. “Handwritten Bangla Character Recognition Using the State-of-the-Art Deep Convolutional Neural Networks.” *Computational Intelligence and Neuroscience* 2018: 1–13.

Ameryan, Mahya, and Lambert Schomaker. 2021. “A Limited-Size Ensemble of Homogeneous CNN/LSTMs for High-Performance Word Classification.” *Neural Computing and Applications* 33(14): 8615–34. https://doi.org/10.1007/s00521-020-05612-0.

Atienza, Rowel. 2021. “Vision Transformer for Fast and Efficient Scene Text Recognition.” In *International Conference on Document Analysis and Recognition (ICDAR), ,* 319–34.

Basu, Saikat et al. 2015. “Learning Sparse Feature Representations Using Probabilistic Quadtrees and Deep Belief Nets.” *Neural Processing Letters* 45(3): 855–67.

Bhunia, Ayan Kumar, Ankan Kumar Bhunia, Aneeshan Sain, and Partha Pratim Roy. 2019. “Improving Document Binarization via Adversarial Noise-Texture Augmentation.” In *International Conference on Image Processing (ICIP), ,* 2721–25.

Bluche, T, H Ney, and C Kermorvant. 2013. “Feature Extraction with Convolutional Neural Networks for Handwritten Word Recognition.” In *International Conference on Document Analysis and Recognition (ICDAR), ,* 285–89.

Bluche, Théodore. 2015. “Deep Neural Networks for Large Vocabulary Handwritten Text Recognition.” Universite´ Paris Sud-Paris XI, France. https://tel.archives-ouvertes.fr/tel-01249405.

Boracchi, Giacomo, and Alessandro Foi. 2011. “Uniform Motion Blur in Poissonian Noise : Blur / Noise Tradeoff.” *IEEE Transactions on Image Processing* 20(2): 592–98.

Butt, Hanan et al. 2021. “Attention-Based CNN-RNN Arabic Text Recognition from Natural Scene Images.” *Forecasting* 3(3): 520–40.

Chamchong, Rapeeporn, Wei Gao, and Mark D. McDonnell. 2019. “Thai Handwritten Recognition on Text Block-Based from Thai Archive Manuscripts.” In *International Conference on Document Analysis and Recognition (ICDAR), ,* 1346–51.

Chamchong, Rapeeporn, Umaporn Saisangchan, and Pornntiwa Pawara. 2021. “Thai Handwritten Recognition on BEST2019 Datasets Using Deep Learning.” In *International Conference on Multi-Disciplinary Trends in Artificial Intelligence (MIWAI)*, , 152–63.

Chen, Xiaoxue et al. 2021. “Text Recognition in the Wild: A Survey.” *ACM Computing Surveys* 54(2): 1–35.

Chen, Ye et al. 2021. “Transformer Text Recognition with Deep Learning Algorithm.” *Computer Communications* 178: 153–60. https://www.sciencedirect.com/science/article/pii/S0140366421001754.

Cho, Kyunghyun et al. 2014. “Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation.” In *Conference on Empirical Methods in Natural Language Processing*, , 1724–34.

Choudhary, Amit, Rahul Rishi, and Savita Ahlawat. 2013. “A New Character Segmentation Approach for Off-Line Cursive Handwritten Words.” In *Procedia Computer Science*, , 88–95. https://www.sciencedirect.com/science/article/pii/S1877050913001464.

Deng, Jia et al. 2009. “ImageNet: A Large-Scale Hierarchical Image Database.” In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, , 248–55. https://www.researchgate.net/publication/221361415 (September 14, 2019).

Donahue, Jeffrey et al. 2017. “Long-Term Recurrent Convolutional Networks for Visual Recognition and Description.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39(4): 677–91.

Dong, Chao, Chen Change Loy, Kaiming He, and Xiaoou Tang. 2016. “Image Super-Resolution Using Deep Convolutional Networks.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38(2): 295–307.

