ICIC Express Letters
Volume **18**, Number **5**, May **2024**

ICIC International ©2024 ISSN 1881-803X
pp. **433–442**

DETECTING AND RECOGNIZING THAI TEXTS IN SCENE AND GRAPHIC TEXT IMAGE

Kraingsak Rukpukdee and Phatthanaphong Chomphuwiset*

Computer Science Department
Faculty of Informatics
Mahasarakham University
Khamreang, Kantarawichai, Maha Sarakham 44150, Thailand
66011294002@msu.ac.th; *Corresponding author: phatthanaphong.c@msu.ac.th

Received June 2023; accepted September 2023

Abstract. *This work focuses on the development of detection methods and character recognition of Thai texts in scene and graphic images. The proposed technique consists of two main tasks: text detection and text recognition. In the text detection phase, a Maximally Stable Extremal Regions (MSER) algorithm and a Fast Region-based Convolutional Neural Network method (Faster R-CNN) with various backbones were evaluated. The MSER algorithm achieved a mean Average Precision (mAP) of 0.80 at a parameter value of $\Delta = 0.5$, while Faster R-CNN with VGG-16 as the backbone achieved the highest mAP of 0.85 at a confidence score threshold of 0.5. For the text recognition, a comparison between network configurations with and without additional dense layers revealed that the configuration with additional dense layers achieved better performance, with a lower Character Error Rate (CER) of 25.2% and higher precision, recall, and F1-score.*
Keywords: Text detection, Text recognition, Maximally Stable Extremal Regions (MSER), Faster R-CNN, Hybrid networks

1. Introduction.

Detecting and recognizing of text in images have gained substantial attention and interest in the research community. Their potential applications span various domains, notably including augmented and virtual reality [1]. However, this task poses a considerable challenge, as images can contain diverse and substantial object types, including text fragments and non-text artifacts [2]. In order to tackle these challenges in text detection, researchers have introduced various computer vision and machine vision methods in the past years. Threshold-based techniques have been widely used and considered pioneers in detecting objects, including text regions, within images [3, 4]. In general, text within images often consists of characters represented by a relatively uniform pixel intensity. Therefore, the identification of text regions can be achieved by applying either a static or dynamic thresholding technique. However, the threshold technique is sensitive to the image variation and cannot cope with the uncertainty of the image illumination [5]. Connected component analysis methods have been proposed and utilized [5, 6, 7]. Among these techniques, Stroke Width Transforms (SWT) have been introduced and proven effective [6, 8, 9]. SWT is capable of detecting diverse text appearances and different text properties, such as fonts, language types, colors, and illumination changes in images. However, it can produce poor segmentation results if inaccurate stroke maps are derived during the process [9]. To overcome this limitation, edge-based techniques, for instance, the Hough transform, have been utilized [10].

Recently, deep learning techniques have made significant advancements in various computer vision tasks, including text segmentation [11, 12, 13, 14, 15]. Convolutional Neural Networks (CNNs) have been particularly effective in learning and segmenting text regions

DOI: 10.24507/icicel.18.05.433

in images. Zhao et al. proposed a technique that divides an image into small window and performed a binary classification (text and non-text) using a CNN. Without local contexts, a post-processing is required to aggregate the classification results for segmenting homogeneous text regions. A two-step process using deep learning was then proposed [15]. In the first step, a CNN is used for low-level feature extraction. In the second step, the extracted features are merged, and post-processing techniques, such as Locality-Aware Non-Maximum Suppression (LANMS), are applied to refining the text region segmentation. Another approach is the use of a single-shot detector with region attention, as proposed by He et al. [13]. This method incorporates attention mechanisms to highlight text regions during the detection process. Additionally, inception-based models have been utilized to detect multi-scale text in natural scene images [16], and inception models with multi-orientation capabilities have been employed for text detection [14]. Overall, these deep learning techniques, particularly CNN-based models and their variants, have demonstrated remarkable progress in accurately detecting and segmenting text regions in images.

In addition to text detection, text recognition also plays a crucial role in the overall process. Conventionally, in text recognition methods, the characters in text images are first segmented and then recognized using machine learning techniques. Descriptive features such as local primitive structure, shape, and morphology are commonly used to represent each character [17, 18, 19]. Subsequently, a classification process is applied to performing the recognition task. However, an alternative approach involves utilizing Recurrent Neural Networks (RNNs) for recognizing text sequences instead of individual characters [20]. Unlike conventional approaches that rely on accurate character segmentation, RNNs consider the contextual information present in the sequence of image components. This enables them to capture the overall structure and semantics of the text, enhancing the recognition performance. To address this, a hybrid method is proposed in this work, which combines feature extraction and learning the dependencies of the components within the image to predict the corresponding texts. The hybrid method consists of two deep learning structures: (i) a CNN network implemented to extract feature maps from the input images, and (ii) an RNN network incorporated to learn the dependencies of the extracted feature maps. By leveraging the strengths of both CNN and RNN architectures, this hybrid method aims to improve the accuracy of text recognition in images. The feature extraction capability of the CNN enhances the representation of the input images, while the RNN captures the sequential dependencies of the text components.

