សាកលវិទ្យាល័យភូមិន្ទភ្នំពេញ

ROYAL UNIVERSITY OF PHNOM PENH

ការបំប្លែងរូបភាពឯកសារទៅជាអក្សរកែប្រែបានទៅលើភាសាខ្មែរនិងអង់គ្លេស ដោយប្រើប្រាស់ ម៉ូដែល Craft ជាមួយនឹង TrOCR

An End-to-End approach for Khmer & English Text Recognition using Craft with TrOCR Architecture

Mr. Vitou Soy

A Thesis

In Partial Fulfilment of the Requirement for the Degree of
Bachelor of Engineering in Information Technology Engineering

June 2025

សាកលវិទ្យាល័យភូមិន្ទភ្នំពេញ

ROYAL UNIVERSITY OF PHNOM PENH

ការបំប្លែងរូបភាពឯកសារទៅជាអក្សរកែប្រែបានទៅលើភាសាខ្មែរ និង អង់គ្លេស ដោយប្រើប្រាស់ ម៉ូដែល Craft ជាមួយនឹង TrOCR

An End-to-End approach for Khmer & English Text Recognition using Craft with TrOCR Architecture

A Thesis

In Partial Fulfilment of the Requirement for the Degree of
Bachelor of Engineering in Information Technology Engineering

Vitou Soy

Examination committee: Mr. Sokchea Kor
Mr. Chanpiseth Chap
Mrs. Daly Chea
Dr. . . . . .

June 2025

SUPERVISOR's RESEARCH SUPERVISION STATEMENT

Name of program: Bachelor of Engineering in Information Technology Engineering
Name of candidate: Vitou Soy

Title of thesis: An End-to-End approach for Khmer & English Text Recognition using Craft with TrOCR Architecture

This is to certify that the research carried out for the above titled master's research report was completed by the above named candidate under my direct supervision. This thesis material has not been used for any other degree. The candidate has demonstrated strong research capabilities and independence in developing novel approaches for Khmer text recognition. The research methodology, implementation, and results are original contributions to the field of Khmer OCR technology. I have provided guidance and oversight throughout the research process while allowing the candidate to explore innovative solutions.

Supervisor's name: Sokchea Kor

Supervisor's signature:……

Date……

សេចក្តីសង្ខេប

នៅក្នុងអត្ថបទស្រាវជ្រាវមួយនេះ យើងនឹងបង្ហាញអំពី ការបំប្លែងរូបភាពឯកសារទៅជាអក្សរ ដែលអាចកែប្រែបាន នៅលើ ភាសាខ្មែរ និងភាសាអង្គ្លេស។ ចំពោះការបំប្លែងរូបភាពឯកសារ នៃភាសាខ្មែរ ទៅជាអក្សរដែលអាចកែប្រែបាន នៅតែមានបញ្ហាសម្រាប់ OCR System ។ មិនត្រឹមតែភាពខ្វះខាតនៃ ទិន្នន័យ (dataset) សម្រាប់ ឲ្យម៉ូដែលរៀនប៉ុណ្ណោះ ទេ ប៉ុន្តែ ភាពស្មុកស្មាញនៃភាសាខ្មែរក៏តែមានភាពពិបាកក្នុងការចាប់យកអក្សរចេញពីរូបភាពផងដែរ ដូចជា ការសរសេរតម្រួតជាន់លើគ្នា ការដកឃ្លាមានភាពមិនទៀតទាត់ ដែលធ្វើឲ្យការចាប់ យកអក្សរចេញពីរូបភាព មានការលំបាក ខ្លាំង។ ការស្រាវជ្រាវភាគច្រើន គឺគេធ្វើតែទៅលើភាសា អក្សរឡាតាំងតែប៉ុណ្ណោះ ជាប្រភេទ (latin-based scripts) ដែលជាអក្សរ System មួយប្រភេទ. ដែលមិនសូវមានការសរសេរតម្រួតគ្នាច្រើនតង់។ ការកើនឡើងនៃការប្រើប្រាស់ភាសាចូលគ្នានៅ ក្នុងប្រទេសកម្ពុជា មានការកើនឡើងយ៉ាងខ្លាំង នៅពីរទសវត្សរ៍ចុងក្រោយមកនេះ ទៅលើឯក សារដូចជា ស្លាកសញ្ញា អត្ថបទការសែត មេរៀន និង ការសន្ទនាផ្សេងៗទៀតដែលបង្កឲ្យមាន ភាពស្មុកស្មាញ ខ្លាំងសម្រាប់ការចាប់យកអក្សរ ចេញពីរូបភាពដែលមានអក្សរលាយលំគ្នាច្រើន ភាសាបែបនេះ។

នៅក្នុងការសិក្សាស្រាវជ្រាវមួយនេះ ពួកយើងបានបង្កើត វិធីសាស្ត្រមួយដែលហៅថា End-to-end OCR approach ដែលធ្វើឡើងសម្រាប់ដោះស្រាយបញ្ហាខាងលើ។ វិធីសាស្ត្រមួយនេះបានធ្វើការបែង ចែកជាពីរផ្នែកធំៗ ដំណាក់កាលទីមួយនោះគឺ ការចាប់យកទីតាំងអក្សរនៅលើរូបភាពដើម្បី បង្កឲ្យ មានភាពងាយស្រួលសម្រាប់ ការចាប់យកអក្សរចេញពី រូបភាពឲ្យបានប្រសិទ្ធភាពខ្ពស់ជាងមុន (text-detection)។ នៅក្នុងដំណាក់កាលទី២វិញ គឺការបម្លែងរូបភាពទាំងអស់នោះ ទៅជា អត្ថបទដែលអាចធ្វើការកែប្រែបាន (text-detection)។ ដំណាក់កាលទាំងពីរនេះ គឺបង្កើត ឡើងសម្រាប់ធ្វើការដោះស្រាយភាពស្មុកស្មាញទៅលើភាសាខ្មែរ និងការលាយលំភាសាខ្មែរជាមួយ នឹងភាសាអង់គ្លេស។ ដើម្បីបង្រៀនម៉ូដែល និងធ្វើការវាយតម្លៃបាន ដំបូងយើង បានរៀបចំ (dataset) ដែលមាន high-quality ទាំង real dataset និង synthetic dataset, យើងបានប្រមូល data ទាំងអស់នោះដោយខ្លួនឯង ចំនួន 1000 រូបភាព ហើយបានធ្វើការកំណត់ទីតាំងអក្សរ នៅលើរូបភាព ទទួលបាន 13200 ទីតាំងផ្សេងគ្នា។ ដើម្បីបង្កើន data ឲ្យកាន់តែច្រើន យើងបានប្រើប្រាស់វិធីសាស្ត្រ (text-to-image) ដើម្បី បង្កើនរូបភាពឲ្យកាន់តែច្រើនជាងមុន ដោយសារតែបើពឹងផ្អែកតែទៅលើ ការប្រមូលម្តងមួយៗ គឺចំណាយពេលវេលានឹងថវិការច្រើន ដូច្នេះមានតែប្រើប្រាស់វិធីសាស្ត្រ generate នេះឯង។ បន្ទាប់មកយើងទទួលបានពីការ generate ប្រហែល 8 លានរូបភាព។

សម្រាប់ ការកំណត់ទីតាំងនៃអក្សរនៅលើរូបភាព (text detection stage), ពួកយើងបាន ប្រើប្រាស់ Craft ម៉ូដែល, ដែលវាល្អសម្រាប់ ការចាប់យកទីតាំងអក្សរ នៅលើរូបភាព។ ពួក យើងបាន បង្រៀនម៉ូដែល ទៅលើ dataset manually collected ប្រហែល 1000 រូបភាព ដែលអាចកំណត់ទីតាំងអក្សរនៅលើរូបភាពបានល្អ។ យើងទទួលបាន recall 90%, precision 89%, និងចុងក្រោយគឺ F1-score 86.8% ។

នៅក្នុង ការចាប់យកអក្សរចេញពីរូបភាពយើងបានប្រើប្រាស់ TrOCR ម៉ូដែលដែលបាន បង្កើត ឡើងដោយក្រុមស្រាវជ្រាវរបស់ microsoft ដែលវាបានប្រើប្រាស់ transformer-based architecture ម៉ូដែលមួយនេះមិនមាន feature ដើម្បី បម្លែងរូបភាពនៅលើភាសាខ្មែរនោះទេ យើងមានតែធ្វើការ កែប្រែ processor របស់ម៉ូដែលមួយនេះ ដើម្បីឲ្យម៉ូដែលមួយនេះ មាន សមត្ថភាព បំប្លែងរូបភាពភាសាខ្មែរបាន។ យើងបានបង្រៀនវាទៅលើ dataset ទាំងពីរប្រភព real dataset និង synthetic dataset ដែលទទួលបាន CER 0.02 សម្រាប់ទូទៅ, CER 0.04 សម្រាប់ ភាសាខ្មែរ, CER 0.01 សម្រាប់ភាសាអង្គ្លេស និងចុងក្រោយទទួលបាន CER 0.06 សម្រាប់ ការលាយលំអក្សរខ្មែរ ជាមួយនឹងភាសាអង្គ្លេស។

ដើម្បីឲ្យការស្រាវជ្រាវមួយនេះ មានការប្រើប្រាស់ជាក់ស្តែងបាន យើងបាន សរសេរពីរបៀប hosting ការស្រាវជ្រាវមួយនេះ សម្រាប់អនុញ្ញាតឲ្យ មានការ យកទៅប្រើប្រាស់បាន តាមរយៈបង្កើតជា API service សម្រាប់ការតភ្ជាប់ជាមួយនឹងកម្មវិធីផ្សេងៗទៀតបាន។ វាបានបំពេញចន្លោះប្រហោងរវាងការស្រាវជ្រាវនិងការអនុវត្តន៍ក្នុងភាពពិតជាក់ស្តែងក្នុងសហគមន៍។ ដែលវាមានសមត្ថភាពស្គាល់អក្សរពហុភាសាខ្មែរ-អង់គ្លេស បានយ៉ាងល្អ។

Abstract

Khmer text recognition presents persistent challenges for OCR systems—not only due to the limited availability of annotated training data, but also because of the language's complex script structure, including stacked characters, overlapping diacritics, and inconsistent font rendering. These features make Khmer particularly difficult for OCR models, Most of OCR systems which are often designed around Latin-based scripts. The increasing use of English alongside Khmer Cambodia's signage, documents, and digital content further adds to the complexity of building an effective multilingual OCR system.In this work, we introduce an end-to-end OCR approach tailored for both Khmer and English, designed specifically for low-resource and multilingual interpretation. Our OCR system follows a two-stage pipeline, consisting of (1) a text detection model and (2) a text recognition model, both optimized to handle the structural and linguistic challenges unique to Khmer script. To train and evaluate the system, we first constructed a high-quality dataset combining both real and synthetic data. This included manually collecting 1,000 real-world images and annotating 13,200 text-line bounding boxes. To increase data diversity and simulate real-world variability, we also generated thousands of synthetic images using a text-to-image method, based on a large corpora of Khmer and English sentences rendered with varied fonts, layouts, real world background environments and visual noise. For the text detection stage, we adopted the CRAFT model, which is well-suited for detecting irregular and curved text. Trained on our annotated dataset, CRAFT achieved strong results: 90% recall, 89% precision, and an F1-score of 86.8%—demonstrating that accurate detection is achievable even in low-resource settings. In the recognition stage, we used TrOCR, a transformer-based OCR architecture. Since TrOCR's original processor lacked support for Khmer language, we customize it to properly tokenize and decode Khmer characters by build on top of original processor. After training on the combined real and synthetic datasets, the model achieved a character error rate (CER) of 0.02 overall —0.04 for Khmer, 0.01 for English, and 0.06 for mixed-language text-line. To ensure practical usability, we deployed the system as a production-ready public API, allowing direct integration into real-world applications. Unlike many academic models that remain unused after publication, our deployment helps close the gap between research and application—supporting broader adoption of Khmer-English OCR in multilingual, low-resource environments.

CANDIDATE'S STATEMENT

TO WHOM IT MAY CONCERN

This is to certify that the dissertation that I, Vitou Soy, hereby present, entitled "Advancing Khmer Optical Character Recognition: A Synthetic Data-Driven Approach," for the degree of Bachelor of Engineering in Information Technology at the Royal University of Phnom Penh, is entirely my own work. Furthermore, it has not been used to fulfill the requirements of any other qualification, in the whole or in part, at this or any other University or equivalent institution. The research methodology, implementation, and findings represent original con- tributions to the field of Khmer OCR technology, particularly in developing novel approaches for synthetic data generation and transformer-based text recognition. Through this work, I have demonstrated strong research capabilities and independence in addressing the critical challenges of Khmer text digitization and recognition.

No reference to, or quotation from, this document may be made without the written approval of the author.

Name of Candidate: Vitou Soy

Signed by the candidate: .....

Date: .....

Name of Supervisor: Mr. Sokchea Kor

Countersigned by the Supervisor: .....

Date: .....

ACKNOWLEDGMENTS

I would like to begin by expressing my sincere gratitude for the opportunity to pursue this research. I am deeply thankful to the Royal University of Phnom Penh for offering such a well-structured academic program within the Faculty of Engineering, which has provided a strong foundation for this success. The comprehensive curriculum, practical exposure, and academic environment have been instrumental in shaping my journey.

I am especially grateful to all the faculty members for their dedicated teaching and continuous support throughout my studies. Their commitment to student learning has inspired and guided me significantly. I would also like to thank my classmates for their collaboration, encouragement, and the shared experiences that made this journey memorable.

My deepest appreciation goes to my supervisor, Mr. Kor Sokchea, whose expertise, mentorship, and unwavering support have been vital to the completion of this work. He has been not only an exceptional lecturer and supervisor but also a strong supporter at every stage. This research would not have been possible without his guidance.

