B.Comp. Dissertation

Benchmarking and Improving OCR Systems for Southeast Asian Languages

By

Qiu Jiasheng, Jason

Department of Computer Science

School of Computing

National University of Singapore

2024/2025

B.Comp. Dissertation

Benchmarking and Improving OCR Systems for Southeast Asian Languages

By

Qiu Jiasheng, Jason

Department of Computer Science

School of Computing

National University of Singapore

2024/2025

Project ID: H0792230

Supervisor: A/P Min-Yen Kan

Advisor: Tongyao Zhu

Deliverables:

Report: 1 Volume

Abstract

While Optical Character Recognition (OCR) has been widely studied for high-resource languages such as English and Chinese, the efficacy and limitations of OCR models on Southeast Asian (SEA) languages remain largely unexplored. This study aims to bridge this gap by evaluating OCR technologies for SEA languages and exploring script-specific challenges. We propose a pipeline to collect textual data from Wikipedia and benchmark open-source OCR tools. Additionally, we demonstrate the potential of fine-tuning existing models on SEA languages, aiming to expand OCR capabilities for these languages.

Subject Descriptors:

H.3.3 Information Search and Retrieval

I.2.7 Natural Language Processing

I.2.10 Vision and Scene Understanding

Keywords:

Optical Character Recognition, Southeast Asian Languages

Implementation Software and Hardware:

Python, Tesseract, EasyOCR

Acknowledgements

I would like to thank my supervisor, A/P Kan Min-Yen, and my advisor, Tongyao Zhu, for their invaluable guidance and mentorship. Their encouragement and constructive guidance have been a significant source of inspiration throughout the project.

List of Figures

3.1 Pipeline for data collection from Wikipedia 13

3.2 Pipeline for OCR evaluation 14

3.3 Sample synthetic image with different types of noise 19

4.1 Error classification by character type for English articles 22

4.2 Error classification by character type for Indonesian articles 22

4.3 Error classification by character type for Vietnamese articles 23

4.4 Error classification by character type for Thai articles 23

4.5 Performance of fine-tuned GOT on Vietnamese 24

4.6 Performance of fine-tuned GOT on Thai 24

List of Tables

3.1 Benchmarked Languages 9

3.2 Benchmarked OCR Systems 10

3.3 Types of noise applied 17

3.4 Character types used for error classification 18

4.1 Average CER and WER on real-world data 20

4.2 Average CER and WER on synthetic data 21

4.3 Average OCR runtime per page (seconds) 21

A.1 Dataset of 100 Wikipedia articles used for benchmarking real-world data in Experiment 1 (See Section 3.2) 31

Table of Contents

Title **i**

Abstract **ii**

Acknowledgements **iii**

List of Figures **iii**

List of Tables **iv**

Table of Contents **v**

1 Introduction **1**

2 Related Work **3**

2.1 Overview of OCR Systems 3

2.1.1 Evolution of OCR Models 3

2.2 Benchmarking OCR on Low-resource Languages 5

2.3 Using Synthetic Data for OCR Evaluation 6

2.4 Fine-tuning OCR Systems 6

3 Methodology **8**

3.1 Experiment Setup 8

3.1.1 Languages 8

3.1.2 Data Source 9

3.1.3 OCR Systems 10

3.1.4 Evaluation Metrics 11
3.2 Experiment 1: Benchmarking on Real-world Data 12
3.2.1 Data Collection 12
3.2.2 OCR Evaluation 14
3.3 Experiment 2: Benchmarking on Synthetic Data 15
3.3.1 Synthetic Data Generation 16
3.3.2 OCR Evaluation on Different Types of Noise 16
3.3.3 Error Classification by Character Type 17
3.4 Experiment 3: Fine-tuning for Vietnamese and Thai 18

4 Results and Analysis **20**
4.1 RQ1: How do popular OCR tools perform on SEA scripts? 20
4.1.1 OCR Accuracy on Real-world Data 20
4.1.2 OCR Accuracy on Synthetic Data 21
4.1.3 Runtime 21
4.2 RQ2: What script-related challenges affect OCR accuracy on SEA languages? 21
4.3 RQ3: What techniques and recommendations can enhance OCR accuracy on SEA languages? 21

5 Discussion **25**

6 Conclusion **26**

References **27**

A Wikipedia Dataset **31**

Chapter 1

Introduction

Current research in Natural Language Processing (NLP) is heavily concentrated on only 20 of the 7,000 languages in the world (Magueresse et al., 2020). In particular, Southeast Asia (SEA) is home to over 1,000 languages but remains a relatively under-researched region in NLP (Aji et al., 2023). A similar trend can be observed in Optical Character Recognition (OCR) research, where the focus is predominantly on high-resource languages (Salehudin et al., 2023; R. Smith, 2007), leaving many SEA languages underserved.

OCR, the process of converting textual images into machine-readable formats, offers significant potential for languages with limited datasets. While many scanned documents and books in these low-resource languages are available online, the text within them often remains inaccessible due to formats like images and PDFs. By extracting the text from these documents, OCR can generate valuable datasets for low-resource languages, which can then be used for downstream NLP tasks, such as machine translation and named-entity recognition (Agarwal & Anastasopoulos, 2024; Ignat et al., 2022). Therefore, studying OCR performance on SEA languages is crucial to accelerating NLP research and technology development in the region.