Dosovitskiy, Alexey et al. 2021. “An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale.” In *International Conference on Learning Representations (ICLR)*, https://arxiv.org/abs/2010.11929.

Eltay, Mohamed, Abdelmalek Zidouri, Irfan Ahmad, and Yousef Elarian. 2022. “Generative Adversarial Network Based Adaptive Data Augmentation for Handwritten Arabic Text Recognition.” *PeerJ Computer Science* 8: 1–22.

Enkvetchakul, Prem, and Olarik Surinta. 2021. “Effective Data Augmentation and Training Techniques for Improving Deep Learning in Plant Leaf Disease Recognition.” *Applied Science and Engineering Progress* 15(3): 1–12.

Fogel, Sharon et al. 2020. “ScrabbleGAN: Semi-Supervised Varying Length Handwritten Text Generation.” In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, , 1–12.

Giménez, Adrià, and Alfons Juan. 2009. “Embedded Bernoulli Mixture HMMs for Handwritten Word Recognition.” In *International Conference on Document Analysis*

and Recognition (ICDAR), , 896–900.

Gondara, Lovedeep. 2016. “Medical Image Denoising Using Convolutional Denoising Autoencoders.” In *IEEE International Conference on Data Mining Workshops (ICDMW)*, , 241–46.

Gonwirat, Sarayut, and Olarik Surinta. 2020. “Improving Recognition of Thai Handwritten Character with Deep Convolutional Neural Networks.” In *International Conference on Information Science and Systems (ICISS)*, , 82–87.

———. 2021. “Optimal Weighted Parameters of Ensemble Convolutional Neural Networks Based on a Differential Evolution Algorithm for Enhancing Pornographic Image Classification.” *Engineering and Applied Science Research* 48(5): 560–69.

Goodfellow, Ian J. et al. 2014. “Generative Adversarial Networks.” In *International Conference on Neural Information Processing System (NIPS)*, , 2672–80. http://arxiv.org/abs/1406.2661.

Graves, Alex;, and Navdeep Jaitly. 2014. “Towards End-To-End Speech Recognition with Recurrent Neural Networks.” In *International Conference on Machine Learning (ICML)*, , 1764–72.

Graves, Alex, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. “Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks.” In *International Conference on Machine Learning (ICML)*, ICML ’06, New York, NY, USA: Association for Computing Machinery, 369–376. https://doi.org/10.1145/1143844.1143891.

Gulrajani, Ishaan et al. 2017. “Improved Training of Wasserstein GANs.” In *International Conference on Neural Information Processing System (NIPS)*, , 1–20. http://arxiv.org/abs/1704.00028.

Guo, Leida et al. 2019. “A Multi-Model Ensemble Method Using CNN and Maximum Correntropy Criterion for Basal Cell Carcinoma and Seborrheic Keratoses Classification.” In *International Joint Conference on Neural Networks (IJCNN)*, , 1–6.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. “Deep Residual Learning for Image Recognition.” In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, , 770–78.

Hochreiter, Sepp, and Jürgen Schmidhuber. 1997. “Long Short-Term Memory.” *Neural Computation* 9(8): 1735–80.

Howard, Andrew G. et al. 2017. “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.” *ArXiv* abs/1704.0. http://arxiv.org/abs/1704.04861.

Hu, Jie, Li Shen, and Gang Sun. 2018. “Squeeze-and-Excitation Networks.” In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE Computer Society, 7132–41.

Huang, Gao, Yixuan Li, et al. 2017. “Snapshot Ensembles: Train 1, Get M for Free.” In *International Conference on Learning Representations (ICLR)*, , 1–14.

Huang, Gao, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2017. “Densely Connected Convolutional Networks.” In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, , 2261–69.

Inkeaw, Papangkorn et al. 2018. “Recognition-Based Character Segmentation for Multi-Level Writing Style.” *International Journal on Document Analysis and Recognition (IJDAR)* 21(1): 21–39. https://doi.org/10.1007/s10032-018-0302-5.

———. 2019. “Recognition of Similar Characters Using Gradient Features of Discriminative Regions.” *Expert Systems with Applications* 134: 120–37.