Finally, I want to express my heartfelt thanks to my family for their financial support, constant encouragement, and unconditional love. Their sacrifices and belief in me have been the driving force behind my achievements. I am especially grateful to my sister, who has always believed in me and supported every decision I made—thank you for always standing by my side.

TABLE OF CONTENTS

Preliminary Pages

List of Tables

List of Figures

LIST OF ABBREVIATIONS

AI: Artificial Intelligence
API: Application Programming Interface
BART: Bidirectional and Auto-Regressive Transformers
CER: Character Error Rate
CNN: Convolutional Neural Network
CRAFT: Character Region Awareness for Text Detection
CTC: Connectionist Temporal Classification
DCT: Discrete Cosine Transform
EAST: Efficient and Accurate Scene Text Detector
GPU: Graphics Processing Unit
Grad-CAM: Gradient-weighted Class Activation Mapping
HMM: Hidden Markov Model
HTK: Hidden Markov Model Toolkit
ICDAR: International Conference on Document Analysis and Recognition
IoU: Intersection over Union
LSTM: Long Short-Term Memory
MLP: Multilayer Perceptron
NAR: Non-Autoregressive
NLP: Natural Language Processing
OCR: Optical Character Recognition
RAM: Random Access Memory
RNN: Recurrent Neural Network
Seq2Seq: Sequence-to-Sequence
SOM: Self-Organizing Map
SVM: Support Vector Machine
TrOCR: Transformer Optical Character Recognition
URL: Uniform Resource Locator
VGG: Visual Geometry Group
ViT: Vision Transformer
VRAM: Video Random Access Memory
WER: Word Error Rate
YOLO: You Only Look Once

Chapter 1
Introduction

1.1 Background to the Study

OCR technology has revolutionized how we convert printed documents into digital text. With recent AI breakthroughs, OCR systems now use deep learning to automatically find and recognize characters in images. This technology is everywhere from digital libraries to search engines to language translation tools. For popular languages like English, Chinese, and Japanese, OCR works incredibly well. These languages have massive amounts of training data and researchers have studied their text patterns for decades. However, when it comes to complex scripts like Khmer, OCR technology is still way behind. Cambodia is in desperate need of better OCR right now. The country has been rapidly digitizing over the last two decades. Everyone wants to convert Khmer documents such as textbooks, historical records, cultural materials into digital formats for education and research. There's a huge problem with Khmer script which is extremely complicated. English letters just line up in a row from left to right while Khmer characters pile on top of each other in complex ways. They have tiny accent marks, subscripts, and vowel symbols that can appear above, below, or wrapped around the main letter. If there is missing even a small mark, the word meaning is completely changed.

Table 1.1: Why Khmer OCR is Desperately Needed

Sector	Cause	Effect
Education	Physical Textbooks	Students can't search or edit content
Libraries	Books Rotting on Shelves	Knowledge becomes inaccessible
Government	Tranditional Paper Work	Slow bureaucracy, hard to find documents
Administrative	Handwritten Document	Manual Data Entry
Culture	Ancient Texts Deteriorating	Losing heritage records

1.2 Problem Statement

To develop high accuracy OCR model needs tons of millions annotated images. Khmer language resources are extremely limited, with only a few thousand quality annotated samples available. Moreover, as mentioned from previous section Khmer scripts is very complicated. Characters stack on top of each other with just tiny changes in the script marks could differentiate the word meaning. There are multiple problems with Khmer writing system that make it difficult to implement OCR model as listed below:

1. **Complex Character Structure:** Khmer OCR faces unique challenges with character stacking and diacritics. Characters pile on top of each other in complex ways with tiny

accent marks, subscripts, and vowel symbols that can appear above, below, or wrapped around the main letter. Missing even a small mark completely changes the word meaning.

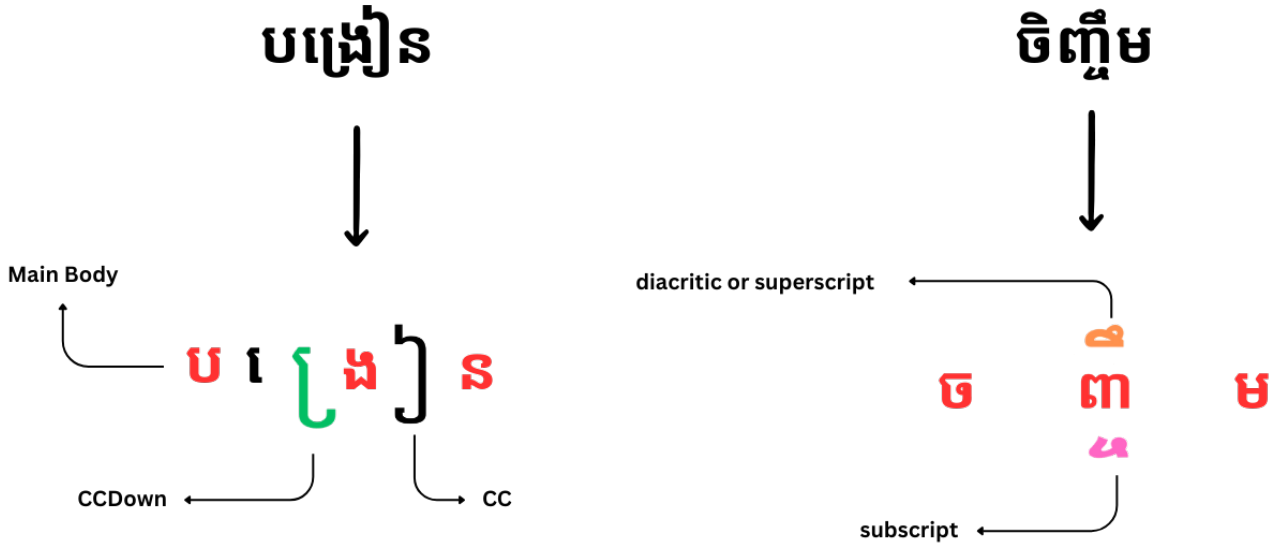

Figure 1.1: Example of Khmer text format showing the complexity of character combinations and diacritics

2. **Inconsistent Word Boundaries:** Khmer OCR faces unique challenges with inconsistency in word boundaries. English have spaces to separate words unlike Khmer writing with no standard spacing format. This kind of case creates a major problem for AI to tell where one word ends and another begins.

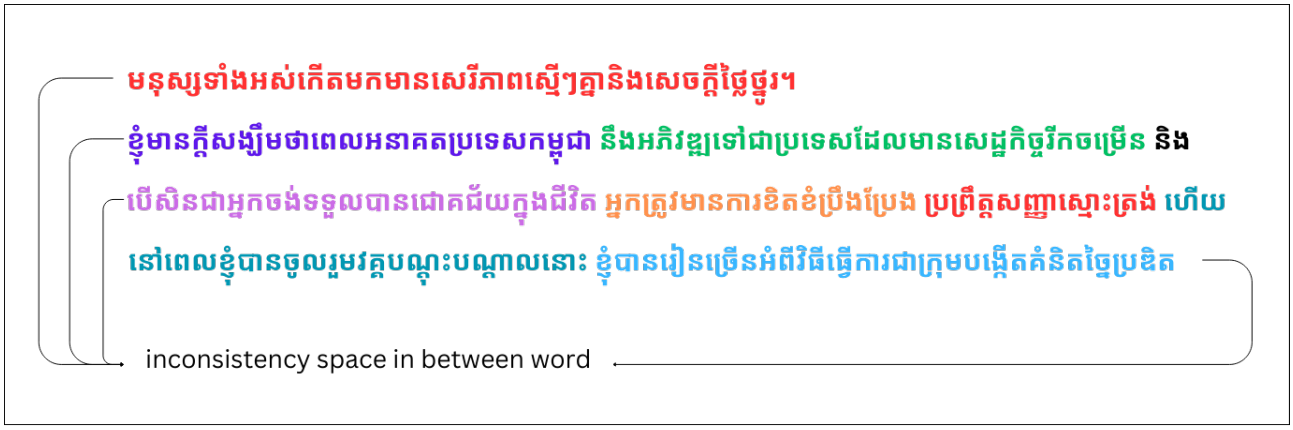

Figure 1.2: Example of sequential Khmer text showing how characters combine to form syllables and words

3. **Mixed-Language Complexity:** Modern Cambodia faces a growing challenge with mixed-language documents, as the rise of English education and international business over the past two decades that made it common to see signs, textbooks, and digital content mixing both languages within the same sentence or paragraph. This mixed-language phenomenon creates several specific problems for OCR systems including script switching confusion, different text layouts, and font inconsistencies.

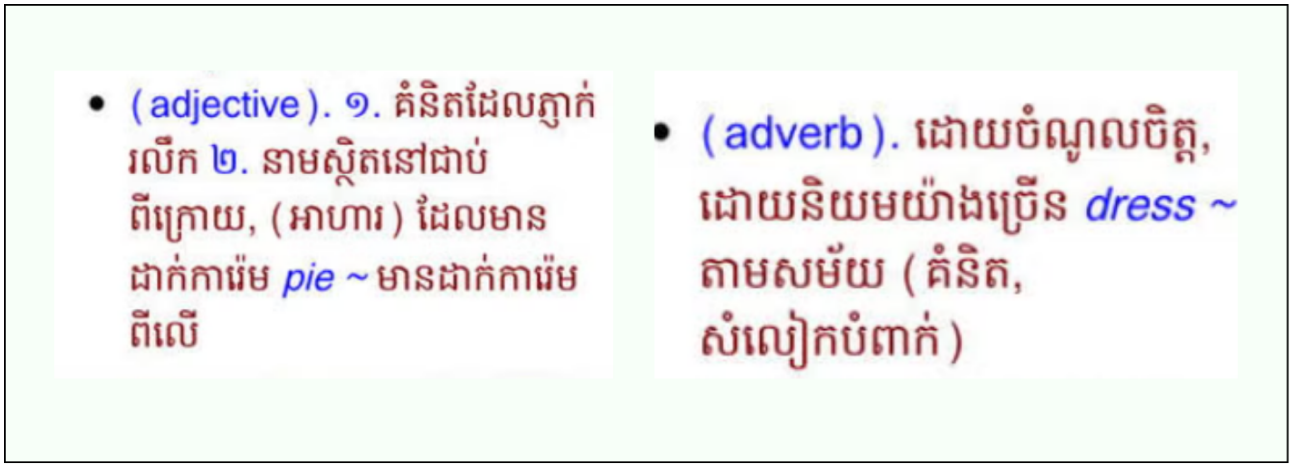

Figure 1.3: Example of mixed Khmer-English text showing how both languages appear together in modern Cambodian documents

Most existing Khmer OCR systems are designed for single-language scenarios and perform poorly when encountering this mixed-language reality that exist everywhere in Cambodia today. This situation becomes even more challenging when seeking properly labeled mixed-language datasets that combine Khmer and English text. Without enough training data, OCR model stays dumb. It can't learn the patterns to recognize text accurately.

4. **Font Variation Challenges:** English has maybe around 15 to 25 common fonts that most people use, based on reported study use case, from websites such as rigorous-themes.com, lifehack.org, and indeed.com. Khmer has many different fonts that look very unique from each other, some are thick and bold, others are thin and delicate, some have fancy decorations. To train model on one font and it fails completely on another.

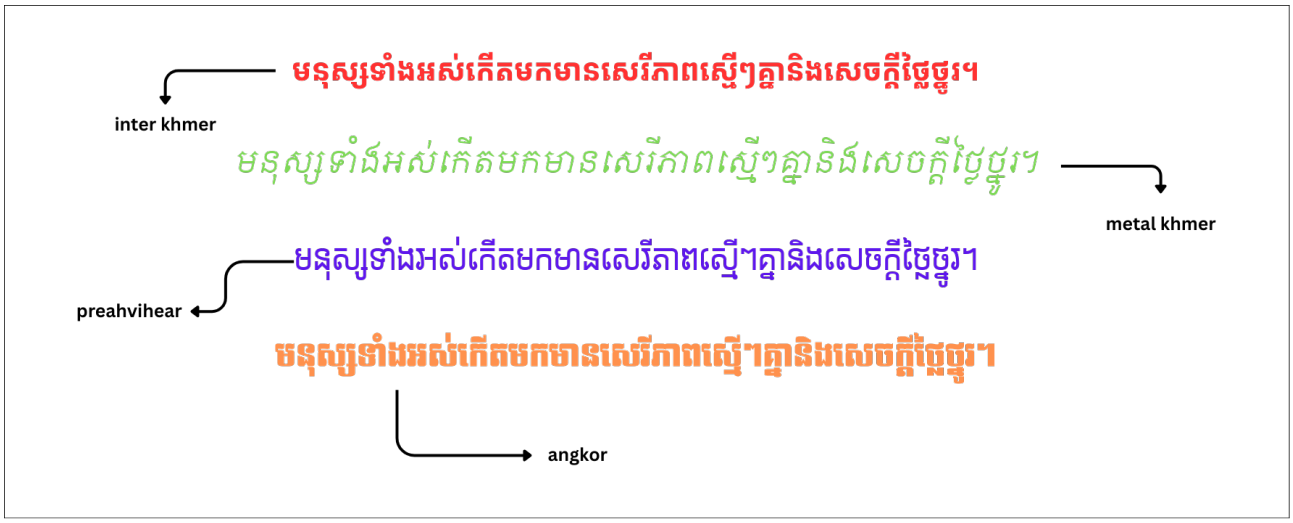

Figure 1.4: Examples of the same Khmer text rendered in different fonts, demonstrating the significant visual variations that OCR systems must handle

5. **Character Stacking Patterns:** Khmer text has complex stacking patterns where characters combine vertically and horizontally to form syllables and words. This creates spatial relationships that are difficult for traditional OCR systems to handle.