This paper is organized as follows. Section 2 provides an overview of our approach for detecting and recognizing scene and graphic text in images. We applied both conventional MSER (Maximally Stable Extremal Regions) and deep learning-based methods for text detection task. For text recognition, we employ a hybrid method that predicts character sequences in the images. Additionally, in this section, we provide detailed information about the images and data domain used in our study. In Section 3, we present the experimental setup and methodology used to evaluate the performance of our text detection and recognition tasks. Subsequently, in Section 4, we present the results and analysis of our experiments. We discuss the performance of the text detection and recognition methods, comparing the outcomes of the conventional MSER and deep learning-based approaches.

2. Research Method.

This work presents a comprehensive method for detecting and recognizing text in images, which consists of two main stages: text detection and text recognition. In this work, the text detection process employs two distinct approaches to facilitate a comparative analysis. Firstly, the MSER algorithm is employed to identify potential text regions in the image. Then, a post-processing step is performed to merge similar regions and eliminate false positives. For a deep learning-based technique, Faster R-CNN is implemented to classify and refine these text regions. After the detection, we describe text recognition process. A hybrid deep learning network is utilized. This network combines a convolutional neural network (CNN) and a recurrent neural network (RNN). The CNN extracts high-level features from the image, capturing spatial information and identifying distinctive patterns related to text. These features are then fed into the RNN, which processes the sequence of features to predict the character sequences present in the text. An overview of the entire process is depicted in Figure 1.

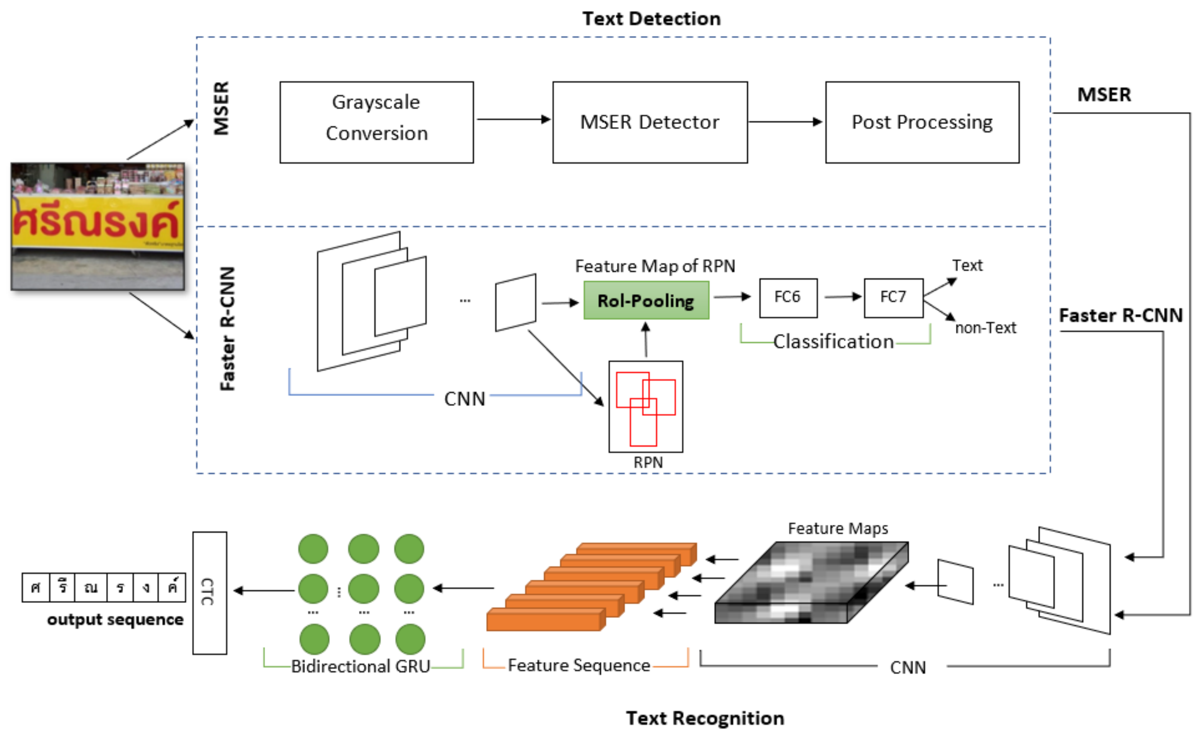

FIGURE 1. Overview of the process for detecting and recognizing text in both scene and graphic images

Therefore, this section will begin by providing an explanation of the images and data domain used in our study, followed by a description of the text detection and recognition processes in a subsequent section.