While OCR has been widely studied for high-resource languages such as English and Chinese, the efficacy and limitations of OCR models on SEA languages remain largely unexplored. To address this gap, this study presents a pipeline to collect textual data from Wikipedia and benchmark several open-source OCR tools on the collected data.

Additionally, we explore the potential of fine-tuning existing models to improve OCR performance on SEA languages. The primary objective is to evaluate and enhance the performance of OCR technologies on SEA languages, thereby advancing NLP applications in this linguistically diverse region.

Specifically, this project seeks to answer the following research questions (RQs):

- **RQ1:** How do popular OCR tools perform on SEA scripts?
- **RQ2:** What script-related challenges affect OCR accuracy on SEA languages?
- **RQ3:** What techniques and recommendations can enhance OCR accuracy on SEA languages?

Chapter 2

Related Work

2.1 Overview of OCR Systems

Most OCR systems consist of two stages: Text detection and text transcription. Text detection identifies text present in an image and extracts cropped regions containing the detected text. A text transcription model then converts these cropped images into text. Generally, separate models are used for each stage, allowing for greater training flexibility and a clearer understanding of challenges within each component (Subramani et al., 2020). More recently, end-to-end models that combine both stages have shown promise in reducing errors for certain use cases (Feng et al., 2019).

2.1.1 Evolution of OCR Models

Early OCR models employ traditional machine learning techniques, such as K-nearest Neighbors (KNN) and Support Vector Machines (SVMs), to classify textual characters from cropped images. Tesseract, an established OCR engine developed since the 1990s, recognizes character patterns by extracting small fragments of character outlines as features (R. W. Smith, 2013). These features are then classified into character clusters using an optimized KNN algorithm. While effective for structured text, these traditional approaches struggled with variations in handwriting, fonts, and image distortions (Subramani et al., 2020).

The rise of deep learning brought significant advancements in OCR. Convolutional Neural Networks (CNNs) improve feature extraction by automatically detecting edges, textures, and shapes within text images. Unlike traditional handcrafted features, CNNs learn visual patterns by applying small filters across an image. The Character Region Awareness for Text Detection (CRAFT) algorithm, for example, uses a fully convolutional network to achieve state-of-the-art character localization (Baek et al., 2019). For text transcription, Recurrent Neural Networks (RNNs) have been widely adopted due to their ability to model sequential dependencies over time. Tesseract v4 integrated a Long Short-Term Memory (LSTM) model, a specialized type of RNN, to recognize entire lines of text instead of individual characters (Tesseract OCR, 2025). By combining CNNs for feature extraction and RNNs for sequence modeling, Shi et al. (2015) proposed the Convolutional Recurrent Neural Network (CRNN), which significantly improved text recognition accuracy in end-to-end OCR systems.

More recently, transformer-based models have emerged as a powerful alternative. Unlike CNNs and RNNs, transformers process entire input sequences in parallel using self-attention mechanisms, which allows them to capture long-range dependencies in text images more efficiently (Vaswani et al., 2017). This approach avoids image-specific inductive biases present in CNNs, such as the assumption that neighboring pixels are relevant. TrOCR, an end-to-end model that combines an image transformer and a separate text transformer, demonstrates another advantage of transformers: the ability to leverage self-supervised pre-training (M. Li et al., 2021). Since transformers can be pre-trained individually to learn useful patterns from unlabeled images and text, there is less reliance on manually annotated OCR training data to achieve high accuracy. Going beyond traditional text recognition, General OCR Theory (GOT) is another transformer-based model that extends character recognition capabilities to non-text elements, such as sheet music, charts, and geometric shapes (Wei et al., 2024). By integrating Large Visual-Language Models (LVLMs), GOT seeks to address the bottlenecks of traditional OCR systems, which often struggle with generalization. As transformer-based OCR continues to evolve, these models are expected to push the boundaries of text recognition, enabling more flexible and adaptable OCR systems for diverse applications.

2.2 Benchmarking OCR on Low-resource Languages

To evaluate OCR performance accurately, textual data in the form of images or PDFs paired with reliable ground truth is essential. Similar to most NLP tasks, data scarcity poses a major obstacle to advancing OCR technology in low-resource languages. The limited availability of annotated textual data restricts both model training and evaluation, leading to disparities in OCR accuracy across different scripts. OCR tools generally perform better on Latin-based scripts (Hegghammer, 2022; Ignat et al., 2022), partly due to market incentives that prioritize the development of English-language OCR systems, resulting in more extensive training data and refinement. Beyond data availability, the complexity of scripts with ornate diacritics or unique letter shapes often yield lower OCR accuracy (Agarwal & Anastasopoulos, 2024).

A recent study by Ignat et al. (2022) provides the most relevant benchmarking of OCR on SEA languages. Their benchmark grouped 60 low-resource languages by region and script, including SEA languages such as Khmer, Lao, Burmese, Thai, and Vietnamese. They found that while OCR models perform well on synthetic SEA-language data, their accuracy drops significantly on real-world data. This discrepancy underscores the need for more diverse and realistic training datasets to improve OCR outcomes for SEA languages.