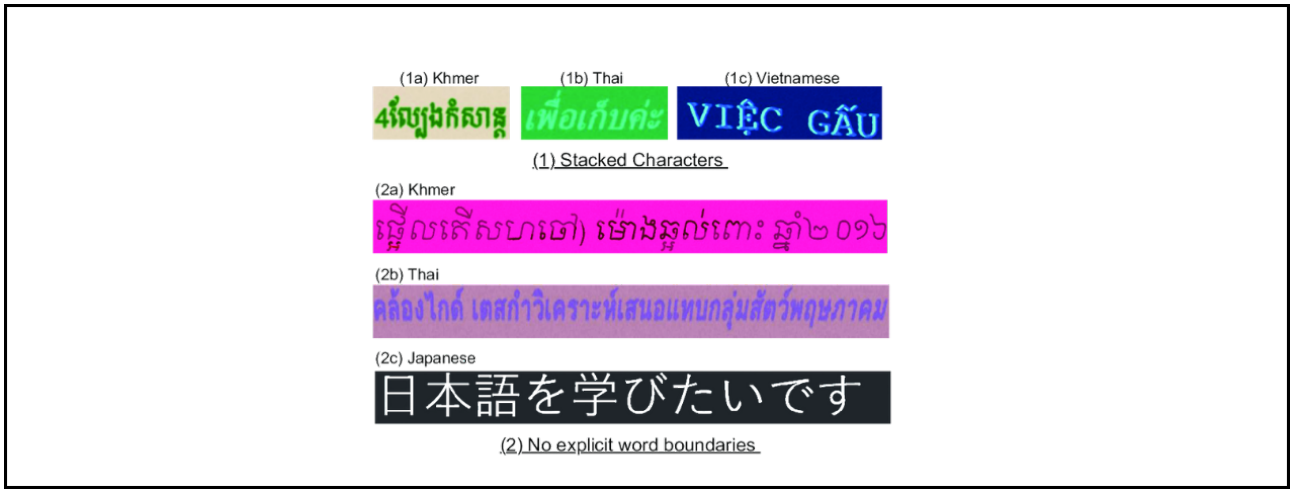

Figure 1.5: Illustration of Khmer text stacking patterns, showing how characters combine vertically and horizontally to form syllables and words (Buoy et al., 2023)

1.3 Aim and Objectives of the Study

Our main goal is to create high performance OCR system that can handle mixed writing language of Khmer and English in the same document. Our objectives mainly are:

1. **Create a text annotation tool:** Build a tool that can efficiently mark up images with bounding boxes and text labels for both text detection and recognition tasks.

2. **Generate synthetic data that actually helps:** To collect real data is expensive and time-consuming, we create millions of synthetic images with different fonts, backgrounds, and realistic distortions that look authentic enough to train the models effectively.

3. **Build an end-to-end OCR pipeline:** Customize CRAFT for text detection and TrOCR for recognition, specifically for Khmer script and mixed-language document.

4. **Improve over existing solutions:** Aim to achieve better performance and reliable across diverse samples than previous research works or OCR tools such as Tesseract.

5. **Provide an easy-to-use model:** Deploy our result allowing enthusiast individual easily accessible to experiment with minimal setup. This approach hope to expand the application of OCR in Khmer language.

1.4 Rationale of the Study

This research is motivated by several compelling factors. First, there is an urgent need to digitize and preserve Cambodia's vast textual heritage, including historical documents, educational materials, and cultural artifacts. Without effective OCR technology for Khmer script, this digitization process remains labor-intensive and prone to errors. Second, the current limitations of OCR systems for Khmer significantly hinder educational and academic initiatives in Cambodia. Many educational institutions struggle to convert physical textbooks and learning materials into digital formats, impacting accessibility and modernization efforts in education. Third, the unique challenges posed by Khmer script from character stacking to the absence of word boundaries present an opportunity to advance the field of OCR technology as a whole. Solutions developed for Khmer may benefit other scripts with similar characteristics. Finally,

improving Khmer OCR technology aligns with broader digital transformation goals in Cambodia, supporting efforts to preserve cultural heritage while enabling more efficient information processing and accessibility in various sectors.

1.5 Limitations and Scope

While this research aims to advance Khmer OCR technology significantly, it is important to acknowledge certain limitations and define the scope of the study:

1. The research focuses specifically on printed text of Khmer and English and does not address handwritten in text recognition context, which presents additional challenges requiring further development.
2. The study primarily considers modern Khmer fonts and typography, with limited coverage of historical or decorative text styles.
3. While the system aims to handle various levels of document quality, the extremely degraded or damaged documents may fall outside the scope of reliable recognition.
4. The study focuses on optical character recognition and does not extend to higher-level natural language processing tasks such as semantic analysis or machine translation.
5. Resource constraints may limit the size and diversity of the training dataset, further efforts will be made to ensure sufficient representation of common use cases.

These limitations help maintain a focused research scope while acknowledging areas that may require future investigation.

1.6 Structure of the Thesis

This thesis is structured into the following chapters:

1. **Introduction**: Establishes the research context, defines objectives, outlines key research questions, explains the study's importance, and sets clear boundaries for the investigation.
2. **Literature Review**: Examines current OCR technologies, identifies specific challenges in Khmer script processing, and explores relevant deep learning methodologies.
3. **Dataset Construction**: Describes our data collection process, synthetic data generation techniques, preprocessing methods, and dataset quality evaluation.
4. **Model Architecture and Experiments**: Details our proposed OCR pipeline, explains model customizations for Khmer script, and outlines training methodologies.
5. **Results and Analysis**: Reports experimental findings, analyzes performance metrics, and compares our system against existing OCR solutions.
6. **Discussion and Future Work**: Interprets results, discusses limitations, and identifies opportunities for future research and improvements.
7. **Conclusion**: Synthesizes key contributions, summarizes main findings, and highlights the broader impact on Khmer OCR development.

Each chapter progressively builds on previous sections to deliver a complete investigation of advanced Khmer OCR technology development.

Chapter 2
Literature Review

2.1 Overview

This chapter provides a comprehensive literature review covering several key aspects of Optical Character Recognition (OCR). It begins with a definition of OCR, followed by an overview of its technological evolution over time. Particular attention is given to the unique challenges associated with Khmer OCR, a low-resource language with complex script characteristics. The chapter also explores the significant role of synthetic data in addressing data scarcity for low-resource language processing and dataset development. Finally, the chapter concludes by identifying and summarizing the existing research gaps in the current literature, highlighting areas that remain under-explored and underscoring the need for further investigation.

2.2 Definition of Optical Character Recognition (OCR)

Optical Character Recognition (OCR) is a field of computer vision and pattern recognition that focuses on the automatic identification and digitization of printed or handwritten text from images, scanned documents, or other visual media (Singh et al., 2012). OCR systems aim to convert visual representations of text into machine-encoded formats, enabling automated indexing, editing, and data extraction (Muaz and LengIeng, 2015b).

Modern OCR technology has evolved significantly from its early rule-based and template-matching roots to incorporate advanced machine learning techniques, particularly deep learning, which allow for improved accuracy in character detection, segmentation, and classification across diverse languages and scripts.

OCR systems typically consist of several key components: image preprocessing (e.g., noise removal, binarization), text detection, character segmentation, feature extraction, and recognition. These systems must be adapted to handle various font styles, image distortions, complex layouts, and script-specific features. While OCR for Latin-based languages has become highly accurate, extending such systems to non-Latin scripts—such as Khmer—remains a significant research challenge due to unique linguistic and structural characteristics.

2.3 Text Detection

Hiremath et al. (2024) The task of detecting text regions in images is a crucial first step in many document analysis pipelines.Character Region Awareness by Baek et al. (2019) proposed an efficientand accurate scene text detection approach by incorporating character region masking and attention.

Hiremath et al. (2024) Some other popular research Methodology for text detection and highly cited text detection algorithms such as:

EAST (Efficient and Accurate Scene Text Detector) Zhou et al. (2017) uses a fully convolutionalnetwork to directly predict word or text line bounding boxes along with their orientation. It was one of the first modern deep learning models for this task. TextBoxes++ Liao et al. (2016) extended the TextBoxes model with more powerful convolutional features and anangle vector to better handle oriented and curved text instances. CRAFT (CharacterRegion Awareness for Text Detection) Baek et al. (2019) akes a different approach, using a regionscore map to localize individual character regions which are then grouped into words/text lines. Mask TextSpotter Liao et al. (2019) combines semantic segmentation + attention-based recognition to directly read text of any shape from natural scenes—all in one unified model. TextFuseNet Ye et al. (2020) is detects text by extracting and fusing features at three levels: character-level, word-level, and global-level. It uses a semantic segmentation branch for global context, detects characters and words through a modified Mask R-CNN pipeline, and combines all levels using a multi-path fusion architecture. This fusion enriches the text representation, enabling the model to better localize and segment text instances of arbitrary shapes. ABCNet Liu et al. (2020) have been proposed which aim to jointly optimize text detection and recognition in a unified architecture. Such unified frameworks potentially allow the two tasks to benefit from each other during training.

Table 2.1: Text Detection Accuracy

Model	**Backbone**	**Dataset**	**Recall (%)**	**Precision (%)**	**F1-score (%)**
EAST*	VGG-16	ICDAR 2015	78.3	83.3	80.7
He et al.	ResNet-50	ICDAR 2015	80.0	82.0	81.0
R2CNN	VGG-16	ICDAR 2015	79.7	85.6	82.5
TextSnake	VGG-16	ICDAR 2015	80.4	84.9	82.6
TextBoxes++*	VGG-16	ICDAR 2015	78.5	87.8	82.9
EAA	ResNet-50	ICDAR 2015	83.0	84.0	83.0
Mask TextSpotter	ResNet-50	ICDAR 2017	81.2	85.8	83.4
PixelLink*	VGG-16	ICDAR 2017	82.0	85.5	83.7
RRD*	ResNet-50	ICDAR 2017	80.0	88.0	83.8
Lyu et al.*	ResNet-50	ICDAR 2017	79.7	89.5	84.3
FOTS	ResNet-50	ICDAR 2017	82.0	88.8	85.3
CRAFT	VGG-16	ICDAR 2015	84.3	89.8	86.9

Source: Results sourced from the CRAFT research paper Hiremath et al. (2024). Results are reported on quadrilateral-type datasets such as ICDAR. Asterisks (*) denote results based on multi-scale tests. The table compares various scene text detection models—including the CRAFT model used in the proposed system—across key performance metrics such as precision, recall, and F1-score.

2.4 Khmer Optical Character Recognition (OCR)

Chey et al. (2006) The first significant research on Optical Character Recognition (OCR) for Khmer script was proposed in 2006, marking a foundational contribution to Khmer language digitization. The study addressed critical challenges inherent to Khmer printed characters, notably the variation in character shapes across fonts and the high visual similarity between certain characters. To overcome these issues, the authors introduced a novel recognition method utilizing Wavelet Descriptors. The approach involved transforming printed character images

into their skeleton forms, then converting these skeletons into the temporal domain. Character templates were generated using wavelet coefficients from a training set, and recognition was performed through a deformable wavelet descriptor and a Euclidean distance classifier. The recognition process selected the template with the smallest distance to the input character as the result. Interestingly, deformation was later excluded from the final method due to its adverse impact on distinguishing similar characters. Experimental evaluation demonstrated promising results: recognition rates of 92.85%, 91.66%, and 89.27% for 22-point, 18-point, and 12-point font sizes respectively, across 10 Khmer fonts. A second test used a 21-page document scanned at varying resolutions (150, 300, 600 dpi) and fax inputs, achieving recognition rates of 92.99% at 300 dpi, 88.61% at 150 dpi, and 80.05% for faxed documents. This pioneering work laid the groundwork for future Khmer OCR systems and remains a landmark study in Khmer script recognition.

Sok and Taing (2014) The second significant research on Khmer Optical Character Recognition (OCR) introduces a Support Vector Machine (SVM)-based approach for printed Khmer character recognition in bitmap documents. Given the inherent complexity of the Khmer script —which includes 74 alphabets and the potential for up to five vertical levels in word compounds —the study focuses on identifying the most effective SVM kernel for classification.

The proposed method evaluates three SVM kernels: Gaussian, Polynomial, and Linear. Rather than training on large datasets, the system adopts a modular approach, segmenting characters into smaller parts. Each segment is transformed into a binary matrix, with 1s representing black pixels and 0s for whitespace. Feature extraction is applied to these matrices for SVM training and classification.

Following recognition, post-processing rules are employed to merge character clusters and correct common misclassifications based on character levels. The training utilized one font ("Khmer OS Content") at 32pt and tested recognition performance across three font sizes (28pt, 32pt, and 36pt). The results demonstrated strong accuracy:

- 98.17% for 28pt,
- 98.62% for 32pt,
- 98.54% for 36pt.

The Gaussian kernel outperformed the other kernels, confirming its suitability for Khmer character recognition. This study highlights the effectiveness of machine learning-based OCR for complex scripts like Khmer and marks a major advancement over earlier template-based systems.

Meng and Morariu (2015) proposed a Khmer Character Recognition (KCR) system utilizing artificial neural networks to address the complexity of recognizing individual Khmer script characters. Developed in a MATLAB environment, the system integrates a Self-Organizing Map (SOM) for unsupervised clustering with a Multilayer Perceptron (MLP) trained via the backpropagation algorithm for supervised classification. The recognition process follows a two-stage pipeline: first, the input image—resized to 20×20 pixels—is categorized by the SOM into one of nine coarse groups. Then, each group is assigned to a dedicated MLP model, which performs fine-grained classification into one of 82 target classes, including Khmer consonants, vowels, and numerals. This modular approach aims to enhance classification efficiency by narrowing the decision space for each MLP. The system achieved an average recognition accuracy of 65% on the training dataset and 30% on the testing dataset, highlighting both the potential and the challenges of neural network-based Khmer OCR at the time.