Ioffe, Sergey, and Christian Szegedy. 2015. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.” In *Proceedings of the International Conference on International Conference on Machine Learning*, Lille, France, 448–56. http://arxiv.org/abs/1502.03167 (September 14, 2019).

Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. 2017. “Image-to-Image Translation with Conditional Adversarial Networks.” In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, , 11125–34.

Ji, Shuiwang, Wei Xu, Ming Yang, and Kai Yu. 2013. “3D Convolutional Neural Networks for Human Action Recognition.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35(1): 221–31.

Johnson, Justin, Alexandre Alahi, and Li Fei-Fei. 2016. “Perceptual Losses for Real-Time Style Transfer and Super-Resolution.” In *European Conference on Computer Vision (ECCV)*, , 1–18. https://arxiv.org/abs/1603.08155.

Joseph, Ferdin Joe John, and Panatchakorn Anantaprayoon. 2018. “Offline Handwritten Thai Character Recognition Using Single Tier Classifier and Local Features.” In *Proceeding of the 3rd International Conference on Information Technology (InCIT)*, Institute of Electrical and Electronics Engineers Inc., 8–11.

Karki, Manohar et al. 2018. “Pixel-Level Reconstruction and Classification for Noisy Handwritten Bangla Characters.” In *International Conference on Frontiers in Handwriting Recognition (ICFHR)*, , 511–16.

Karnewar, Animesh, and Oliver Wang. 2020. “MSG-GAN: Multi-Scale Gradients for Generative Adversarial Networks.” In *Conference on Computer Vision and Pattern Recognition (CVPR)*, , 7796–7805.

Kavitha, B R, and C Srimathi. 2019. “Benchmarking on Offline Handwritten Tamil Character Recognition Using Convolutional Neural Networks.” *Journal of King Saud University - Computer and Information Sciences*: 1–8. https://www.sciencedirect.com/science/article/pii/S1319157819303295.

Keddous, Fekhr Eddine, and Amir Nakib. 2022. “Optimal CNN–Hopfield Network for

Pattern Recognition Based on a Genetic Algorithm." *Algorithms* 15(1).

Khamekhem Jemni, Sana, Mohamed Ali Souibgui, Yousri Kessentini, and Alicia Fornés. 2022. "Enhance to Read Better: A Multi-Task Adversarial Network for Handwritten Document Image Enhancement." *Pattern Recognition* 123: 108370. https://www.sciencedirect.com/science/article/pii/S0031320321005501.

Khan, Asifullah, Anabia Sohail, Umme Zahoora, and Aqsa Saeed Qureshi. 2020. "A Survey of the Recent Architectures of Deep Convolutional Neural Networks." *Artificial Intelligence Review* 53: 5455–5516. https://doi.org/10.1007/s10462-020-09825-6.

Kim, In-Jung, and Xiaohui Xie. 2015. "Handwritten Hangul Recognition Using Deep Convolutional Neural Networks." *International Journal on Document Analysis and Recognition* 18(1): 1–13.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. 2012. "ImageNet Classification with Deep Convolutional Neural Networks." In *Advances in Neural Information Processing Systems 25*, Curran Associates, Inc., 1090–1098. http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf (September 14, 2019).

Kupyn, Orest; et al. 2018. "DeblurGAN: Blind Motion Deblurring Using Conditional Adversarial Networks." In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, , 8183–92.

LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. "Gradient-Based Learning Applied to Document Recognition." *Proceedings of the IEEE* 86(11): 2278–2324. http://ieeexplore.ieee.org/document/726791/ (September 13, 2019).

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. "Deep Learning." *Nature* 521(7553): 436–44.

Lee, C et al. 2014. "Region-Based Discriminative Feature Pooling for Scene Text Recognition." In *Conference on Computer Vision and Pattern Recognition (CVPR)*, , 4050–57.