2.3 Using Synthetic Data for OCR Evaluation

To bridge the gap in data availability, many studies rely on artificial images and PDFs generated from plain text to create evaluation datasets. For instance, Ignat et al. (2022) generated synthetic PDFs from the Flores 101 dataset, which consists of text from Wikipedia in 101 languages. Expanding on this approach, Gupte et al. (2021) developed an open-source Python package that creates document images from plain text, incorporating several document styling templates. These methods enable the large-scale generation of high-quality, low-resource language data with corresponding ground truth annotations.

However, one challenge with artificial datasets is their tendency to lack the imperfections found in real-world documents. Real-world scanned documents often feature complex layouts, stains, and handwritten scribbles (Hegghammer, 2022). Studies have shown that OCR systems often perform better on synthetic datasets than on real-world data, highlighting a gap in generalization (Ignat et al., 2022). To address this, researchers frequently apply noise augmentation to synthetic documents. Common techniques include changing the font style, size, color, and letter spacing, as well as adding Gaussian blur, bleed-through effects, and salt-and-pepper noise (Gupte et al., 2021; Ignat et al., 2022). These modifications help artificial datasets better approximate the challenges of real-world OCR tasks.

2.4 Fine-tuning OCR Systems

To enhance OCR performance in new domains with limited labeled data, many studies explore fine-tuning, or further training pre-trained models on a smaller, task-specific dataset. Instead of training from scratch, fine-tuning updates a model's existing weights, allowing it to adapt to new datasets while retaining prior knowledge. For instance, Parres and Paredes (2023) demonstrated that transformer-based models can successfully adapt to new languages and historical documents with minimal training data, achieving competitive OCR performance. Similarly, Laurent and Lauar (2024) fine-tuned the English TrOCR model for Spanish text, yielding strong results. Fine-tuning thus provides an effective strategy for overcoming the scarcity of labeled data in new OCR domains, particularly for low-resource languages, while maintaining high accuracy.

Chapter 3

Methodology

To answer the research questions, this study conducted the following three experiments to benchmark and improve OCR performance on SEA languages:

- **Experiment 1:** Benchmarking on Real-world Data
- **Experiment 2:** Benchmarking on Synthetic Data
- **Experiment 3:** Fine-tuning for Vietnamese and Thai

3.1 Experiment Setup

3.1.1 Languages

In this study, we chose to benchmark on English, Indonesian, Vietnamese, and Thai. English serves as a baseline comparison due to its extensive OCR research and established tool support. Meanwhile, Indonesian, Vietnamese, and Thai were selected as a representative subset of SEA languages for several reasons.

Firstly, these three languages encompass a range of script types: Latin scripts for Indonesian, Latin scripts with diacritics for Vietnamese, and Brahmic scripts for Thai. By covering these scripts, we capture a broad spectrum of orthographic features, from diacritics to tone marks and from Latin-based scripts to complex character shapes. This allows us to examine how these unique linguistic features impact OCR performance. Furthermore, many other SEA languages, including Malay, Filipino, and Cebuano, use modified Latin scripts, while languages like Khmer, Burmese, and Javanese use Brahmic scripts. Thus, findings from this study can be applied to other languages with similar script types, accelerating OCR research in the region.

Table 3.1: Benchmarked Languages

	Speaker Population	Script Type	Example
English	1.5 billion	Latin	Good morning
Indonesian	252 million	Latin	Selamat pagi
Vietnamese	97 million	Latin with diacritics	Chào buổi sáng
Thai	71 million	Brahmic	สวัสดีตอนเช้า

Note: Speaker population data from Wikipedia (2025).

Secondly, the wide usage of these languages makes it feasible to obtain textual data. The high number of speakers, active online communities, and abundant digital content ensure sufficient resources for OCR benchmarking. Their prominence in SEA further highlights their relevance, as improving OCR for these languages benefits a large portion of the region's population.

While this study covers only a small fraction of the languages spoken in SEA, the selection of these languages provides a strong starting point, as they cover popular script types and offer abundant online data for benchmarking.

3.1.2 Data Source

To collect textual data, this study uses Wikipedia due to its accessibility and multilingual scope. Wikipedia articles can be converted into images via screenshots, simulating real-world OCR scenarios. The platform also offers a convenient Application Programming

Interface (API) that allows retrieval of plain text from most articles, serving as a reliable reference for evaluating OCR accuracy and generating synthetic documents. Moreover, the availability of large corpora in various SEA languages, including Thai, Vietnamese, Indonesian, Tamil, and Burmese, makes Wikipedia suitable for this study's language needs ("List of Wikipedias", 2024).

3.1.3 OCR Systems

In our selection of OCR systems for benchmarking, we prioritize open-source solutions that support a diverse range of SEA languages, promoting accessibility and reusability for the proposed evaluation pipeline. Additionally, we aim to include models with different underlying architectures, enabling a more comprehensive assessment of their performance across different languages. Consequently, we selected EasyOCR, Tesseract, and General OCR Theory (GOT), each open-source and representing distinct modeling approaches to OCR.