Muaz and LengIeng (2015a) This study presents a comprehensive OCR system for printed Khmer text using the HTK Toolkit, covering the full pipeline of pre-processing, segmentation, recognition, and mapping. The system is tailored for the widely used Limon S1 font at 22pt

and introduces a structural segmentation approach that decomposes characters into five categories: Main Body, SuperScript, SubScript, CCDown, and Complex Character (CC). Feature extraction relies on vertical framing and Discrete Cosine Transform (DCT), with classification handled by Hidden Markov Models (HMMs). The system was trained on over 35,000 labeled shape samples and evaluated on 10 pages of scanned newspaper text, achieving an overall recognition rate of 96.34%. While the results are promising, the system is limited by its reliance on a single font style and size, making it less robust to font variation. Additionally, recognition performance drops for complex shapes like CC (70.88%) and depends heavily on manually crafted mismatch and rule files for post-processing, indicating scalability and generalization challenges for real-world applications involving diverse document types and fonts.

Valy et al. (2018) proposed a character-level Convolutional Neural Network (CNN) model for recognizing ancient Khmer script from digitized palm leaf manuscripts. The study focuses on two core tasks: isolated character recognition and word-level glyph localization. For the character recognition task, a CNN architecture comprising three convolutional blocks followed by a linear classifier was developed. The output layer of the network predicts one of 106 character classes, achieving a reported test accuracy of 95.96%. In addition to CNNs, the study also explores other neural architectures including LSTM-RNN and hybrid CNN-RNN models. For the word/text image recognition task, the authors leveraged both one-dimensional and two-dimensional RNNs to handle the complex spatial structure of Khmer script and to localize glyphs within variable-length text patches. This research highlights the potential of deep learning approaches in historical Khmer OCR, particularly in handling the challenges posed by degraded manuscripts and complex writing structures.

Annanurov and Noor (2018) conducted a pilot study on Khmer handwritten symbol recognition, focusing on the use of Convolutional Neural Networks (CNNs) for digitizing large-scale handwritten document corpora. The study used image data from six handwriting datasets containing 33 Khmer consonants and 17 vowels, forming a total of 561 syllables. For consonant recognition, the authors trained 33 individual CNNs—one for each root radical—which were later integrated into a recognition assembly. The performance of the CNN-based model was evaluated against two alternative systems: an artificial neural network (ANN) using the full feature set, and another ANN with reduced feature dimensions. Feature extraction techniques such as two-dimensional Fourier transformation (FT2D) and Gabor filters were employed for dimensionality reduction. The CNN-based approach achieved a recognition accuracy of up to 94.85%, outperforming the ANN-based models. However, the system was limited to recognizing only Khmer consonants, without support for vowel or syllable-level recognition.

Sokphyrum and Samak (2019) fine-tuned the Tesseract OCR engine (version 4.0) to improve recognition of Khmer Unicode and legacy Limon fonts. Tesseract 4.0 employs a deep convolutional recurrent neural network with a connectionist temporal classification (CTC) loss function and attention mechanism, enabling it to recognize entire text-line images. The fine-tuning process involved training on 14 Khmer Unicode fonts and 20 pre-Unicode Limon fonts (all at 12pt size). The ISRI Analytic Tools were used to evaluate OCR accuracy at both the character level (CHL) and cluster level (CLL) by comparing OCR outputs to ground truth text. The Khmer pre-trained engine achieved an average accuracy of 87.49% at the character level and 89.43% at the cluster level on Unicode fonts, while legacy Limon fonts yielded a significantly lower median accuracy of 62.27%. After fine-tuning, recognition accuracy reached up to 90% for specific fonts, demonstrating the potential of domain-specific adaptation to enhance OCR performance for complex scripts like Khmer.

Buoy et al. (2022) Khmer printed character recognition presents significant challenges due to the script's complex structure, including stacked consonants, dependent vowels, and diacritics placed in various positions. Traditional approaches, such as SVMs, wavelet-based templates, and CNNs, have largely focused on recognizing standalone characters and depend heavily on accurate segmentation, making them less robust for noisy or continuous text-line images. Tesser-

act OCR, though improved with deep neural networks and CTC loss, still performs poorly on heavily augmented images with a reported Character Error Rate (CER) of 35.9%. In contrast, the reviewed study introduces an end-to-end attention-based Sequence-to-Sequence (Seq2Seq) model that directly maps entire text-line images to character sequences without needing pre- or post-processing. Trained on 3 million synthetically generated and augmented images across multiple Khmer fonts, the model achieved a CER of 0.7% on noisy images and 0.24% on clean images, significantly outperforming existing methods and demonstrating the effectiveness of attention mechanisms and deep learning for low-resource scripts like Khmer.

Nom et al. (2024) The Khmer script poses significant challenges for scene text detection and recognition due to its complex structure involving stacked consonants, multiple diacritics, subscript characters, and the lack of explicit word boundaries. To address the scarcity of resources for this low-resource language, this study introduces KhmerST, the first annotated scene-text dataset for the Khmer language, consisting of 1,544 real-world images captured from various public settings in Cambodia. Unlike existing synthetic Khmer datasets, KhmerST includes indoor and outdoor images with diverse fonts, orientations, and background conditions, and provides polygon-based line-level annotations for more accurate localization. Benchmarking with YOLO models shows that YOLOv8 performs best in detection tasks due to its ability to handle small and variable text elements, while in recognition tasks, both TrOCR and Tesseract perform poorly, with TrOCR achieving CER of 0.90 and Tesseract CER of 1.30, highlighting the need for Khmer-specific OCR models. The study underscores the gap between current OCR systems and the needs of Khmer script recognition, calling for future research on synthetic data generation, Khmer-aware model design, and multimodal approaches to improve accuracy in real-world applications.

Buoy et al. (2025) This research addresses the challenge of Khmer textline recognition, which is particularly difficult due to the absence of spaces between words in the Khmer script. Unlike Latin-based languages, this structure requires recognition to be performed at the textline level, leading to high latency when using autoregressive (AR) decoders that predict one character at a time while considering previous characters. In contrast, non-autoregressive (NAR) decoders can decode all characters in parallel but lack awareness of character dependencies, resulting in lower linguistic accuracy. To solve this, the authors propose an efficient Khmer textline recognition method using an NAR decoder, enhanced by Khmer-specific subword modeling. Instead of predicting single characters, the model recognizes character clusters (subwords), capturing the syntactic, morphological, and orthographic structure of the language implicitly. This design retains the speed of NAR models while improving accuracy. Experimental results show that the proposed method outperforms the character-level NAR baseline in accuracy and matches or exceeds the AR baseline while maintaining lower latency. However, the abstract does not report detailed accuracy or metrics, and access to full experimental results requires a subscription, limiting transparency for non-subscribers. This work contributes an important step toward faster and more linguistically aware Khmer OCR by bridging the gap between speed and accuracy in scene text recognition.

Khmer OCR research has progressed significantly, yet the script's structural complexity —including stacked consonants, subscript forms, diacritics, and the absence of spaces between words—continues to pose major challenges. Early methods relying on wavelet descriptors, SVMs, and HMMs showed moderate success but struggled with font variations, segmentation, and real-world noise. Deep learning has since enabled end-to-end approaches like attention-based Sequence-to-Sequence (Seq2Seq) models, achieving state-of-the-art accuracy with character error rates (CER) as low as 0.24% on clean images and 0.7% on noisy inputs. However, general-purpose engines like Tesseract still perform poorly, especially in scene text scenarios, as highlighted by results from the newly introduced KhmerST dataset. A promising direction involves non-autoregressive (NAR) models enhanced with Khmer-specific subword modeling, which offer faster inference and comparable or better accuracy than autoregressive (AR) mod-

els. Despite these advances, many studies lack open access to detailed results, and gaps remain in handwritten recognition, font diversity, and domain adaptation. Future research must prioritize Khmer-aware model design, synthetic data generation, and robust multimodal solutions to improve performance in real-world OCR applications.

Author & Year	Technique	Dataset	Accuracy	Limitation
Chey - 2006	Wavelet Descriptors Euclidean Distance	10 Khmer fonts	92.99%	Template-based, limited generalization
Sok - 2014	SVM with Gaussian Kernel	1 font (Khmer OS)	98.62%	Single font; limited font diversity
Meng - 2014	SOM + MLP Neural Networks	82 classes (char)	30%	Low test accuracy
Muaz - 2015	HMM with DCT features using HTK	Limon S1 font	96.34%	Single font; weak on complex shapes
Valy - 2018	CNN (3 conv blocks) + RNN (1D & 2D for glyph localization)	Palm leaf 106 classes	95.96%	Historical texts; focused on ancient script
Annanurov - 2018	CNN vs. ANN with FT2D/Gabor filters	3366 handwritten samples	94.85%	Consonants only; no vowel/ syllable recognition
Sokphyrum - 2019	Fine-tuned Tesseract V-4.0	14 Unicode + 20 Limon fonts	Unicode: 89.43% Limon: 62.27% Fine-tuned: 90%	Limon performance poor without tuning; line-level only
Rina - 2022	End-to-end Sequence-to-Sequence (Seq2Seq)	3 millions images	CER 7%	Repeated characters before EOS token was reached
Nom - 2024	YOLOv10 + (TrOCR, Tesseract)	1,544 real images	CER 0.90, 1.30	Build Dataset only
Rina - 2025	Non-Autoregressive (NAR) Predicts character clusters (subwords) in parallel	Not explicitly detailed need subscription	Comparable or better than: Autoregressive (AR)	Not reported in the abstract need subscription to see detail

Figure 2.1: Summary of the literature review, which includes the types of models used, the datasets used, the metrics used to evaluate the performance of the models, and the results of the experiments. The table highlights the key findings of the literature review, including the challenges faced when developing Khmer OCR models and the need for Khmer-specific models and datasets.

2.5 Challenges in Khmer OCR

Khmer OCR poses numerous technical challenges stemming from the script's unique structure and linguistic features. Unlike Latin-based scripts, Khmer characters can include complex stacking of consonants (e.g., subscript consonants using Coeng), placement of diacritics, and variable vowel positions that appear above, below, before, after, or even wrap around base characters (Buoy et al., 2021). This spatial complexity makes segmentation particularly difficult and often unreliable, especially in cluttered or noisy environments. Accurate character separation is further complicated by the fact that multiple elements can visually overlap, causing traditional segmentation-based approaches to fail.

Font diversity also adds a layer of complexity. Khmer is written using a wide array of fonts, each introducing variations in stroke thickness, character proportions, and spacing. These stylistic differences can drastically alter a character's appearance, requiring OCR systems to be highly font-invariant (Buoy et al., 2021). Yet, many models are trained on limited font sets, leading to poor generalization across unseen styles.

Another critical issue is the scarcity of annotated datasets. As a low-resource language, Khmer lacks the extensive labeled corpora available for Latin or Chinese scripts. This data scarcity hampers the training of robust models and forces researchers to rely on synthetic data or heavily augmented datasets to simulate real-world variability.

Additionally, many legacy OCR systems for Khmer rely on rule-based or modular pipelines with explicit character segmentation, manual pre- and post-processing, and assumptions about text structure. These approaches are brittle and poorly suited for real-world applications where

scanned documents may contain noise, skew, blur, uneven lighting, or distorted text lines. Their reliance on handcrafted rules also limits scalability and adaptability to other domains.

Lastly, the lack of word boundaries in Khmer—where text is often written without spaces—presents an additional recognition barrier. This linguistic feature necessitates full line-level recognition, which increases computational complexity and requires models to understand broader contextual dependencies across characters.

2.6 Role of Synthetic Data

To overcome the challenges associated with data scarcity in Khmer OCR, recent studies have emphasized the critical role of synthetic data generation. A prominent strategy involves leveraging the open-source `text2image` utility from the Tesseract OCR engine to create high-quality, rendered images of Khmer text lines (Buoy et al., 2021). These images are generated from a curated text corpus that includes a diverse mix of numbers, words, phrases, and full sentences, providing broad linguistic coverage.

Multiple commonly used Khmer fonts are applied during rendering to simulate font diversity and improve the model's ability to generalize across different typographic styles. Each rendered image is converted to grayscale and resized to a fixed height (e.g., 32 pixels) to meet the input dimensional requirements of neural network models, while maintaining proportional width to preserve text structure.

To emulate real-world conditions and enhance robustness, extensive data augmentation techniques are applied. These include Gaussian blurring, morphological operations like dilation and erosion, speckle and blob noise injection, background texture overlays, rotational distortions, and random concatenation of multiple text-line images. Each augmentation has a 50% probability of being applied, with combinations occurring dynamically during training to maximize variability and minimize overfitting.

The resulting synthetic dataset scales to millions of samples, covering a wide spectrum of distortions, font styles, and text structures. This scale and diversity enable deep learning models—particularly end-to-end architectures such as attention-based Sequence-to-Sequence networks—to learn robust visual and contextual features, improving performance on both clean and noisy inputs. Ultimately, synthetic data serves as a vital resource for building Khmer OCR systems capable of handling the script's inherent complexity in the absence of large, annotated real-world datasets.

2.7 Summary of Research Gaps

Despite recent progress in Khmer Optical Character Recognition (OCR), several critical gaps persist in the current body of research, hindering the development of robust and generalizable systems. First and foremost is the issue of data scarcity—there is a lack of large-scale, high-quality annotated datasets for Khmer, particularly for scene text and handwriting, which restricts model training, benchmarking, and cross-domain evaluation. Although synthetic datasets have partially mitigated this, they cannot fully replicate the variability and unpredictability of real-world documents. Second, the complex structural features of Khmer script—such as stacked consonants, overlapping vowel markers, and the absence of explicit word boundaries—are insufficiently addressed in many existing models, especially those adapted from Latin-script OCR frameworks. These models often fail to capture the script's spatial dependencies and morphological nuances. Third, font variability remains a major bottleneck: Khmer documents are written in a wide range of stylistic fonts, and OCR systems trained on limited font sets often generalize poorly. Lastly, there is a lack of robustness to real-world document

conditions, including noisy backgrounds, image blur, skew, and low resolution. Many current systems, including Tesseract and some CNN-based models, show significant performance drops under such conditions. Addressing these gaps—through Khmer-specific model design, comprehensive dataset creation, and more resilient training strategies—is essential for building high-performance OCR systems tailored to the Khmer language and its real-world use cases.