2.1. **Image data.** In this work, the image dataset used was collected manually. The images are billboard images found in various scenes across the Ubon Ratchathani province, Thailand. The images are divided into two groups, i.e., (i) scene images and (ii) graphic images. The image dataset used in the study consists of Thai language text regions, with each image containing at least one explicit text region. For the sake of simplicity, the dataset was limited to 1,700 images, each of which was chosen to contain at least a single text region, resulting in a total of 1,912 text regions across the dataset. To prepare a set of ground-truth data for the study, each image was manually segmented using the annotation tool ImageJ [21] and Roboflow [22], with the position of the text region specified as a bounding box. In addition, for the recognition task, each text region in the images was associated with a text label, which was also performed manually.

2.2. **Text region detection.** This section explains the text region detection process carried out in this work. We apply two techniques to performing the detection task (i.e., MSER and Faster R-CNN).

2.2.1. *MSER.* We derive from the original method and carry out the detection process, followed by a pre-processing task. In general, the intensity variation of extremal region (R_i) in an Image I can be defined as follows:

$$I(R_i) = \frac{|R_{i+\Delta} - R_{i-\Delta}|}{|R_i|} \tag{1}$$

where, $|R|$ is the area of the extremal region R, $R_{+\Delta}$ denotes the extremal region which is Δ levels containing R, and $|R_{+\Delta} - R|$ is the area difference of the two regions. The stability of an extremal region R may depend on the inverse of the relative area variation of R; if the intensity level is increased by the value Δ. After detecting MSER regions, we proceed to merge the sets of detected fragments to form text regions. This is achieved by applying the MeanShift algorithm, as described below.

```
Merge Algorithm:
1. Divide the image into 6 non-overlapping horizontal regions.
2. For each character bounding box in *character bounding boxes*:
    (a) Compute the centroid of the bounding box.
    (b) Determine the region in which the centroid falls.
    (c) Generate a hot vector representing the spatial location: for example, $[1, 0, 0, 0,$
        $0, 0]$ represents a bounding box located in the first region.
3. Use the aspect ratio and the vector representing the spatial location of the bounding
   box as the features.
4. Apply the MeanShift algorithm using the generated hot vectors and aspect ratios
   as features to clustering the character bounding boxes.
5. Merge the character bounding boxes within each cluster to form the text regions,
   as described in the previous algorithms.
6. Return the list of merged text regions: *text regions*.
```

To minimize false negatives (non-text regions), we employ a classification-based technique using the Bayes classifier. The Histogram of Oriented Gradients is utilized as the feature descriptor for each text region. During the non-text region rejection process, a rule-based technique is applied. Let $P(\text{text})$ and $P(\neg\text{text})$ represent the prior probabilities of a region being text and non-text, respectively. According to Bayes' theorem, the posterior probability of a region being sensitive to non-text can be expressed as

$$P(\neg\text{text}|O) = \frac{P(O|\neg\text{text})P(\neg\text{text})}{P(O)} \tag{2}$$

where

$$P(O) = P(O|\text{text})P(\text{text}) + P(O|\neg\text{text})P(\neg\text{text}) \tag{3}$$

A non-text region is rejected if $P(\neg\text{text}|O) \geq \epsilon$, where ϵ is a predefined threshold. An example of the detection results is demonstrated in Figure 2, including the sequential outcomes of the explained detection method, encompassing MSER, the identification of candidate text regions, and ultimately, the final text region detection.

2.2.2. *Faster R-CNN.* The technique first derives Region Proposals using a Region Proposal Network (RPN). The RPN generates potential text region proposals in the image by proposing candidate bounding boxes that might contain text. It takes the input image I and produces a set of region proposals, with each proposal represented by a bounding box. Then, a Region of Interest (RoI) pooling layer is applied. Consequently, the resulting pooled features are concatenated into a fixed-length feature vector. This feature vector is then fed into the fully connected layers for object classification and bounding box regression.