Table 3.2: Benchmarked OCR Systems

	Architecture	# Supported Languages
EasyOCR	CRAFT + CRNN	83 (includes all benchmarked languages)
Tesseract	LSTM	116 (includes all benchmarked languages)
GOT	VED	2 (English and Simplified Chinese)

EasyOCR is a modern OCR framework that integrates a text detection model based on the Character Region Awareness for Text Detection (CRAFT) algorithm with a recognition model utilizing a Convolutional Recurrent Neural Network (CRNN) (Jaided AI, 2025). Readily available as a Python package, EasyOCR supports 83 languages, including English, Indonesian, Vietnamese, and Thai.

Tesseract is one of the most well-known open-source OCR engines. Since releasing version 4 in 2018, Tesseract uses an underlying Long Short-Term Memory (LSTM) model for line recognition (Tesseract OCR, 2025). Similar to EasyOCR, Tesseract is accessible via a Python package and supports the four chosen languages in this study

GOT is a transformer-based model designed to recognize artificial characters beyond traditional text, such as sheet music, mathematical equations, and charts (Wei et al., 2024). Using a Vision Encoder Decoder (VED) architecture with 580 million parameters, GOT fine-tunes ViTDeT[1] as its vision encoder and Qwen-0.5B[2] as its language decoder. GOT is conveniently available on Hugging Face[3]. While GOT officially supports only English and Simplified Chinese, it does not support Indonesian, Vietnamese, or Thai. This study seeks to address this limitation by fine-tuning GOT on these languages in Section 3.4.

3.1.4 Evaluation Metrics

$$CER = \frac{I + D + S}{N} \tag{3.1}$$

Similar to most similar studies, we utilize Character Error Rate (CER) and Word Error Rate (WER) as our evaluation metrics to measure OCR accuracy (Hegghammer, 2022; Ignat et al., 2022). CER measures the accuracy of character recognition and is calculated using the Levenshtein or edit distance, which represents the minimum number of single-character insertions (I), deletions (D), and substitutions (S) required to transform one word into another. As shown in Equation 3.1, CER is defined as the edit distance

[1]ViTDeT is an object detection model using the Vision Transformer (ViT) as a backbone network (Y. Li et al., 2022).

[2]Qwen-0.5B is a Large Language Model (LLM) with 500 million parameters developed by Alibaba Cloud (Alibaba Cloud, 2025).

[3]https://huggingface.co/stepfun-ai/GOT-OCR2_0

between the OCR-predicted text and ground truth text, divided by the total number of characters in the ground truth text (N). A lower CER value indicates higher accuracy, with 0 representing perfect recognition. Notably, CER can exceed 1 when there is a significant number of insertions. WER serves as the word-based counterpart to CER.

3.2 Experiment 1: Benchmarking on Real-world Data

To explore the performance of OCR tools on SEA scripts (RQ1), Experiment 1 benchmarks OCR systems using screen-captured, real-world data from Wikipedia. Unlike synthetic data, these screenshots contain formatting variations and complex layouts that better reflect real-world OCR challenges. This approach ensures that the evaluation closely mirrors practical use cases, where OCR tools must handle noisy and visually complex text.

3.2.1 Data Collection

To ensure substantial data availability across our chosen languages, we compiled a dataset of 100 popular Wikipedia articles. Specifically, we selected the 20 most viewed English articles from each of five categories: people, present countries, cities, life, and buildings and structures ("Wikipedia:Popular pages", 2024). These categories were also chosen to create a diverse corpus in terms of content. Table A.1 lists the articles included in our dataset.

From the dataset of 100 Wikipedia articles, we collected article images and ground

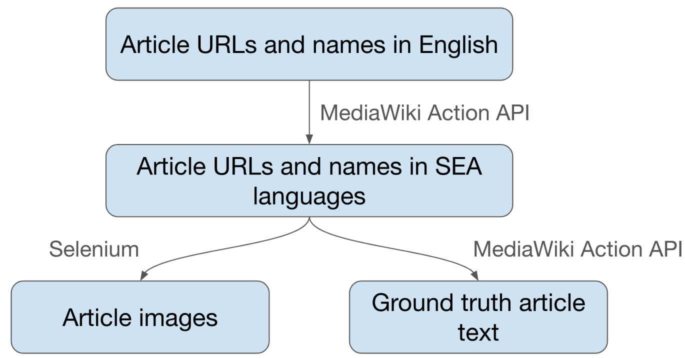

Figure 3.1: Pipeline for data collection from Wikipedia

truth article text in our selected languages using Python, Selenium[4], and the MediaWiki Action API[5]. Figure 3.1 illustrates the overall pipeline for data collection. The detailed steps are as follows:

1. Manually compile the dataset's article names and URLs in English.
2. Fetch the article names and URLs in Thai, Vietnamese, and Indonesian from the MediaWiki Action API.
3. Download the article PDFs in all languages using Selenium.
4. Convert the article PDFs into PNG images, with each image representing one page of the PDF.
5. Download the ground truth article text into TXT files from the MediaWiki Action API.