Chapter 3
Dataset

3.1 Khmer Text Data Collection

The dataset collection process began with gathering Khmer word-by-word data from the Chuon-Nath Dictionary, which provided over 50,000 words. However, it was recognized that sentence-by-sentence data was also necessary for comprehensive language modeling. To address this, a web scraping script was developed using the BeautifulSoup library to collect sentence data from the website khsearch.com. This resulted in the collection of approximately 500,000 Khmer sentences.

Upon analyzing the collected data, it was found that there was an imbalance between Khmer and English language data. To address this, the Alpha-Word dataset was acquired, which contained over 300,000 English words. Furthermore, the Google-Word dataset was also collected, which provided a large volume of natural language data commonly used on the internet. Additionally, the Hugging Face dataset was used to supplement the collection with more natural language data.

In total, the dataset collection process yielded over 1.5 million character/symbols/words/ sentences, including Khmer word-by-word data, English word-by-word data, Khmer sentences by sentences, English sentences by sentences, and natural language data from the internet.

3.2 Data Manual Collection

In this step we collection dataset manually from the web and other sources. we use our own data annotation application to collect the data. The data is not just for Optical Character Recognition (OCR) but also for text detection as well.

3.3 Text Cleaning and Preprocessing

The text data collected from the Chuon-Nath Dictionary, Alpha-Word, and Google-Word datasets was found to be clean and required no preprocessing. However, the text collected from internet scraping from khsearch.com was not clean and required preprocessing to enhance OCR performance. The text data was analyzed and any uncommon links or URLs, excessive spacing, tabs, and invisible characters were removed. Furthermore, the sentences in the text data were found to be too long, so the text data was segmented into word-by-word format using the khmer-nltk library and then random sentences were generated from 1 to 110 characters in length, while maintaining the order of the sentences. This was done to ensure that the OCR model could predict missing characters based on the natural order of the sentences. Additionally, the text data was normalized to ensure that all characters were in the same format, which is important for OCR performance. The normalization process involved converting all

characters to lowercase and removing any non-alphanumeric characters. After preprocessing, the dataset was split into two parts: a training set and a validation set. The training set was used to train the OCR model, while the validation set was used to evaluate the performance of the OCR model.

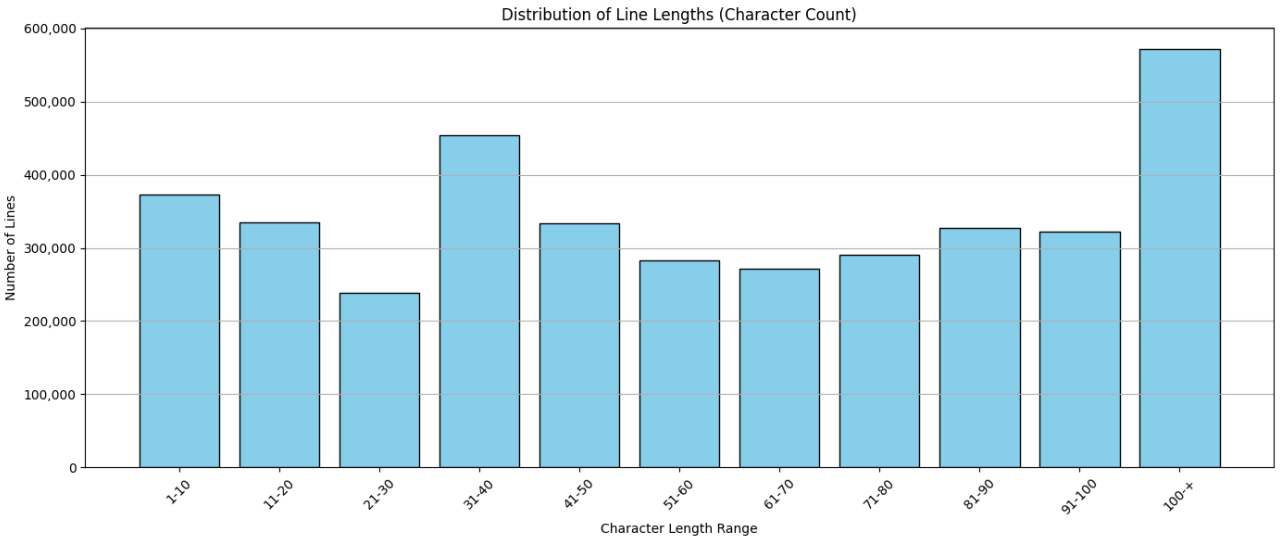

Figure 3.1: Frequency of text length in the synthetic dataset, showing the distribution of text lengths. The majority of the text lengths are between 1-50 characters, with a peak at around 20-30 characters. The longer text lengths are less frequent, but still present in the dataset.

3.4 Image Generation Pipeline

The synthetic dataset was generated by loading text from data_khmer_text.txt line by line and applying different fonts using the Pillow library. The aim of this step was to generate a wide variety of text styles and fonts, so that the OCR model could be trained on diverse text styles. To achieve this, approximately 50 different font styles were applied to each line of text. Additionally, each line was combined with 20 different background images to simulate real-world image text. This was done to ensure that the OCR model could recognize text regardless of the background of the image.

Various types of noise were applied to the text images to simulate real-world conditions. This included gaussian blurring, dilation and erosion, blob noise, speckle noise, multi-scale noisy backgrounds, random concatenation of augmented images, and salt-paper noise. The text on the images was rotated from -3 to 3 degrees to simulate variations in text orientation. Additionally, margins of 1-5 pixels were randomly added to the text images to account for inconsistent text detection. As each text line could generate two images, the total number of images produced was 3 million.

The resulting synthetic dataset contained 3 million images, each with a unique combination of font, background, and noise. The images were designed to simulate real-world conditions, such as text orientation, text margins, and various types of noise. The application of these techniques resulted in a high-quality synthetic dataset that was suitable for training the OCR model. When examining the resulting synthetic dataset, it is clear that the images are highly diverse and realistic, with a wide range of font styles, colors, and backgrounds. Additionally, the noise types and levels are varied, which will help to improve the robustness of the OCR model when trained on this dataset. It is also clear that the images are of high quality, with crisp and clear text and minimal artifacts. The overall quality of the dataset is high, which

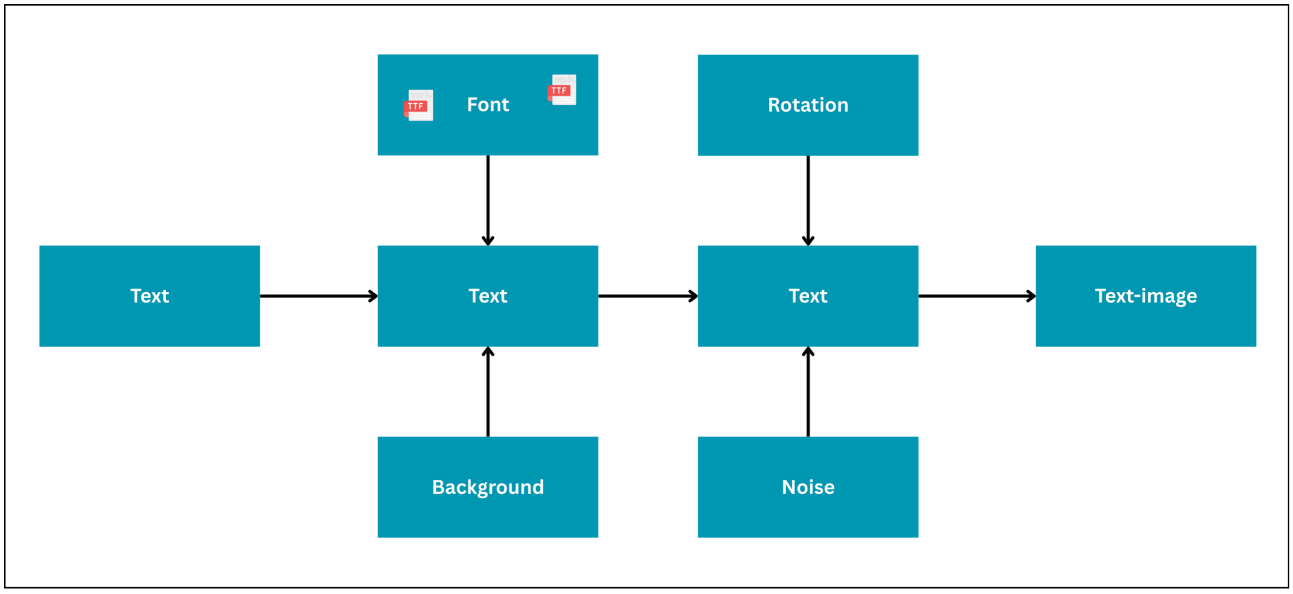

Figure 3.2: Example of a synthetic image generated for the OCR training dataset, illustrating the application of random fonts, backgrounds, and noise to simulate real-world conditions.

will help to ensure that the OCR model is well-trained and can accurately recognize text in a variety of contexts.

Figure 3.3: Result of the synthetic dataset generation pipeline, showing the diversity of fonts, backgrounds, and noise types.

Chapter 4
Model Architecture and Experiments

4.1 Experimental Environment and Tools

All experiments in this study were conducted on a high-performance workstation running Ubuntu 20.04 LTS. The system was equipped with dual NVIDIA RTX 4090 GPUs, each with 24 GB of VRAM, and 128 GB of system RAM, ensuring ample memory and computational power for training large-scalec deep learning models efficiently.

The models were implemented using the PyTorch deep learning framework, with integration of the Hugging Face Transformers library for state-of-the-art model architectures such as TrOCR. GPU acceleration was enabled via CUDA, allowing for fast and parallelized training across the two GPUs.

For experiment tracking and metric logging, MLflow was used. It captured key training and evaluation metrics such as character error rate (CER), word error rate (WER), training loss, and validation loss in real-time. This facilitated better experiment management and reproducibility, especially when comparing multiple model versions or hyperparameter configurations.

4.2 Model Architecture and Configuration

We propose the end-to-end Khmer & English OCR solution using a CRAFT model combine with TrOCR architecture, as shown in Figure 4.1 and 4.2. In this section we will describe the details of the model architecture and configuration from document image to digital text output in whole pipeline.

4.2.1 CRAFT for Text Detection

For the text detection stage, we adopted the Character Region Awareness for Text Detection (CRAFT) model, which is well-regarded for its ability to detect text at the character level rather than relying solely on word-level bounding boxes. CRAFT produces dense predictions of character regions and affinity scores, enabling it to localize irregular and closely spaced text lines—an essential requirement for handling complex scripts like Khmer.

The CRAFT model architecture consists of a VGG16-based backbone followed by a series of convolutional layers to produce two output maps:

- A **region score map**, indicating the likelihood of each pixel belonging to a character region.
- An **affinity score map**, capturing the spatial relationships between adjacent characters to form text lines.

In this study, we fine-tuned a pretrained CRAFT model on a custom Khmer dataset composed of synthetic and real-world scene text images. We applied data augmentation techniques

such as rotation, scaling, blurring, and illumination changes to increase the model's robustness. The input images were resized to a fixed height of 768 pixels while preserving the aspect ratio.

We used the official implementation of CRAFT with some modifications to better support Khmer character characteristics, such as tight spacing, stacked glyphs, and diacritic marks. The model was trained using Adam optimizer with a learning rate of 1e-4 and batch size of 16. Early stopping and validation-based checkpointing were employed to prevent overfitting.

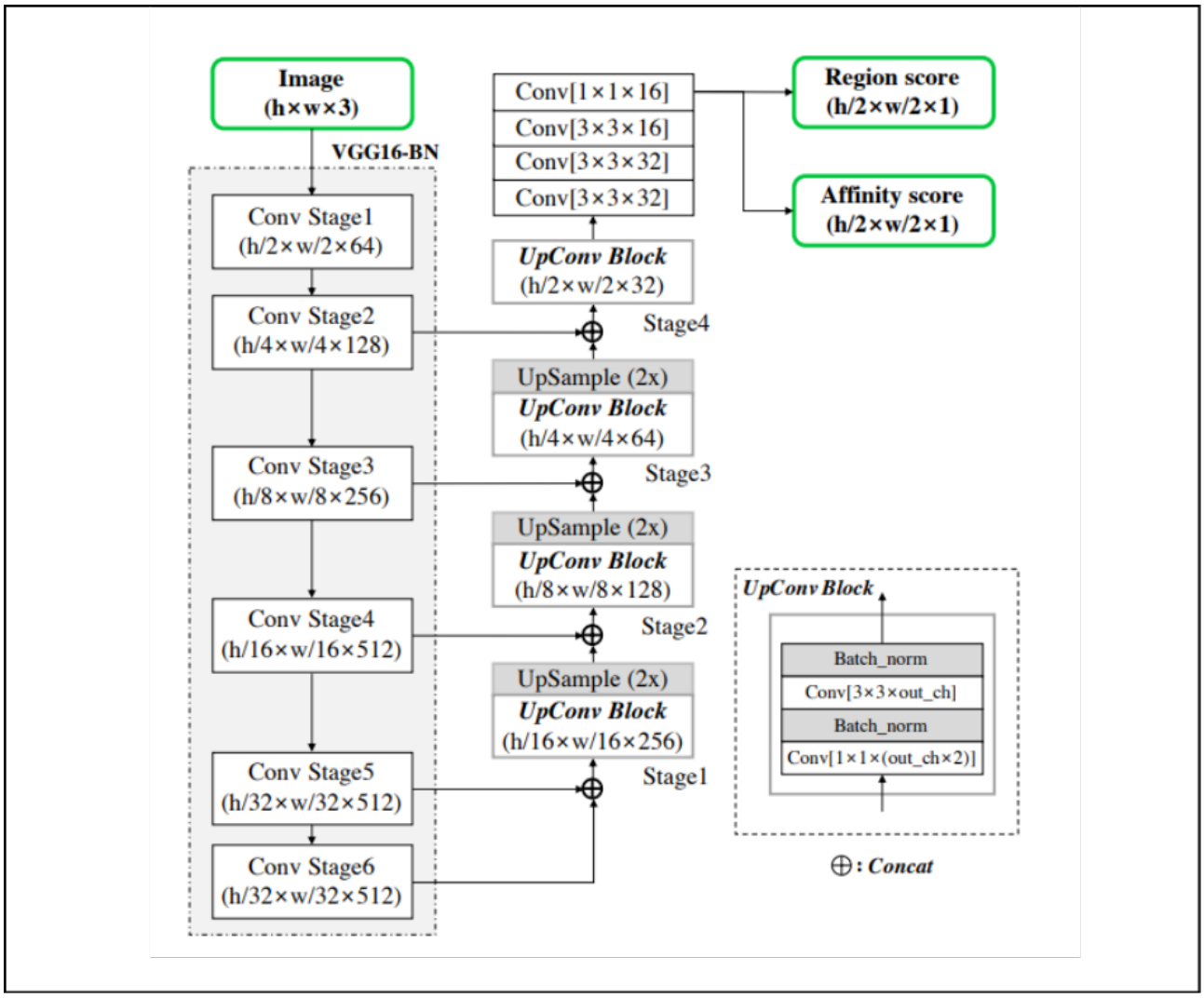

Figure 4.1: Illustration of the CRAFT model architecture used for text detection. The model consists of a VGG16-based backbone followed by a series of convolutional layers to produce two output maps: a region score map and an affinity score map. The model is fine-tuned on a custom Khmer dataset composed of synthetic and real-world scene text images. bae