In the classification process, the model predicts the probability of each RoI being text or non-text. A softmax activation is applied to the output of the classification layer. Let us denote the output probabilities as P_c, where P_c represents the probability distribution

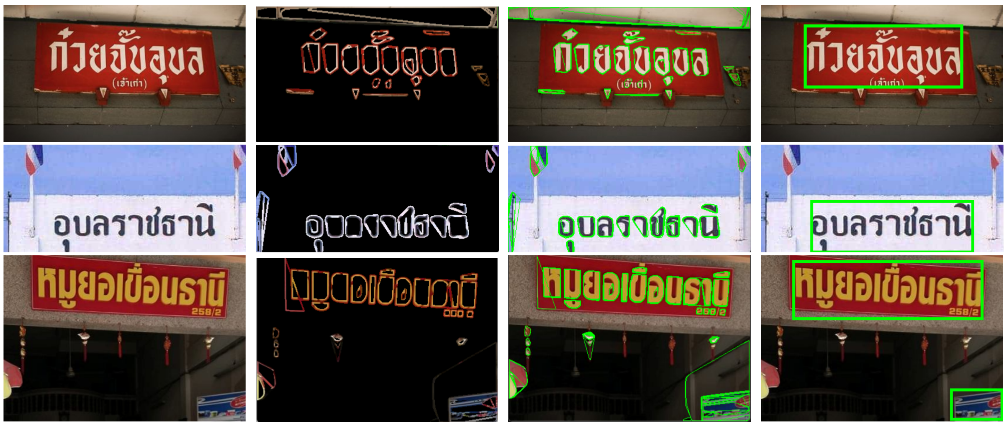

FIGURE 2. Example of the detection results: *Column 1* displays the input images, *Column 2* presents the MSER in the images, *Column 3* shows the detected candidate texts, and *Column 4* presents the identified text regions

over classes (text or background) for each RoI. The classification loss L_{cls} is computed using the following equation:

$$L_{\text{cls}} = -\sum_i y_i \log(P_{c_i}), \tag{4}$$

where y_i represents the ground truth class label (1 for text, 0 for non-text) for RoI i, and P_{c_i} represents the predicted probability of the correct class.

On the other hand, the regression procedure predicts the refined coordinates of the bounding box for each RoI. It outputs bounding box positions, representing the offsets in terms of (x, y, w, h) from the original proposal. These offsets are used to refine the initial bounding box. Thus, given the predicted offsets as P_{reg}, the regression loss L_{reg} is computed as follows:

$$L_{\text{reg}} = \sum_i \mathrm{S}_{L1}\left(P_{\text{reg}_i} - \hat{P}_{\text{reg}_i}\right), \tag{5}$$

where $\hat{P}_{\text{reg}_i}$ represents the ground truth bounding box offset for RoI_i, and S_{L1} is the smooth L1 loss function.

2.3. **Text recognition.** After detecting the text regions in the images using the previous process, the next step is to perform a recognition task. This task is accomplished using a hybrid method that combines a CNN network for extracting feature maps from the images and an RNN network for capturing dependencies within those feature maps. During the training process, the model is trained on the predicted character sequence using the Connectionist Temporal Classification (CTC) loss scheme. A diagram illustrating this process is shown in Figure 3.

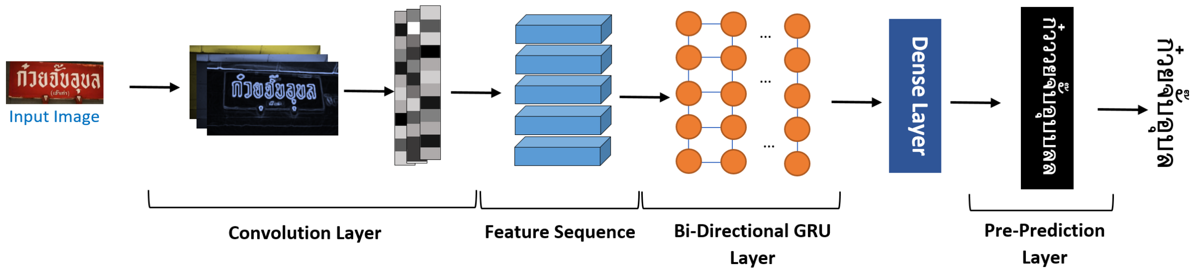

FIGURE 3. The hybrid network for the text recognition process used in this work

Let T be the text region that has been detected from the previous process. The Feature maps (F) for a specific region can be obtained using a convolution layer, which produces a fixed-length vector. We denote the feature map as a function of P, where

$$P(T, F) = [p_1, p_2, \ldots, p_k], \tag{6}$$

where $p_1, p_2, \ldots, p_k$ are k fixed-length of feature map vectors in T. Then, the feature sequences (P') are generated as