[4] Selenium is a framework for automating web browsers, commonly used for web scraping by programmatically interacting with websites.

[5] The MediaWiki Action API allows access to wiki page operation features such as search and retrieval.

The end result is a real-world Wikipedia dataset with 3,590 English images, 1,450 Indonesian images, 1,925 Vietnamese images, and 1,011 Thai images.

3.2.2 OCR Evaluation

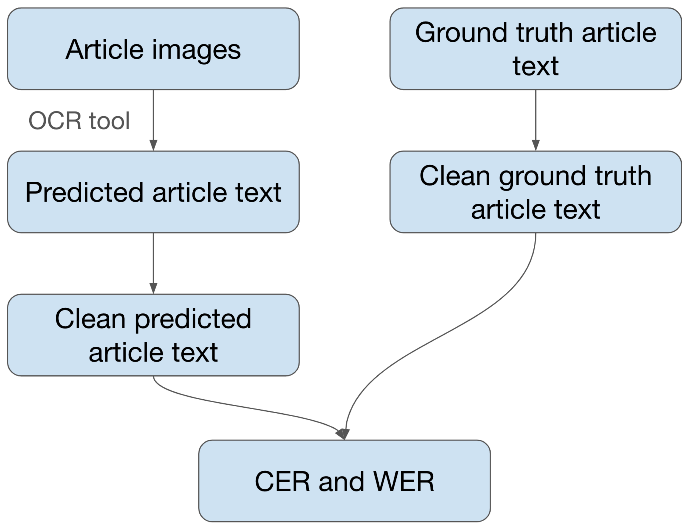

Figure 3.2: Pipeline for OCR evaluation

After collecting the images and corresponding ground truth text, we ran the OCR tools and evaluated the CER and WER for each article. Figure 3.2 summarizes the overall pipeline for OCR evaluation. The detailed steps are as follows:

1. **Apply OCR:** Apply the OCR tools on the article images.

To run EasyOCR, Tesseract, and GOT on all 7,976 images, we used Slurm for job scheduling and execution on the SoC Compute Cluster at the School of Computing, National University of Singapore. Python and shell scripts were utilized to automate OCR execution.

2. **Data Cleaning:** Perform data cleaning on the predicted article text and ground truth article text.

The raw predicted text generated by the OCR tools exhibited some consistent formatting issues. For instance, Tesseract adds an additional space character after every predicted character. The article images also included references and in-text citations, which are not present in the ground truth. To address these issues, we performed data cleaning to align the output text more closely with the ground truth.

3. **Evaluation:** Compute the CER and WER between the predicted text and the ground truth text using JiWER[6].

4. **Data Validation:** Manually review articles with outlier CER values for incorrect ground truth text.

To prevent erroneous results, we implemented a data validation step that automatically checked the CERs for each article. Articles were flagged as outliers if their CER exceeded two standard deviations from the mean (Cousineau & Chartier, 2010). We manually reviewed these outlier articles for anomalies, resulting in the removal of seven articles where the images and ground truth texts contained different content.

3.3 Experiment 2: Benchmarking on Synthetic Data

Unlike Experiment 1, Experiment 2 generates images from plain text for benchmarking. This approach allows us to introduce controlled distortions to the dataset, enabling an

[6] JiWER is a Python package designed for fast calculation of CER and WER.

analysis of OCR robustness against noise on SEA languages and offering a different perspective on RQ1. Additionally, using synthetic data minimizes annotation errors, allowing us to better isolate script-related errors (RQ2).

3.3.1 Synthetic Data Generation

Using the article text collected from Experiment 1, we generated synthetic article images with Python and WeasyPrint[7], a Python package for converting HTML pages into PDF documents. To introduce font noise into the dataset, we randomly applied HTML tags to a specified ratio of space-separated words. For instance, a `<b>` tag could be added randomly to make words appear in bold. The following steps summarizes the process of data generation:

1. If noise is needed, randomly apply HTML tags to a specified ratio of space-separated words.
2. Generate synthetic article PDFs using the ground truth text collected from Experiment 1 with WeasyPrint.
3. Convert the article PDFs into PNG images, with each image representing one page of the PDF.

3.3.2 OCR Evaluation on Different Types of Noise

To investigate how different types of noise impact OCR performance (RQ1), we generated separate datasets, each containing a specific type of noise: bold, italic, link, or heading. The noise was randomly applied to a specified percentage of words in each dataset.

[7] https://pypi.org/project/weasyprint/

Additionally, a control dataset without noise was created for comparison. Table 3.3 summarizes the noise types, their corresponding HTML tags, and the percentage of words affected. Figure 3.3 presents a sample synthetic image with different types of noise.

Table 3.3: Types of noise applied

	HTML Tag	Ratio
Bold	`<b>`	0.3
Italic	`<i>`	0.3
Link	`<a>`	0.3
Heading	`<h1>`	0.03
No noise	-	-

The selected noise types introduce formatting-based distortions commonly found in digital text and web-based content, posing unique challenges for OCR systems. Bold and italic formatting, frequently used for emphasis in scanned documents and articles, may alter character shapes and affect recognition accuracy. Links introduce underlining and color changes, which can interfere with OCR systems. Headings, often bold and larger in size, may also impact recognition. Evaluating these effects thus help assess OCR robustness in processing real-world digital text.