Liu, Qun, Edward Collier, and Supratik Mukhopadhyay. 2019. "PCGAN-CHAR: Progressively Trained Classifier Generative Adversarial Networks for Classification of Noisy Handwritten Bangla Characters." In *International Conference on Asian Digital Libraries (ICADL)*, , 3–15.

Lue, Hsin-Te et al. 2010. "A Novel Character Segmentation Method for Text Images Captured by Cameras." *Etri Journal* 32: 729–39.

Luo, Canjie, Lianwen Jin, and Zenghui Sun. 2019. "MORAN: A Multi-Object Rectified Attention Network for Scene Text Recognition." *Pattern Recognition* 90: 109–18. https://www.sciencedirect.com/science/article/pii/S0031320319300263.

Marinai, Simone. 2008. "Introduction to Document Analysis and Recognition." In *Studies in Computational Intelligence*, Berlin, Heidelberg: Springer Verlag, 1–20.

Mei, Jianhan et al. 2019. “DeepDeblur: Text Image Recovery from Blur to Sharp.” *Multimedia Tools and Applications* 78(13): 18869–85.

Mishra, Anand, Karteek Alahari, and C V Jawahar. 2016. “Enhancing Energy Minimization Framework for Scene Text Recognition with Top-down Cues.” *Computer Vision and Image Understanding* 145: 30–42. https://doi.org/10.1016/j.cviu.2016.01.002.

Nair, Vinod, and Geoffrey E. Hinton. 2010. “Rectified Linear Units Improve Restricted Boltzmann Machines Vinod Nair.” In *Proceedings of the 27th International Conference on Machine Learning*, , 807–14. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.165.6419 (September 14, 2019).

Noppitak, Sangdaow, and Olarik Surinta. 2022. “DropCyclic: Snapshot Ensemble Convolutional Neural Network Based on a New Learning Rate Schedule for Land Use Classification.” *IEEE Access* 10: 60725–37.

Okafor, Emmanuel et al. 2016. “Comparative Study between Deep Learning and Bag of Visual Words for Wild-Animal Recognition.” In *IEEE Symposium Series on Computational Intelligence (SSCI)*, Institute of Electrical and Electronics Engineers Inc., 1–8.

Pawara, Pornntiwa, Emmanuel Okafor, Olarik Surinta, et al. 2017. “Comparing Local Descriptors and Bags of Visual Words to Deep Convolutional Neural Networks for Plant Recognition.” In *International Conference on Pattern Recognition Applications and Methods (ICPRAM)*, , 1–8.

Pawara, Pornntiwa, Emmanuel Okafor, Lambert Schomaker, and Marco A Wiering. 2017. “Data Augmentation for Plant Classification.” In *Advanced Concepts for Intelligent Vision Systems (ACIVS)*, , 1–12.

Ruder, Sebastian. 2016. “An Overview of Gradient Descent Optimization Algorithms.” In , 1–14. http://arxiv.org/abs/1609.04747 (September 14, 2019).

Sae-Tang, Sutat, and L Methasate. 2004. “Thai Handwritten Character Corpus.” In *IEEE International Symposium on Communications and Information Technology (ISCIT)*, , 486–91.

Sandler, Mark et al. 2018. “MobileNetV2: Inverted Residuals and Linear Bottlenecks.” In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, , 4510–20.

Sawada, Yoshihide;, and Kazuki Kozuka. 2016. “Whole Layers Transfer Learning of Deep Neural Networks for a Small Scale Dataset.” *International Journal of Machine Learning and Computing* 6(1): 27–31.

Schomaker, Lambert et al. 2009. “Monk System for Word Search in Handwritten Manuscript Collections.” https://www.ai.rug.nl/~lambert/Monk-collections-english.html (January 7, 2020).

Sharma, Monika, Abhishek Verma, and Lovekesh Vig. 2018. “Learning to Clean: A GAN

Perspective." In *14th Asian Conference on Computer Vision (ACCV)*, , 174–85.

Shi, Baoguang et al. 2018. "ASTER: An Attentional Scene Text Recognizer with Flexible Rectification." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41(9): 2035–48.