$$P' = p'_1 \oplus p'_2 \oplus \cdots \oplus p'_n \tag{7}$$

where n represents the length of the sequence. Each p'_i is a fixed-length vector obtained after a reconstructing process of the feature map vectors. Then, the feature sequence P' is fed into a Bidirectional Gated Recurrent Unit (BiGRU) layer. At each time step t in the forward direction, the forward GRU updates its hidden state using the input feature vector p'_t, the bias for the update gate (b_z), the bias for the reset gate (b_r), the bias for the hidden state transformation (b_h), and the previous forward hidden state h_t^{forward}, which is as follows:

$$\begin{aligned}
z_t^{\text{forward}} &= \sigma\Big(W_z^{\text{forward}} \cdot \Big[p'_t, h_{t-1}^{\text{forward}}\Big] + b_z^{\text{forward}}\Big) \\
r_t^{\text{forward}} &= \sigma\Big(W_r^{\text{forward}} \cdot \Big[p'_t, h_{t-1}^{\text{forward}}\Big] + b_r^{\text{forward}}\Big) \\
\tilde{h}_t^{\text{forward}} &= \tanh\Big(W_h^{\text{forward}} \cdot \Big[p'_t, r_t^{\text{forward}} \odot h_{t-1}^{\text{forward}}\Big] + b_h^{\text{forward}}\Big) \\
h_t^{\text{forward}} &= \Big(1 - z_t^{\text{forward}}\Big) \odot h_{t-1}^{\text{forward}} + z_t^{\text{forward}} \odot \tilde{h}_t^{\text{forward}}
\end{aligned}$$

Similarly, at each time step t in the backward direction, the backward GRU updates its hidden state using the input feature vector p'_t and the previous backward hidden state h_t^{backward}:

$$\begin{aligned}
z_t^{\text{backward}} &= \sigma\Big(W_z^{\text{backward}} \cdot \Big[p'_t, h_{t+1}^{\text{backward}}\Big] + b_z^{\text{backward}}\Big) \\
r_t^{\text{backward}} &= \sigma\Big(W_r^{\text{backward}} \cdot \Big[p'_t, h_{t+1}^{\text{backward}}\Big] + b_r^{\text{backward}}\Big) \\
\tilde{h}_t^{\text{backward}} &= \tanh\Big(W_h^{\text{backward}} \cdot \Big[p'_t, r_t^{\text{backward}} \odot h_{t+1}^{\text{backward}}\Big] + b_h^{\text{backward}}\Big) \\
h_t^{\text{backward}} &= \Big(1 - z_t^{\text{backward}}\Big) \odot h_{t+1}^{\text{backward}} + z_t^{\text{backward}} \odot \tilde{h}_t^{\text{backward}}
\end{aligned}$$

The final output of the BiGRU can be obtained by combining the hidden states from both the forward and backward GRUs at each time step, as follows:

$$h_t^{\text{BiGRU}} = \Big[h_t^{\text{forward}}, h_t^{\text{backward}}\Big] \tag{8}$$

h_t^{BiGRU} denotes the final hidden state of the BiGRU at time step t, which incorporates information from both the forward and backward directions.

To process the output of the BiGRU layers, in this work, we feed it to a dense layer. The dense layer is implemented to transform the input into a format suitable for the target task. It can be represented as

$$Z = H \cdot W_d + b_d \tag{9}$$

H represents the output tensor from the BiGRU layers, W_d is the weight matrix of the dense layer, and b_d is the bias vector of the dense layer. Then, at each time step, we apply the softmax activation function to the tensor Z to obtain a probability distribution over possible output labels ($\mathcal{L}$), which obtains the output (y_t) after the softmax activation.

In the pre-prediction layer, the Connectionist Temporal Classification (CTC) algorithm is utilized to map input sequences to their corresponding output labels. CTC incorporates both the forward and backward algorithms to effectively manipulate sequence labeling tasks. The forward algorithm determines the probability of transitioning to each label at each time step. Given the forward variables as $\alpha_t(l)$ representing the probability of being in label l at time step t, the forward variables are calculated as follows:

$$\alpha_1(l) = y_1(l) \tag{10}$$

$$\alpha_t(l) = \left(\sum_{k\in\mathcal{L}} \alpha_{t-1}(k) + \alpha_{t-1}(\text{blank})\right) \cdot y_t(l) \text{ for } t > 1 \tag{11}$$

The backward algorithm calculates the probability of advancing from each label at each time step to the end of the sequence. We denote $\beta_t(l)$ as the backward variables, representing the probability of transitioning from label l at time step t to the end. The backward variables are calculated as follows:

$$\beta_T(l) = 1 \text{ for } l = \text{blank} \tag{12}$$

$$\beta_T(l) = 0 \text{ for } l \neq \text{blank} \tag{13}$$

$$\beta_t(l) = \left(\sum_{k\in\mathcal{L}} \beta_{t+1}(k) + \beta_{t+1}(\text{blank})\right) \cdot y_{t+1}(l) \text{ for } t < T \tag{14}$$

where T is the total time steps in the sequence.