The model has 20,770,466 trainable parameters. The output of the CRAFT detector was then passed to the text recognition model (TrOCR), enabling a two-stage pipeline for accurate and end-to-end Khmer text reading in general scenes.

4.2.2 TrOCR for Text Recognition

For the text recognition component of the pipeline, we used the **TrOCR** model, a transformer-based OCR system proposed by Microsoft Research. TrOCR stands for *Transformer-based Optical Character Recognition*, and it integrates a vision encoder with a language decoder in a unified encoder-decoder (Seq2Seq) architecture, following the structure of the Vision Transformer (ViT) and pre-trained language models like BART.

The TrOCR model takes the cropped text-line image detected by CRAFT and processes it through a **ViT-based encoder**, which extracts rich visual features. These features are then passed to the **transformer decoder**, which generates the text sequence token-by-token, using cross-attention to focus on relevant image features while decoding. This allows the model to handle complex scripts like Khmer with better accuracy and context-awareness.

We fine-tuned the base version of TrOCR using the Hugging Face `transformers` library on our custom synthetic Khmer dataset. During training, we tracked key metrics such as Character Error Rate (CER) and Word Error Rate (WER) using MLflow. The model demonstrated strong generalization across different fonts and text conditions, benefiting from both large-scale pretraining and our domain-specific fine-tuning.

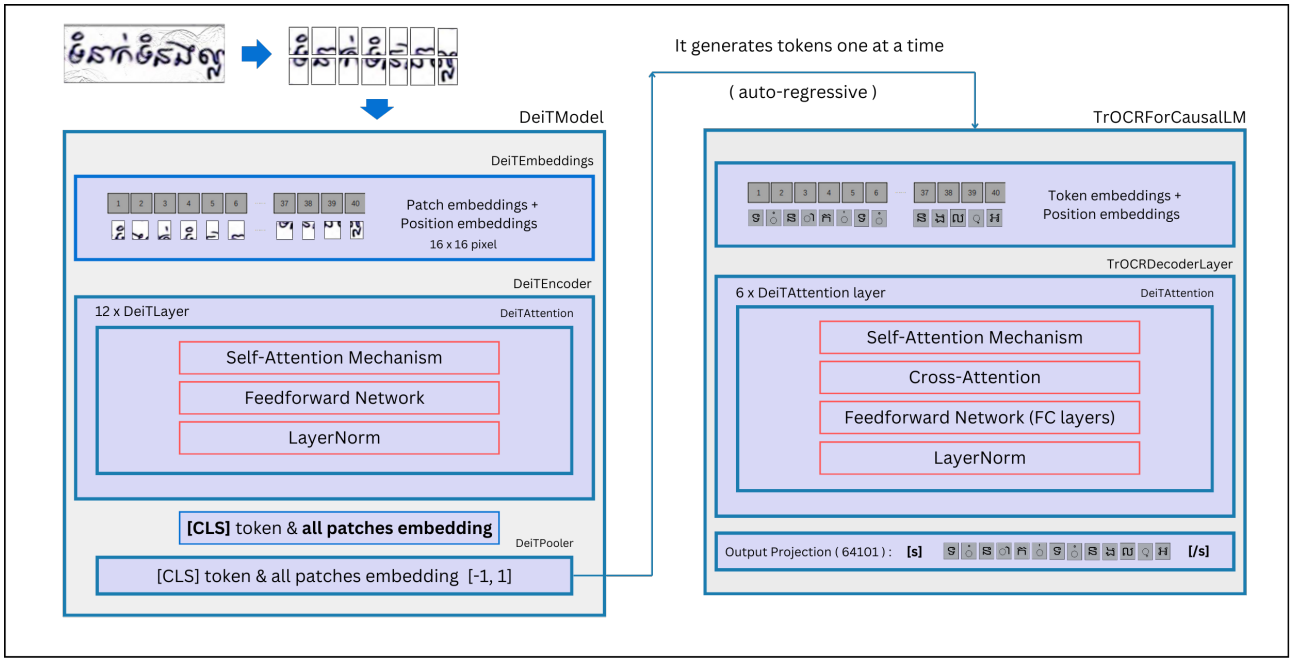

Figure 4.2: Illustration of the TrOCR model architecture used for text recognition.

4.2.3 The end-to-end pipeline

The overall workflow is given in Figure 4.3. It starts with the image-of-text inputs which are English and Khmer image-text, or even mixing language image-text, the pre-processing step is converted image into grayscale because we want to down-stream the task, to make model prediction more accurate. Then the CRAFT model detects the text region in the image, it's detect text-line based, we using post processing stop for reordering the detected text-line to make sure the detected text-line is in the correct order. After that, the detected text-line is cropped and passed to the TrOCR model for text recognition. The output of the TrOCR model is the recognized text.

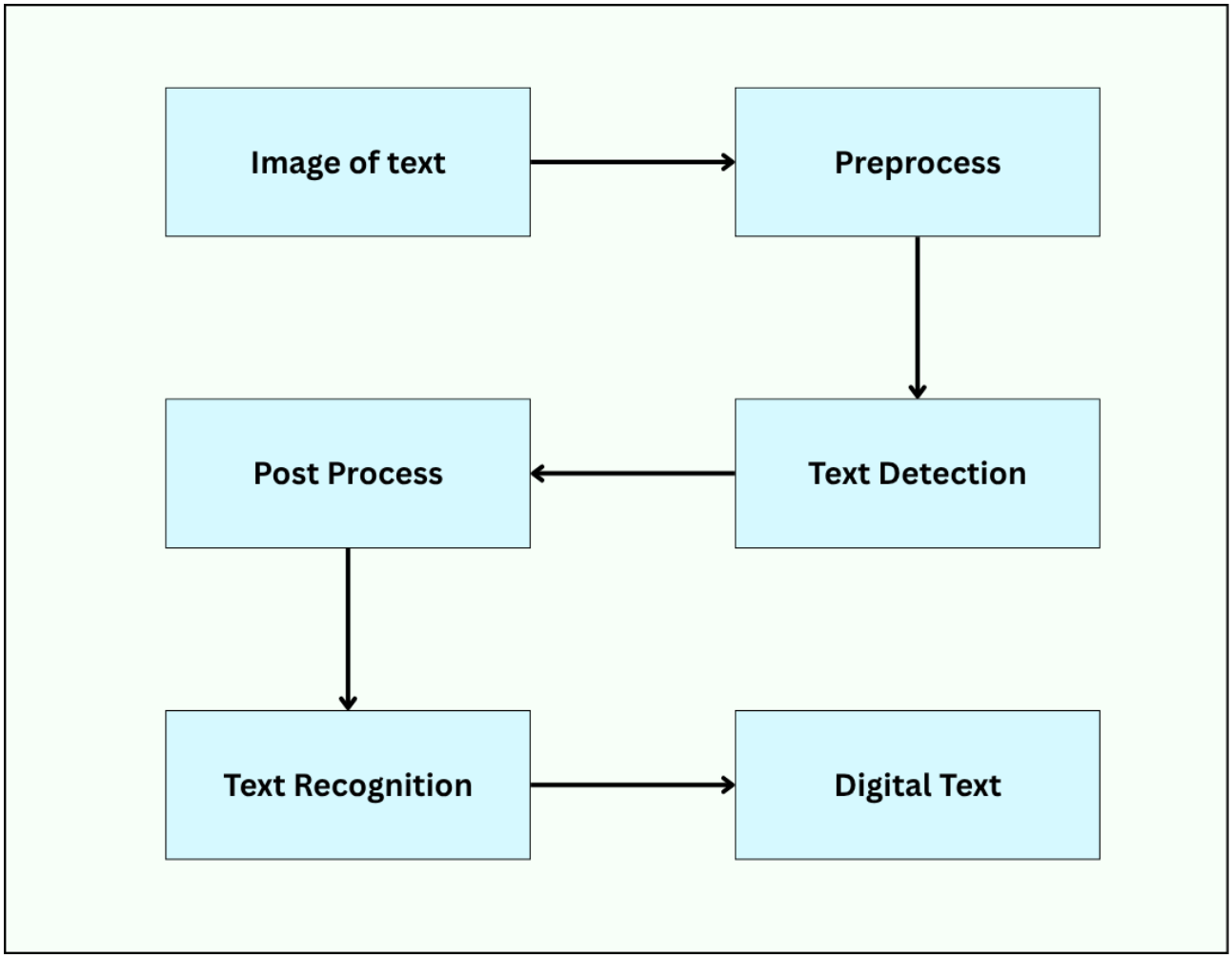

Figure 4.3: End-to-end OCR inference pipeline for converting text images into digital text.

4.3 Training Methodology

This section delves into the training strategies and processes employed to optimize the performance and accuracy of the OCR models. We describe the fine-tuning procedures for both the CRAFT text detector and TrOCR text recognizer, as well as the key hyperparameters and metrics used to evaluate their performance. Additionally, we present a discussion on the importance of robust training and the approaches taken to ensure the models generalize well across different fonts, text conditions, and domains.

4.3.1 Fine-tuning Configuration for CRAFT

The CRAFT model was fine-tuned on a custom Khmer text dataset using weak supervision, leveraging the strengths of annotated real-world images. Throughout the training process, a variety of data augmentation techniques were employed to enhance the model's robustness against font diversity, image noise, and other real-world distortions. These augmentations included random rotation of text instances by up to 20 degrees, random cropping with varying scales and aspect ratios, and horizontal flipping to introduce mirrored perspectives. Additionally, color jittering was applied, involving simultaneous adjustments in brightness, contrast, saturation, and hue, each parameter being varied by a factor of 0.2. These transformations enriched the training set with diverse visual variations, which in turn helped the model generalize more effectively to unseen examples.

To further strengthen the model's discriminative capabilities, we incorporated contrastive learning alongside adversarial training. These approaches were particularly beneficial in enabling the model to distinguish between visually similar Khmer characters and different font styles. Transfer learning also played a pivotal role in adapting the model to the Khmer script. Specifically, we fine-tuned the pretrained model using a manually annotated dataset comprising 4,000 bounding boxes that captured the intricate structure of the Khmer script, including its unique diacritics and ligatures.

The most important configuration parameters for training the CRAFT model are summarized in Table 4.1. These include the selected training mode, backbone architecture, initialization weights, loss function, normalization scheme, and optimization hyperparameters. Each setting was carefully selected to achieve optimal performance on the Khmer text recognition task.

Parameter	**Value**
Backbone architecture	VGG
Pretrained weights	`CRAFT.pth`
Batch size	8
Training iterations	0 to 10,000
Evaluation interval	Every 500 iterations
Learning rate	0.0001

Table 4.1: Key configuration parameters for training the CRAFT model on a custom Khmer dataset using weak supervision. The parameters include the training mode, backbone architecture, dataset, batch size, training iterations, evaluation interval, learning rate, learning rate decay, weight decay, mixed precision, loss function type, negative ratio, minimum negative samples, output image size, normalization mean and standard deviation, region and affinity enlargement factors, Gaussian kernel initialization size, and Gaussian sigma.

4.3.2 Customizing TrOCR Processor

To make the TrOCR model understand Khmer tokens, we customized the processor without modifying the original model architecture. We developed a CustomKhmerTokenizer based on a predefined list of unique Khmer and English characters, including special tokens such as *<s>*, *</s>*, and *<pad>*, to handle the encoding of text into token IDs and the decoding of token IDs back into readable text. This tokenizer was then wrapped inside a custom processor that mimics the behavior of Hugging Face's TrOCRProcessor, allowing seamless integration with TrOCR's image processing pipeline. By doing so, we enabled the model to work with Khmer script during both training and inference, while preserving the original TrOCR model's structure and leveraging its pretrained capabilities.