Shi, Baoguang, Xiang Bai, and Cong Yao. 2017. "An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39(11): 2298–2304.

Simonyan, Karen, and Andrew Zisserman. 2015. "Very Deep Convolutional Networks for Large-Scale Image Recognition." In *International Conference on Learning Representations (ICLR)*, , 1–14. https://arxiv.org/abs/1409.1556.

Singh, Sukhdeep, Anuj Sharma, and Vinod Kumar Chauhan. 2021. "Online Handwritten Gurmukhi Word Recognition Using Fine-Tuned Deep Convolutional Neural Network on Offline Features." *Machine Learning with Applications* 5: 100037. https://www.sciencedirect.com/science/article/pii/S2666827021000189.

Souibgui, M A, and Y Kessentini. 2022. "DE-GAN: A Conditional Generative Adversarial Network for Document Enhancement." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44(3): 1180–91.

Souibgui, Mohamed Ali et al. 2022. "DocEnTr: An End-to-End Document Image Enhancement Transformer." *ArXiv* abs/1704.0: 1–7. http://arxiv.org/abs/2201.10252.

Srinilta, C, and S Chatpoch. 2020. "Multi-Task Learning and Thai Handwritten Text Recognition." In *6th International Conference on Engineering, Applied Sciences and Technology (ICEAST)*, , 1–4.

Su, Jiawei, Danilo Vasconcellos Vargas, and Kouichi Sakurai. 2019. "One Pixel Attack for Fooling Deep Neural Networks." *IEEE Transactions on Evolutionary Computation* 23(5): 828–41.

Sujatha, P., and D. Lalitha Bhaskari. 2019. "A Survey on Offline Handwritten Text Recognition of Popular Indian Scripts." *International Journal of Computer Sciences and Engineering* 7(7): 138–49.

Surinta, Olarik, Mahir F. Karaaba, et al. 2015. "Recognizing Handwritten Characters with Local Descriptors and Bags of Visual Words." In *International Conference on Engineering Applications of Neural Networks (EANN)*, , 255–64.

Surinta, Olarik, Mahir F. Karaaba, Lambert R.B. Schomaker, and Marco A. Wiering. 2015. "Recognition of Handwritten Characters Using Local Gradient Feature Descriptors." *Engineering Applications of Artificial Intelligence* 45: 405–14.

Szegedy, Christian et al. 2015. "Going Deeper with Convolutions." In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, , 2322–30.

———. 2016. "Rethinking the Inception Architecture for Computer Vision." *Proceedings of*

IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2016-Decem: 2818–26.

Szegedy, Christian, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. 2017. "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning." In *The Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*, , 4278–84. http://arxiv.org/abs/1602.07261 (September 14, 2019).

Tan, Mingxing, and Quoc V. Le. 2019. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks." In *International Conference on Machine Learning (ICML)*, , 6105–14.

Thanapol, P et al. 2020. "Reducing Overfitting and Improving Generalization in Training Convolutional Neural Network (CNN) under Limited Sample Sizes in Image Recognition." In *5th International Conference on Information Technology (InCIT)*, , 300–305.

Wang, Baiyang, and Diego Klabjan. 2017. "Regularization for Unsupervised Deep Neural Nets." In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, , 2681–87. http://arxiv.org/abs/1608.04426 (September 14, 2019).

Wang, Kai, B Babenko, and S Belongie. 2011. "End-to-End Scene Text Recognition." In *International Conference on Computer Vision (ICCV)*, , 1457–64.

Wang, Tianwei et al. 2019. "Radical Aggregation Network for Few-Shot Offline Handwritten Chinese Character Recognition." *Pattern Recognition Letters* 125: 821–27. https://www.sciencedirect.com/science/article/pii/S0167865519302181.

Wang, Xintao et al. 2018. "ESRGAN : Enhanced Super-Resolution Generative Adversarial Networks." In *European Conference on Computer Vision (ECCV)*, , 1–16.