Using the forward and backward variables, we can calculate the CTC loss. The CTC loss measures the difference between the predicted sequence and the target sequence, which can be computed as follows:

$$L_{\text{CTC}} = -\log\left(\sum_{l\in\mathcal{L}} \left(\alpha_T(l)\cdot\beta_T(l) + \alpha_T(\text{blank})\cdot\beta_T(\text{blank})\right)\right) \tag{15}$$

We can feed the concatenated hidden states through the softmax activation and use the forward-backward algorithm to calculate the CTC loss. The forward-backward algorithm efficiently considers all possible label alignments and accounts for repeated and blank labels, enabling the model to learn the alignment between the input and target sequences effectively.

3. **Experiment and Result.**

The previous section explains the methodology for detecting and recognizing in the images used in this work. To assess the effectiveness of the technique, two distinct experiments were conducted to evaluate the performance of the detection algorithm and the recognition approach, respectively.

3.1. **Text detection.**

In the detection, we initially implemented the MSER. We examined the best performance from the detection by varying the Δ parameter. In the evaluation, we utilized mean Average Precision (mAP) as a metric to assess the detection performance, as shown in Table 1.

Table 1. Results obtained by MSER for the detection process. The parameter Δ is varied to observe the results of the detection.

	Δ							
	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
MSER	0.72	0.75	0.79	0.80	0.79	0.79	0.67	0.67

Table 1 demonstrates the performance of MSER at different Δ values ranging from 0.2 to 0.9. The highest mAP value of 0.80 was achieved at $\Delta = 0.5$, indicating the best detection performance.

In addition to MSER, we conduct another experiment to compare the detection performance using Faster R-CNN. We implemented two back-bone for the detection task, i.e., (i) Resnet101 and (ii) inception-resnet V2. In the training process, we used default parameters of both network. The train images were augmented to increase the size of the training set. We implemented 3 augmentation functions, i.e., (i) rotation, (ii) scale and (iii) sheer. The training was carried in 100 epochs. The testing images (same as the previous experiment) were fed to the network and determine the results of different confident scores. The results are shown in Table 2.

Table 2. mAP results for text detection using Faster R-CNN with different backbones and confident threshold scores

Backbones	**Confidence score**		
	0.5	0.6	0.7
VGG-16	0.85	0.80	0.75
ResNet-50	0.81	0.75	0.73
MobileNet	0.78	0.76	0.73

Table 2 presents the experimental results for text detection using Faster R-CNN with 3 backbone architectures and confidence score thresholds. The results indicated that the choice of backbone architecture had a noticeable impact on the overall performance. Among the tested backbones, VGG-16 achieved the highest mAP values across all confidence score thresholds, with a maximum of 0.85 at a threshold of 0.5. ResNet-50 and MobileNet followed with mAP values of 0.81 and 0.78, respectively, at the same threshold. As the confidence score threshold increased from 0.5 to 0.7, the mAP values generally decreased, indicating that stricter thresholds resulted in fewer detected text objects.

3.2. **Text recoginition.** After completing the detection process, we proceeded with the recognition procedure. In this experiment, we employed the text recognition technique explained in Section 2.3. To begin, we process the training with 100 training epochs, followed by testing on the evaluation dataset. To evaluate the accuracy of the Optical Character Recognition (OCR) output, we utilized the Character Error Rate (CER) metric. CER is based on the concept of Levenshtein distance, which measures the minimum number of character-level operations required to transform the ground truth text (also known as the reference text) into the OCR output.

$$CER = \frac{S + D + I}{N} \tag{16}$$

where S is number of substitutions, D is number of deletions, I is number of insertions, and N is number of characters in reference text (ground truth). We conducted a comparison between two network configurations: (i) a configuration with additional dense layers and (ii) a configuration without additional dense layers. The results of this comparison are summarized in Table 3 and visualized in Figure 4.