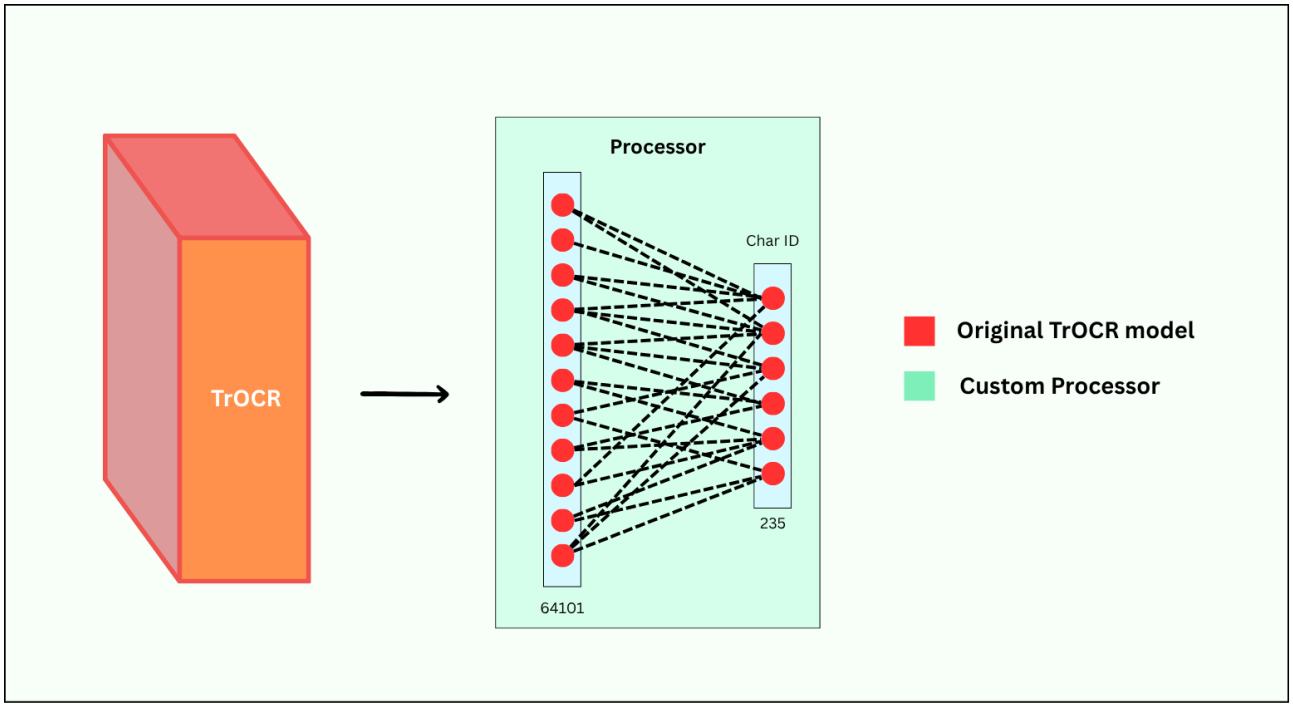

Figure 4.4: Architecture of the customized TrOCR pipeline. The original TrOCR model is extended with a custom processor, which includes two additional fully connected layers (FC1 and FC2) tailored for Khmer character decoding.

4.3.3 Fine-tuning TrOCR Model

For the TrOCR model, we selected the base model because we were working with a multilingual dataset. We wanted the model to have a large parameter size, so we chose the base model for this task.

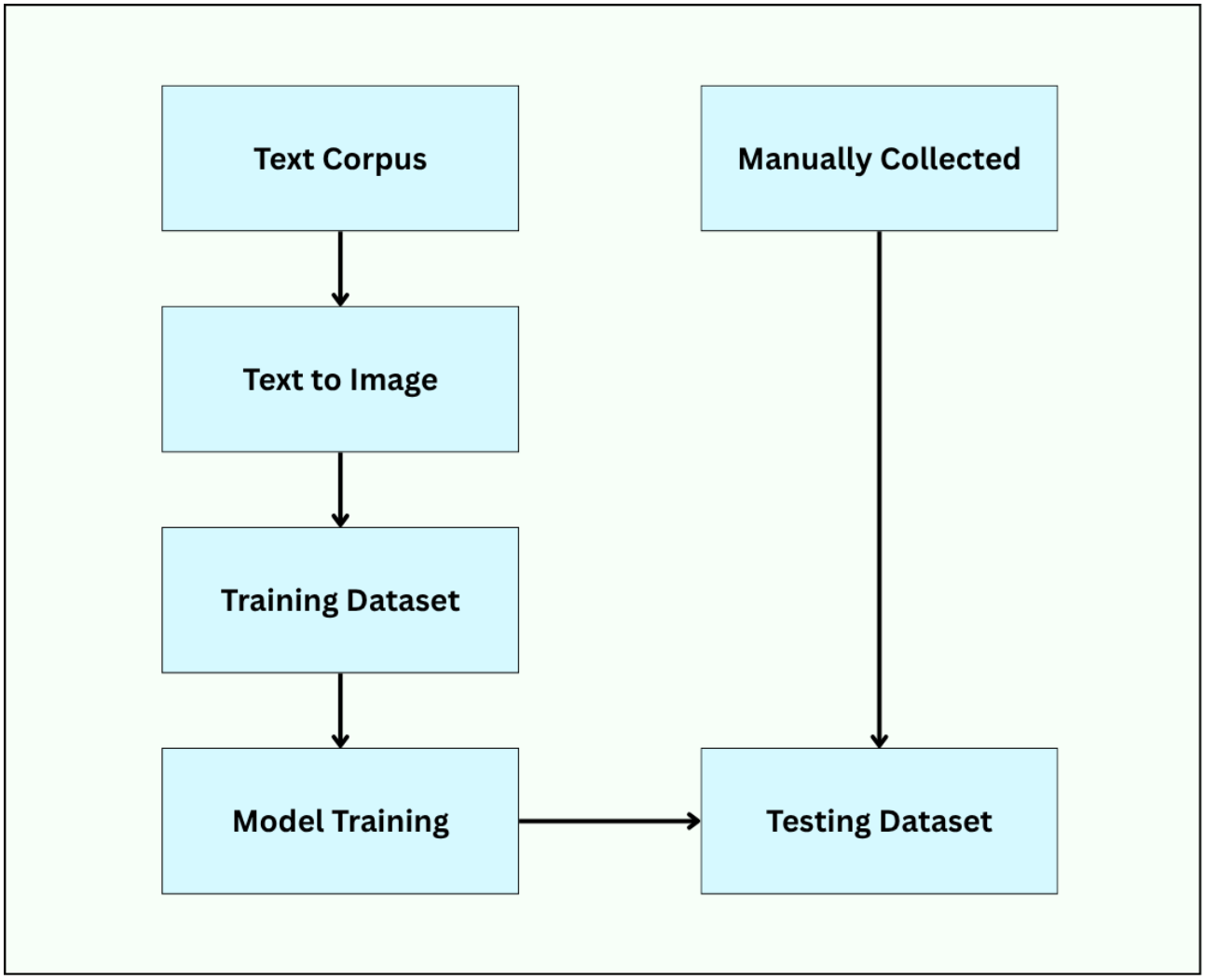

Figure 4.5: Overview of the TrOCR training process. Text-generated images form the training set, while real-world samples are manually collected for testing and evaluation.

After experimenting with different hyperparameters, we found the desired results. Here are the desired hyperparameters: batch size = 1024, learning rate = 0.0001, epoch = 2, dataset = 3.5 million images. We chose a batch size of 1024 because we wanted the model to generalize well. This is a very important hyperparameter because we used to train with 8, 16, 32, 128, 256, but it was not generalizing as expected; it was really weak to overfitting. That's why we chose 1024.

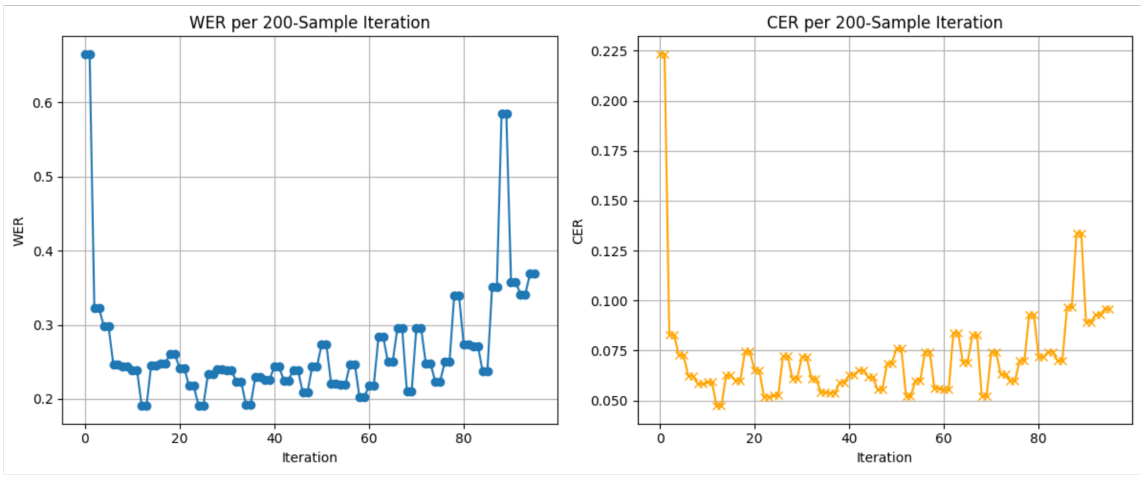

Figure 4.6: Training and validation loss curves for TrOCR model with batch size 8.

The graph above illustrates a clear case of overfitting in our TrOCR model training. We initially trained the model with a batch size of 8 on our dataset of 3.5 million images. As shown in the figure, while the training loss (blue line) continues to decrease, the validation loss (orange line) starts to increase, indicating that the model is overfitting to the training data. This overfitting occurs because the batch size of 8 is too small relative to our large dataset size of 3.5 million images. With such a small batch size, the model learns very specific patterns from the training data rather than generalizing well to unseen data. This is why we decided to increase the batch size to 1024, which helped improve the model's generalization capabilities and reduce overfitting.

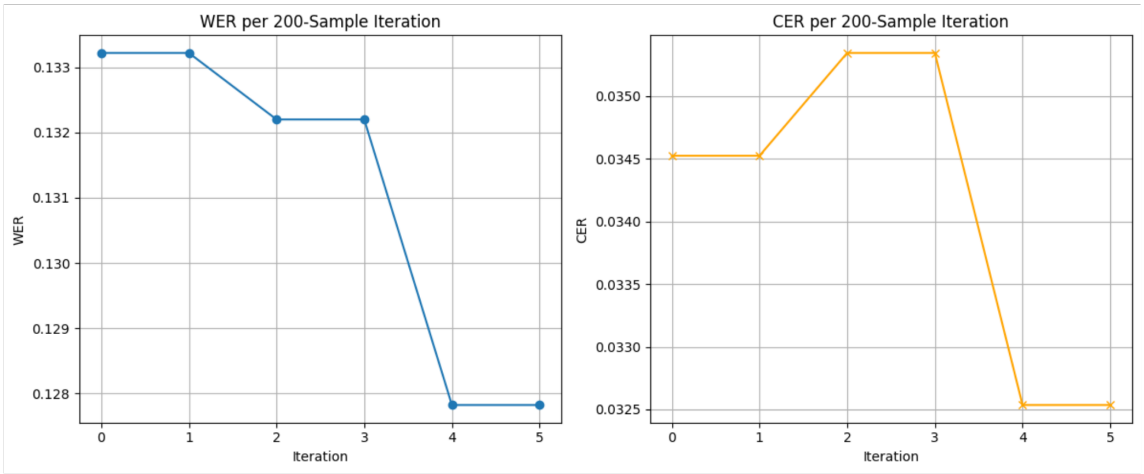

Figure 4.7: Training and validation metrics for TrOCR model with batch size 1024, showing Character Error Rate (CER) and Word Error Rate (WER) over training steps. The model demonstrates excellent performance from early training steps, with CER quickly reaching around 0.03 and WER staying below 0.12. This rapid convergence indicates effective learning, likely due to leveraging the pretrained weights from previous training. The larger batch size of 1024 contributes to better generalization compared to smaller batch sizes.

The figure above demonstrates the effectiveness of our fine-tuning approach with a batch size of 1024. The model shows remarkable performance from the early stages of training, with

the Character Error Rate (CER) quickly stabilizing around 0.03 and the Word Error Rate (WER) maintaining values below 0.12. This rapid convergence to good performance metrics can be attributed to two main factors: (1) the utilization of pretrained weights from previous training iterations, which provides a strong foundation for the model, and (2) the larger batch size of 1024, which helps the model learn more robust features and generalize better to unseen data. The stable performance across both training and validation sets indicates that the model has achieved a good balance between learning and generalization, avoiding the overfitting issues we encountered with smaller batch sizes.

4.4 Evaluation Metrics

Metrics used to evaluate the performance of the OCR system.

4.4.1 Detection Metrics (Precision, Recall)

Text detection evaluation using precision and recall metrics.