After generating the five separate datasets, we ran the OCR tools and evaluated their performance on each dataset, following the approach described in Section 3.2.2.

3.3.3 Error Classification by Character Type

Another area of interest in Experiment 2 was the effect of unique script characteristics, such as diacritics and punctuation, on OCR accuracy (RQ2). Using OCR results from the control dataset without noise, we categorized misclassifications by 11 character types commonly found in English, Indonesian, Vietnamese, and Thai.

Table 3.4: Character types used for error classification

	Included Characters
Arabic digit	0-9
Thai digit	๐-๙
Latin letter	a-zA-Z
Latin letter with diacritic	à-ỹ
Thai letter	ก-ฮ
Thai diacritic	้๊๋์
Punctuation	.,!?;:()-”’
Thai punctuation	ๆฯ
Vietnamese punctuation	«»
Whitespace	␣
Other	

We used the Levenshtein[8]Python package to identify edit operations (insertions, deletion, and substitutions) and classified misrecognized characters using RegEx[9]. Table 3.4 lists the character types we analyzed, with all uncategorized characters grouped under ”Other”.

3.4 Experiment 3: Fine-tuning for Vietnamese and Thai

[8]https://pypi.org/project/Levenshtein/
[9]https://docs.python.org/3/library/re.html

Figure 3.3: Sample synthetic image with different types of noise

Chapter 4

Results and Analysis

In this chapter, we analyze and evaluate the results of the experiments to provide answers to the research questions.

4.1 RQ1: How do popular OCR tools perform on SEA scripts?

4.1.1 OCR Accuracy on Real-world Data

Table 4.1: Average CER and WER on real-world data

	CER			**WER**		
	EasyOCR	Tesseract	GOT	EasyOCR	Tesseract	GOT
English						
Indonesian						
Vietnamese						
Thai						

Table 4.2: Average CER and WER on synthetic data

	CER			WER		
	EasyOCR	Tesseract	GOT	EasyOCR	Tesseract	GOT
English	0.03	0.09	0.02	0.04	0.01	0.04
Indonesian	0.03	0.09	0.02	0.06	0.01	0.05
Vietnamese	0.10	0.17	0.03	0.04	-	-
Thai	0.07	0.68	0.09	0.64	-	-

Table 4.3: Average OCR runtime per page (seconds)

	EasyOCR	Tesseract	GOT
English	3.23	11.68	24.35
Indonesian	2.92	13.19	31.44
Vietnamese	3.91	11.80	-
Thai	2.32	16.76	-

4.1.2 OCR Accuracy on Synthetic Data

4.1.3 Runtime

4.2 RQ2: What script-related challenges affect OCR accuracy on SEA languages?

4.3 RQ3: What techniques and recommendations can enhance OCR accuracy on SEA languages?

English

	Count	EasyOCR % Missed	Tesseract % Missed	GOT % Missed
Arabic digit	38324	0.72%	1.93%	0.27%
Latin letter	1546964	1.32%	1.84%	0.38%
Latin letter with diacritic	424	100.00%	53.07%	14.62%
Punctuation	53403	28.41%	2.32%	3.48%
Whitespace	317587	4.87%	4.31%	3.60%
Other	3298	82.84%	68.53%	76.93%

Figure 4.1: Error classification by character type for English articles

Indonesian

	Count	EasyOCR % Missed	Tesseract % Missed	GOT % Missed
Arabic digit	24947	0.37%	1.82%	0.23%
Latin letter	1208707	0.46%	1.76%	0.36%
Latin letter with diacritic	262	5.34%	100.00%	15.27%
Punctuation	37788	22.05%	3.14%	0.76%
Whitespace	207556	4.82%	5.07%	4.09%
Other	2468	72.24%	80.51%	43.19%

Figure 4.2: Error classification by character type for Indonesian articles

Vietnamese	Count	EasyOCR % Missed	Tesseract % Missed
Arabic digit	31473	1.14%	2.17%
Latin letter	916667	8.51%	1.84%
Latin letter with diacritic	292686	14.79%	1.84%
Punctuation	40420	24.64%	2.17%
Whitespace	367936	10.90%	5.34%
Other	35767	12.46%	7.68%

Figure 4.3: Error classification by character type for Vietnamese articles

Thai	Count	EasyOCR % Missed	Tesseract % Missed
Arabic digit	22580	0.89%	6.69%
Latin letter	36174	100.00%	100.00%
Latin letter with diacritic	96	100.00%	100.00%
Thai letter	617699	0.39%	3.05%
Thai diacritic	90620	3.69%	3.55%
Punctuation	13669	6.43%	8.40%
Thai punctuation	901	78.80%	3.88%
Whitespace	58164	37.51%	37.21%
Other	306647	2.15%	7.07%