The configuration with additional dense layers outperformed the configuration without dense layers in terms of CER, precision, recall, and F1-score. The CER was lower (25.2%) for the configuration with dense layers compared to the configuration without

Table 3. Comparison of results for different network configurations

Metric	**Configuration 1**	**Configuration 2**
	10 additional dense layer	No dense layers
CER (%)	25.2	27.8
Precision (%)	86	82
Recall (%)	85	82
F1-score (%)	86	85

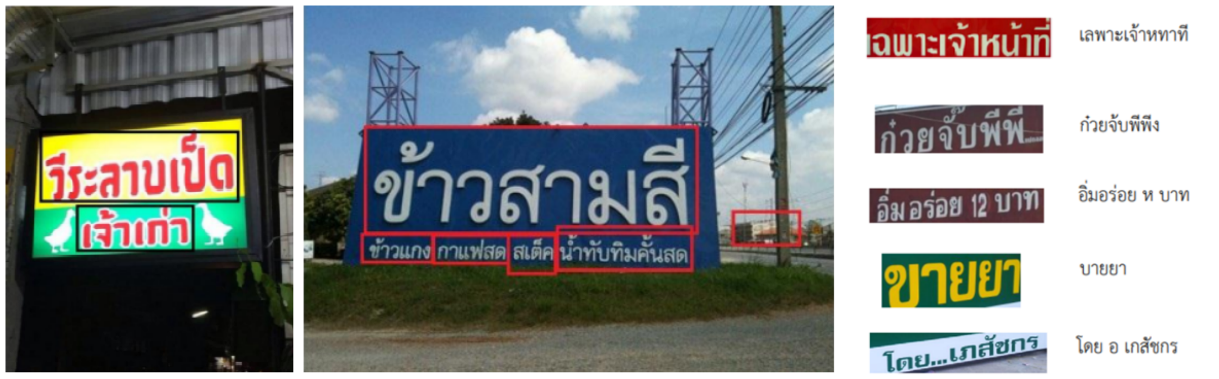

FIGURE 4. Examples of text detection and recognition in natural scene images

dense layers (27.8%). Additionally, the configuration with dense layers achieved higher precision (86%), recall (85%), and F1-score (86%) compared to the configuration without dense layers (precision: 82%, recall: 82%, F1-score: 85%). This demonstrates that the inclusion of additional dense layers improved the accuracy and overall performance of the text recognition system. See Figure 4 for an example of the results.

3.3. **Insight discussion.**

The experimental results (Tables 1 and 2) demonstrate the performance of the text detection and recognition techniques used in this work. The MSER algorithm achieves a promising mean Average Precision (mAP) of 0.80 with $\Delta = 0.5$, although detection performance varied with different parameter settings, suggesting the need for careful tuning. Among the Faster R-CNN approaches, VGG-16 shows superior performance in text detection, with higher mAP values across different confidence score thresholds. However, stricter thresholds result in fewer detected text objects.

For text recognition, the configuration with additional dense layers outperforms the others, achieving a lower Character Error Rate (CER) of 25.2% and higher precision, recall, and F1-score. This indicates the improvement in accuracy and overall performance with the inclusion of dense layers. There are limitations and potential areas for improvement. The evaluation was conducted on specific datasets, and the generalizability of the results should be further investigated. Optimizing parameter values and exploring advanced techniques for text detection and recognition are necessary. Handling complex images with distorted or overlapping text remains a challenge. Future research should address these limitations to enhance the robustness and accuracy of text detection and recognition in diverse real-world scenarios.

4. **Conclusion.**

The work emphasizes the development of detection methods and Thai character recognition in billboards. The proposed technique comprises two main processes: text detection in images and recognition of the detected texts. Through experimental evaluations, the effectiveness of different algorithms and configurations was assessed.

In the text detection phase, the MSER algorithm demonstrated promising performance, achieving a mean Average Precision (mAP) of 0.80 at a parameter value of $\Delta = 0.5$. However, it is important to note that parameter tuning played a crucial role in obtaining optimal detection results. Additionally, the Faster R-CNN approach, with VGG-16 as the backbone, showed superior performance in text detection compared to other backbones, achieving an mAP value of 0.85 at a confidence score threshold of 0.5.

For text recognition, the comparison between network configurations with and without additional dense layers revealed that the configuration with additional dense layers yielded better results. It achieved a lower Character Error Rate (CER) of 25.2% and higher precision, recall, and F1-score, indicating improved accuracy and overall performance of the text recognition system.

Future work should investigate the generalizability of the proposed technique to diverse datasets. Optimization of parameters and exploration of advanced techniques can enhance text detection and recognition performance. Addressing the challenge of handling complex images with distorted or overlapping text requires further research and development.