Wang, Zhengwei, Qi She, and Tomas E. Ward. 2021. "Generative Adversarial Networks in Computer Vision: A Survey and Taxonomy." *ACM Computing Surveys* 54(2): 1–38.

Wu, Bingzhe et al. 2017. "Reducing Overfitting in Deep Convolutional Neural Networks Using Redundancy Regularizer." In *26th International Conference on Artificial Neural Networks (ICANN)*, , 49–55.

Wu, Chunxue, Haiyan Du, Qunhui Wu, and Sheng Zhang. 2020. "Image Text Deblurring Method Based on Generative Adversarial Network." *Electronics* 9(2): 1–14.

Xu, Yan et al. 2018. "End-to-End Subtitle Detection and Recognition for Videos in East Asian Languages via CNN Ensemble." *Signal Processing: Image Communication* 60: 131–43. https://www.sciencedirect.com/science/article/pii/S092359651730173X.

Yan, Hongyu, and Xin Xu. 2020. "End-to-End Video Subtitle Recognition via a Deep Residual Neural Network." *Pattern Recognition Letters* 131: 368–75. https://doi.org/10.1016/j.patrec.2020.01.019.

Zhang, Kai, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. 2017. "Learning Deep CNN

Denoiser Prior for Image Restoration." In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* , 2808–17.

Zhao, Jinyuan et al. 2020. "Adversarial Learning Based Attentional Scene Text Recognizer." *Pattern Recognition Letters* 138: 217–22. https://doi.org/10.1016/j.patrec.2020.07.027.

Zoph, Barret, and Quoc Le. 2017. "Neural Architecture Search with Reinforcement Learning." In *International Conference on Learning Representations (ICLR),* , 1–16. http://arxiv.org/abs/1611.01578 (September 16, 2019).

Zoph, Barret, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. 2018. "Learning Transferable Architectures for Scalable Image Recognition." In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* , 8697–8710.

BIOGRAPHY

NAME	Sarayut Gonwirat
DATE OF BIRTH	9 November 1984
PLACE OF BIRTH	Phu Wiang, Khon Kean, Thailand
ADDRESS	153 Moo 5 Thambon Chumporn Amphoe Moei Wadi Roi Et Thailand
POSITION	Lecturer
PLACE OF WORK	Kalasin University
EDUCATION	2007 Bachelor of Engineering (B.Eng.) Computer Engineering, King Mongkut's Institute of Technology Ladkrabang. 2010 Master of Engineering (M.Eng.) Computer Engineering, King Mongkut's Institute of Technology Ladkrabang. 2022 Doctor of Philosophy (Ph.D.) Information Thechnology, Mahasarakham University.
Research output	Gonwirat, S., & Surinta, O. (2020). Improving Recognition of Thai Handwritten Character with Deep Convolutional Neural Networks. International Conference on Information Science and Systems (ICISS). Chompookham, T., Gonwirat, S., Lata, S., Phiphiphatphaisit, S., & Surinta, O. (2020). Plant Leaf Image Recognition using Multiple-grid Based Local Descriptor and Dimensionality Reduction Approach. International Conference on Information Science and Systems (ICISS), 72–77. https://doi.org/10.1145/3388176.3388180 Noppitak, S., Gonwirat, S., & Surinta, O. (2020). Instance Segmentation of Water Body from Aerial Image using Mask Region-based Convolutional Neural Network. International Conference on Information Science and Systems (ICISS), 61–66. https://doi.org/10.1145/3388176.3388184 Gonwirat, S., & Surinta, O. (2022). CycleAugment: Efficient Data Augmentation Strategy for Handwritten

Text Recognition in Historical Document Images. Engineering and Applied Science Research, 49(4), pages 505-520.
https://doi.org/10.14456/EASR.2022.50

Gonwirat, S., & Surinta, O. (2022). DeblurGAN-CNN: Effective Image Denoising and Recognition for Noisy Handwritten Characters. IEEE Access, 10, 90133-90148.
https://doi.org/10.1109/ACCESS.2022.3201560