Figure 4.4: Error classification by character type for Thai articles

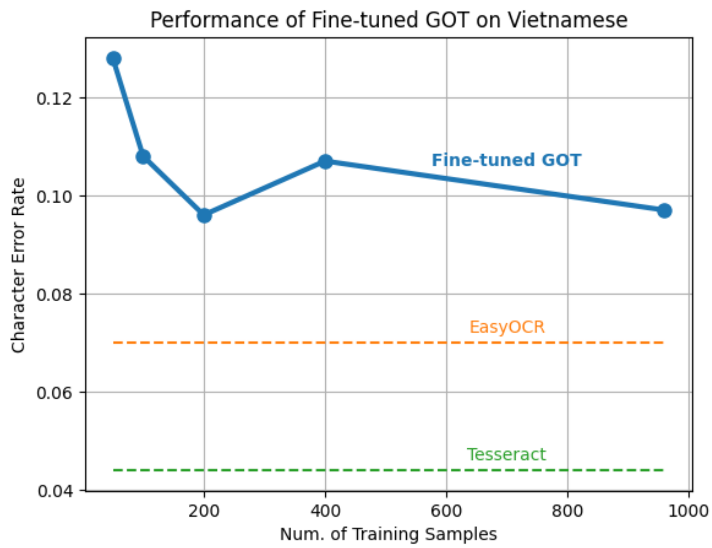

Figure 4.5: Performance of fine-tuned GOT on Vietnamese

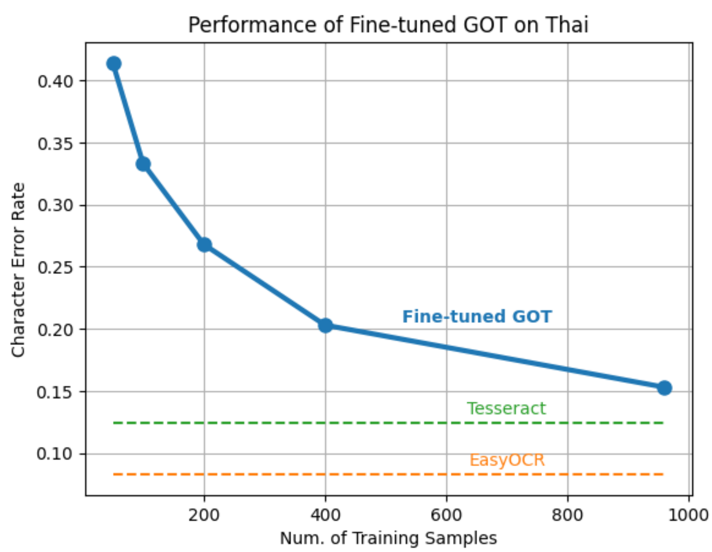

Figure 4.6: Performance of fine-tuned GOT on Thai

Chapter 5

Discussion

Chapter 6

Conclusion

References

Agarwal, M., & Anastasopoulos, A. (2024). A concise survey of OCR for low-resource languages. In M. Mager, A. Ebrahimi, S. Rijhwani, A. Oncevay, L. Chiruzzo, R. Pugh, & K. von der Wense (Eds.), *Proceedings of the 4th workshop on natural language processing for indigenous languages of the americas (americasnlp 2024)* (pp. 88–102). Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.americasnlp-1.10

Aji, A. F., Forde, J. Z., Loo, A. M., Sutawika, L., Wang, S., Winata, G. I., Yong, Z.-X., Zhang, R., Doğruöz, A. S., Tan, Y. L., & Cruz, J. C. B. (2023). Current status of NLP in South East Asia with insights from multilingualism and language diversity. *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, 8–13. https://aclanthology.org/2023.ijcnlp-tutorials.2

Alibaba Cloud. (2025). Qwen. https://github.com/QwenLM/Qwen

Baek, Y., Lee, B., Han, D., Yun, S., & Lee, H. (2019). Character region awareness for text detection. *CoRR*, *abs/1904.01941*. http://arxiv.org/abs/1904.01941

Cousineau, D., & Chartier, S. (2010). Outliers detection and treatment: A review. *International Journal of Psychological Research*, *3*, 58–67. https://doi.org/10.21500/20112084.844

Feng, W., He, W., Yin, F., Zhang, X.-Y., & Liu, C.-L. (2019). Textdragon: An end-to-end framework for arbitrary shaped text spotting. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Gupte, A., Romanov, A., Mantravadi, S., Banda, D., Liu, J., Khan, R., Meenal, L. R., Han, B., & Srinivasan, S. (2021). Lights, camera, action! A framework to improve NLP accuracy over OCR documents. *CoRR*, *abs/2108.02899*. https://arxiv.org/abs/2108.02899

Hegghammer, T. (2022). OCR with Tesseract, Amazon Textract, and Google Document AI: A benchmarking experiment. *Journal of Computational Social Science*, *5*, 861–882. https://doi.org/https://doi.org/10.1007/s42001-021-00149-1

Ignat, O., Maillard, J., Chaudhary, V., & Guzmán, F. (2022). OCR improves machine translation for low-resource languages. In S. Muresan, P. Nakov, & A. Villavicencio (Eds.), *Findings of the association for computational linguistics: Acl 2022* (pp. 1164–1174). Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.findings-acl.92