REFERENCES

[1] M. Petter, V. Fragoso, M. Turk and C. Baur, Automatic text detection for mobile augmented reality translation, *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp.48-55, 2011.

[2] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, D. J. Wu and A. Y. Ng, Text detection and character recognition in scene images with unsupervised feature learning, *2011 International Conference on Document Analysis and Recognition*, pp.440-445, 2011.

[3] J. Gllavata, R. Ewerth and B. Freisleben, Finding text in images via local thresholding, *Proc. of the 3rd IEEE International Symposium on Signal Processing and Information Technology (IEEE Cat. No. 03EX795)*, pp.539-542, 2003.

[4] E. Y. Du, C. Chang and P. D. Thouin, Thresholding video images for text detection, *2002 Intl. Conf. on Pattern Recognition*, vol.3, pp.919-922, DOI: 10.1109/ICPR.2002.1048184, 2002.

[5] Z. Ramadhan and D. Alzubaydi, Text detection in natural image by connected component labeling, *Al-Mustansiriyah Journal of Science*, vol.30, 111, 2019.

[6] B. Epshtein, E. Ofek and Y. Wexler, Detecting text in natural scenes with stroke width transform, *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp.2963-2970, 2010.

[7] I.-S. Oh and J.-S. Lee, Smooth stroke width transform for text detection, *Artificial Intelligence: Methodology, Systems, and Applications*, vol.9, pp.183-191, 2016.

[8] S. Karthikeyan, V. Jagadeesh and B. S. Manjunath, Learning bottom-up text attention maps for text detection using stroke width transform, *2013 IEEE International Conference on Image Processing*, pp.3312-3316, 2013.

[9] T. Titijaroonroj, Modified stroke width transform for Thai text detection, *2018 International Conference on Information Technology (InCIT)*, pp.1-5, 2018.

[10] S. Saha, S. Basu, M. Nasipuri and D. Basu, A hough transform based technique for text segmentation, *Journal of Computing*, vol.2, pp.134-141, 2010.

[11] D. Kavitha and V. Radha, TexNet: A deep convolutional neural network model to recognise text in natural scene images, *Journal of Engineering Science and Technology*, vol.16, pp.1782-1799, 2021.

[12] H. Zhao, Y. Hu and J. Zhang, Reading text in natural scene images via deep neural networks, *2017 4th IAPR Asian Conference on Pattern Recognition (ACPR)*, pp.43-48, 2017.

[13] P. He, W. Huang, H. Tong, Q. Zhu, Y. Qiao and X. Li, Single shot text detector with regional attention, *2017 IEEE International Conference on Computer Vision (ICCV)*, pp.3066-3074, 2017.

[14] Q. Yang, M. Cheng, W. Zhou, Y. Chen, M. Qiu and W. Lin, IncepText: A new inception-text module with deformable PSROI pooling for multi-oriented scene text detection, *The 27th International Joint Conference on Artificial Intelligence (IJCAI-18)*, pp.1071-1077, 2018.

[15] Y. Zhang, Y. Huang, D. Zhao, C.-H. Wu, W. Ip and K. Yung, A scene text detector based on deep feature merging, *Multimedia Tools and Applications*, vol.80, pp.1-12, 2021.

[16] H. Zhang, J. Liu and T. Chen, Scene text detection with inception text proposal generation module, *Proc. of the 2019 11th International Conference on Machine Learning and Computing (ICMLC'19)*, pp.456-460, 2019.

[17] D. Singh, M. A. Khan, A. Bansal and N. Bansal, An application of SVM in character recognition with chain code, *2015 Communication, Control and Intelligent Systems (CCIS)*, pp.167-171, 2015.

[18] N. Dojčinovič, I. Mihajlovič, J. Jokovič, V. Markovič and B. Milovanovič, Neural network based optical character recognition system, *The 11th Symposium on Neural Network Applications in Electrical Engineering*, pp.111-114, 2012.

[19] A. K. Bhunia, G. Kumar, P. P. Roy, B. Raman and U. Pal, Text recognition in scene image and video frame using color channel selection, *Multimedia Tools and Applications*, vol.77, pp.8551-8578, 2017.

[20] B. Rajyagor and R. Rakholia, Isolated gujarati handwritten character recognition (HCR) using deep learning (LSTM), *2021 4th International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, pp.1-6, 2021.

[21] C. Schneider, W. Rasband and K. Eliceiri, NIH Image to ImageJ: 25 years of image analysis, *Nature Methods*, vol.9, 2012.

[22] B. Dwyer and J. Nelson, *Roboflow (Version 1.0)*, Software, 2022.