Jaided AI. (2025). Easyocr. https://github.com/JaidedAI/EasyOCR

Laurent, V., & Lauar, F. (2024). Spanish trocr: Leveraging transfer learning for language adaptation. https://arxiv.org/abs/2407.06950

Li, M., Lv, T., Cui, L., Lu, Y., Florêncio, D. A. F., Zhang, C., Li, Z., & Wei, F. (2021). Trocr: Transformer-based optical character recognition with pre-trained models. *CoRR*, *abs/2109.10282*. https://arxiv.org/abs/2109.10282

Li, Y., Mao, H., Girshick, R., & He, K. (2022). Exploring plain vision transformer backbones for object detection. https://arxiv.org/abs/2203.16527

List of languages by total number of speakers. (2025). https://en.wikipedia.org/wiki/List_of_languages_by_total_number_of_speakers

List of wikipedias. (2024). https://en.wikipedia.org/wiki/List_of_Wikipedias

Magueresse, A., Carles, V., & Heetderks, E. (2020). Low-resource languages: A review of past work and future challenges. *CoRR*, *abs/2006.07264*. https://arxiv.org/abs/2006.07264

Parres, D., & Paredes, R. (2023). Fine-tuning vision encoder–decoder transformers for handwriting text recognition on historical documents. In G. A. Fink, R. Jain, K. Kise, & R. Zanibbi (Eds.), *Document analysis and recognition - icdar 2023* (pp. 253–268). Springer Nature Switzerland.

Salehudin, M., Basah, S., Yazid, H., Basaruddin, K., Safar, M., Som, M. M., & Sidek, K. (2023). Analysis of optical character recognition using easyocr under image degradation. *Journal of Physics: Conference Series*, *2641*(1), 012001. https://doi.org/10.1088/1742-6596/2641/1/012001

Shi, B., Bai, X., & Yao, C. (2015). An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *CoRR*, *abs/1507.05717*. http://arxiv.org/abs/1507.05717

Smith, R. (2007). An overview of the tesseract ocr engine. *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, *2*, 629–633. https://doi.org/10.1109/ICDAR.2007.4376991

Smith, R. W. (2013). History of the Tesseract OCR engine: what worked and what didn't. In R. Zanibbi & B. Coüasnon (Eds.), *Document recognition and retrieval xx* (p. 865802). SPIE. https://doi.org/10.1117/12.2010051

Subramani, N., Matton, A., Greaves, M., & Lam, A. (2020). A survey of deep learning approaches for OCR and document understanding. *CoRR*, *abs/2011.13534*. https://arxiv.org/abs/2011.13534

Tesseract OCR. (2025). Tesseract. https://github.com/tesseract-ocr/tesseract

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *CoRR*, *abs/1706.03762*. http://arxiv.org/abs/1706.03762

Wei, H., Liu, C., Chen, J., Wang, J., Kong, L., Xu, Y., Ge, Z., Zhao, L., Sun, J., Peng, Y., Han, C., & Zhang, X. (2024). General ocr theory: Towards ocr-2.0 via a unified end-to-end model. https://arxiv.org/abs/2409.01704

Wikipedia:popular pages. (2024). https://en.wikipedia.org/wiki/Wikipedia:Popular_pages

Appendix A

Wikipedia Dataset

Table A.1: Dataset of 100 Wikipedia articles used for benchmarking real-world data in Experiment 1 (See Section 3.2)

Category	**Articles**
People	Elizabeth II, Barack Obama, Michael Jackson, Elon Musk, Lady Gaga, Adolf Hitler, Eminem, Lionel Messi, Justin Bieber, Freddie Mercury, Kim Kardashian, Johnny Depp, Steve Jobs, Dwayne Johnson, Michael Jordan, Taylor Swift, Stephen Hawking, Kanye West, Donald Trump, Cristiano Ronaldo
Present countries	United States, India, United Kingdom, Canada, Australia, China, Russia, Japan, Germany, France, Singapore, Israel, Pakistan, Philippines, Brazil, Italy, Netherlands, New Zealand, Ukraine, Spain
Cities[a]	New York City, London, Hong Kong, Los Angeles, Dubai, Washington, D.C., Paris, Chicago, Mumbai, San Francisco, Rome, Monaco, Toronto, Tokyo, Philadelphia, Machu Picchu, Jerusalem, Amsterdam, Boston, Angelsberg
Life	Cat, Dog, Animal, Lion, Coronavirus, Tiger, Human, Dinosaur, Elephant, Virus, Horse, Photosynthesis, Evolution, Apple, Bird, Mammal, Potato, Polar bear, Shark, Snake
Buildings and structures[b]	Taj Mahal, Burj Khalifa, Statue of Liberty, Great Wall of China, Eiffel Tower, Berlin Wall, Stonehenge, Mount Rushmore, Colosseum, Auschwitz concentration camp, Great Pyramid of Giza, One World Trade Center, Empire State Building, White House, Petra, Large Hadron Collider, Hagia Sophia, Golden Gate Bridge, Panama Canal, Angkor Wat

[a] Singapore was replaced because it's already listed under present countries.

[b] Machu Picchu was replaced becuase it's already listed under cities.