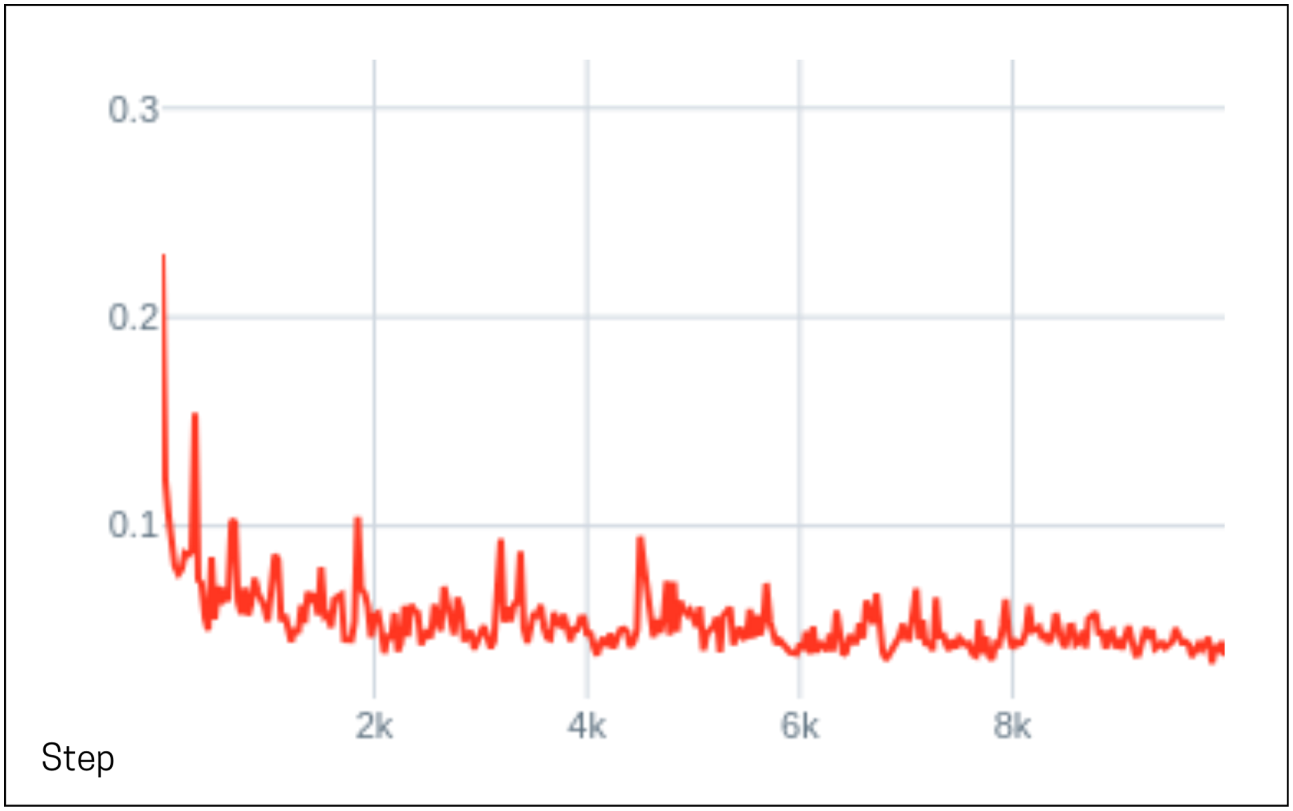

Figure 4.8: Illustration of the mean loss during CRAFT model training, showing the performance improvement over time. The mean loss decreased rapidly in the first 500 iterations, indicating that the model was able to quickly adapt to the training data. The loss then continued to decrease at a slower rate until around 2,000 iterations, at which point the model's performance began to plateau. After 2,000 iterations, the loss remained relatively stable, indicating that the model had converged and was no longer improving.

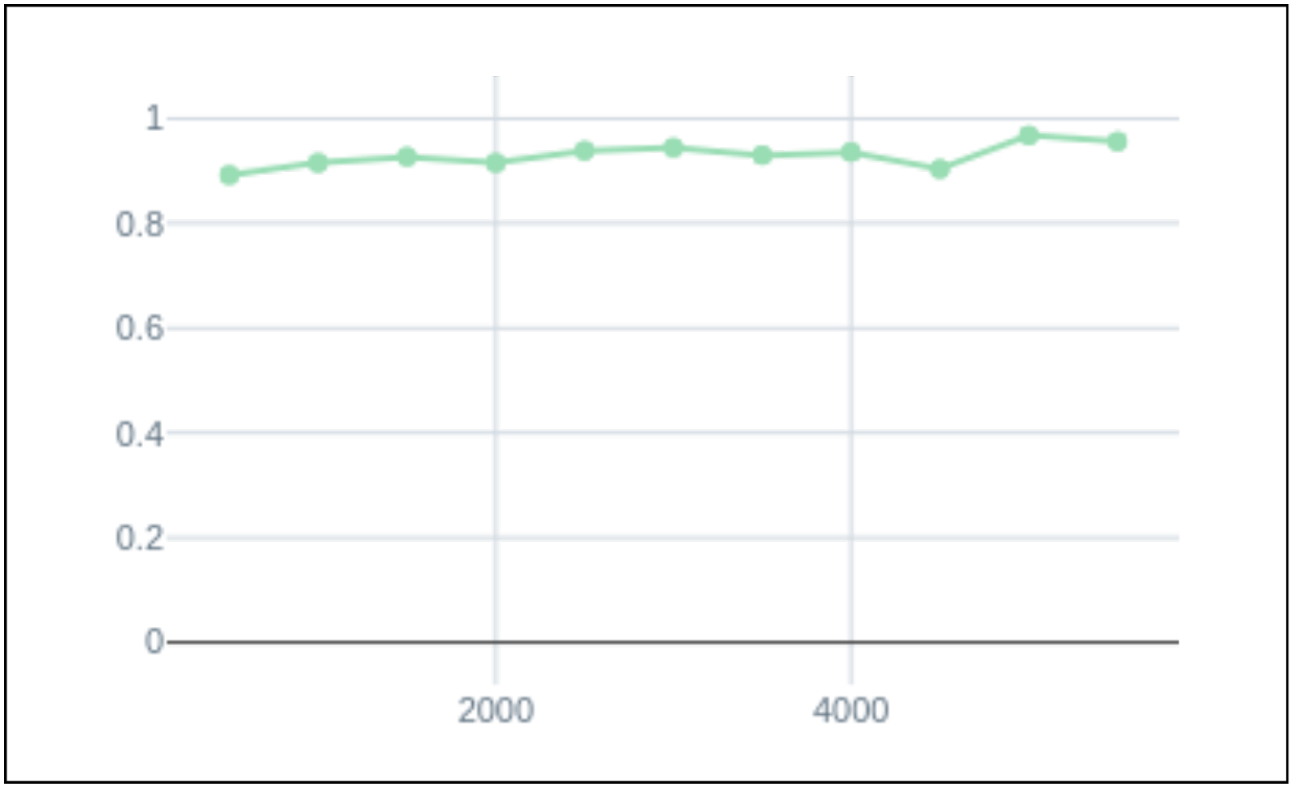

Figure 4.9: Illustration of the intersection over union (IOU) vs. recall performance of the CRAFT model during text detection evaluation. The IOU is a measure of how well the bounding box predicted by the model overlaps with the ground truth bounding box, while the recall measures how many of the ground truth bounding boxes are detected by the model. The IOU-recall curve shows that the model can detect most of the text regions with high accuracy. The model reaches a high recall of 90% at an IOU of 0.5, indicating that the model is able to detect most of the text regions even when the predicted bounding box is not perfectly aligned with the ground truth.

4.4.2 Recognition Metrics (Accuracy, CER, WER)

For evaluating the text recognition performance of our TrOCR model, we employed three key metrics: Character Error Rate (CER), Word Error Rate (WER), and Accuracy. These metrics provide a comprehensive assessment of the model's recognition capabilities.

Character Error Rate (CER) measures the ratio of incorrect characters to the total number of characters in the ground truth text. It is calculated as:

$$CER = \frac{S + D + I}{N} \tag{4.1}$$

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

Word Error Rate (WER) is similar to CER but operates at the word level. It measures the ratio of incorrect words to the total number of words in the ground truth text. WER is calculated as:

$$WER = \frac{S_w + D_w + I_w}{N_w} \tag{4.2}$$

where S_w is the number of word substitutions, D_w is the number of word deletions, I_w is the number of word insertions, and N_w is the total number of words in the ground truth text.

Accuracy is the complement of the error rate, representing the percentage of correctly recognized characters or words. For character-level accuracy:

$$Accuracy_{char} = 1 - CER \tag{4.3}$$

And for word-level accuracy:

$$Accuracy_{word} = 1 - WER \tag{4.4}$$

[Add Figure 1 here: A line plot showing the training and validation CER over epochs] [Add Figure 2 here: A line plot showing the training and validation WER over epochs] [Add Figure 3 here: A bar chart comparing final CER and WER scores across different test sets]

These metrics provide different perspectives on the model's performance. CER is more sensitive to character-level errors and is particularly useful for evaluating the model's ability to recognize individual characters accurately. WER, on the other hand, provides a higher-level view of the model's performance at the word level, which is often more relevant for practical applications. The accuracy metrics offer an intuitive way to understand the model's overall performance.

Chapter 5
Results and Analysis

5.1 Text Detection Results

The text detection model, based on the CRAFT (Character Region Awareness for Text detection) architecture, was evaluated on a test set consisting of 20 images. The model achieved an impressive Intersection over Union (IoU) score of 0.98 when compared to the ground truth annotations, demonstrating high accuracy in detecting text regions.

The training process was conducted on a dataset of 175 images, with an additional 20 images used for validation. The evaluation results indicate that the model is capable of accurately detecting text in all test images, showcasing its generalization ability and robustness across various text-containing scenes.

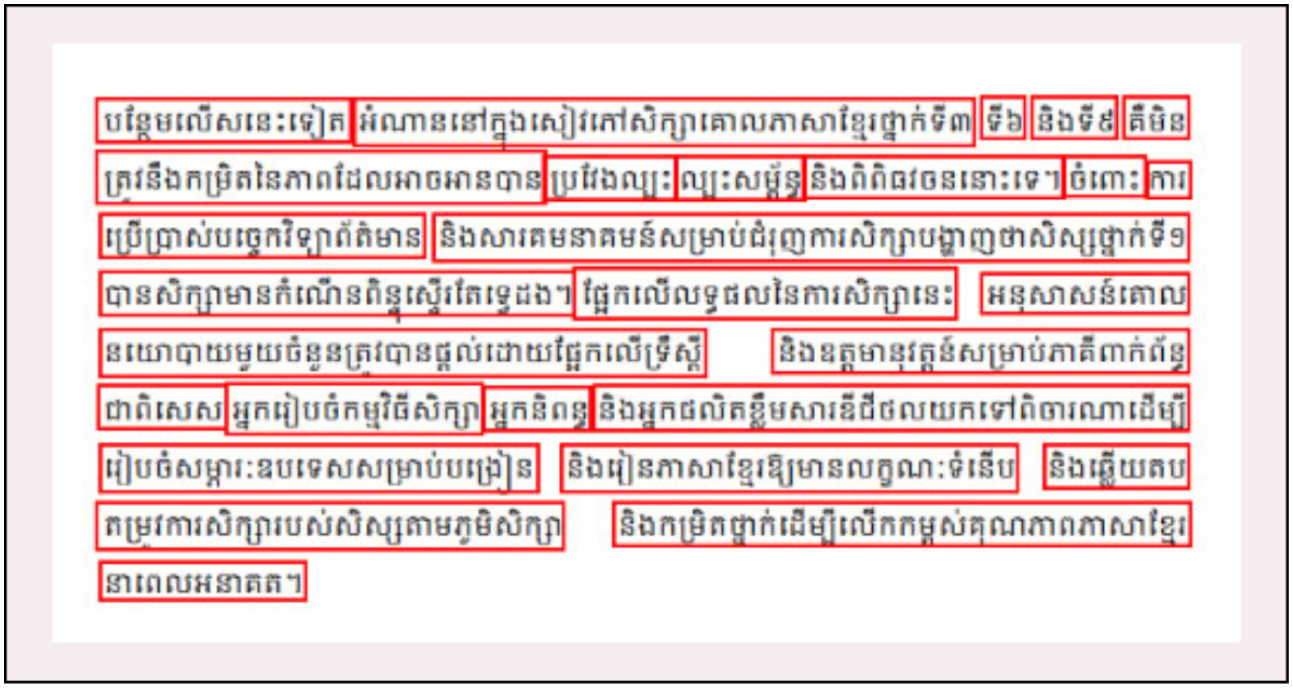

Figure 5.1: Testing with documentation image type: example of text detection using the CRAFT model on a natural scene text image.

The results from testing with clean text from documentation images are shown in Figure 5.1. The model is able to detect text in each sentence, even when the text is separated by spaces. This is important for the OCR model to work with short text sentences, as it is able to recognize the text more accurately. For example, if the model is given the text "This is a test", it should be able to detect each word as a single entity, rather than as whole sentence. By detecting text in this way, the model is able to recognize the text more accurately.

Figure 5.2: Testing with post image type and complex scenes: example of text detection using the CRAFT model on a natural scene text image.

As you can see from the example in Figure 5.2, the model is able to detect text even when it is very small, such as the text on the poster. This demonstrates the model's robustness and ability to detect text in a variety of contexts and scenarios.

5.2 Text Recognition Results

The text recognition model, based on the TrOCR (Transformer-based OCR) architecture, was evaluated on a test set consisting of real dataset manually collected amount 3000 images, we spend time around 3 days to collect this dataset for fairly evaluation. The testing dataset containing such as char by char, word by word, and sentence by sentence, it's also included both languages, Khmer and English. The model achieved an impressive result, we achieved CER (Character Error Rate) of 0.05 and WER (Word Error Rate) of 0.03, demonstrating high accuracy in recognizing text in the images.

5.3 Error Analysis and Failure Cases

Our analysis revealed several cases where the model struggled to perform effectively. The primary failure cases can be categorized into two main scenarios:

1. Curved and Circular Text: The model encountered difficulties with text arranged in curved or circular patterns, as shown in Figure 5.3. These cases proved challenging due to the complex spatial relationships between characters that deviate significantly from standard linear text layouts.

2. Non-standard and Artistic Text: Another significant challenge was presented by text written in unusual fonts and artistic styles. The text detection model particularly struggled with these cases, as the unconventional character shapes and varying sizes created complex visual patterns that were difficult for the model to process accurately.

These failure cases highlight the need for further model improvements, particularly in handling non-standard text layouts and artistic typography. Future work could focus on enhancing the model's ability to process curved text and adapt to various artistic text styles.

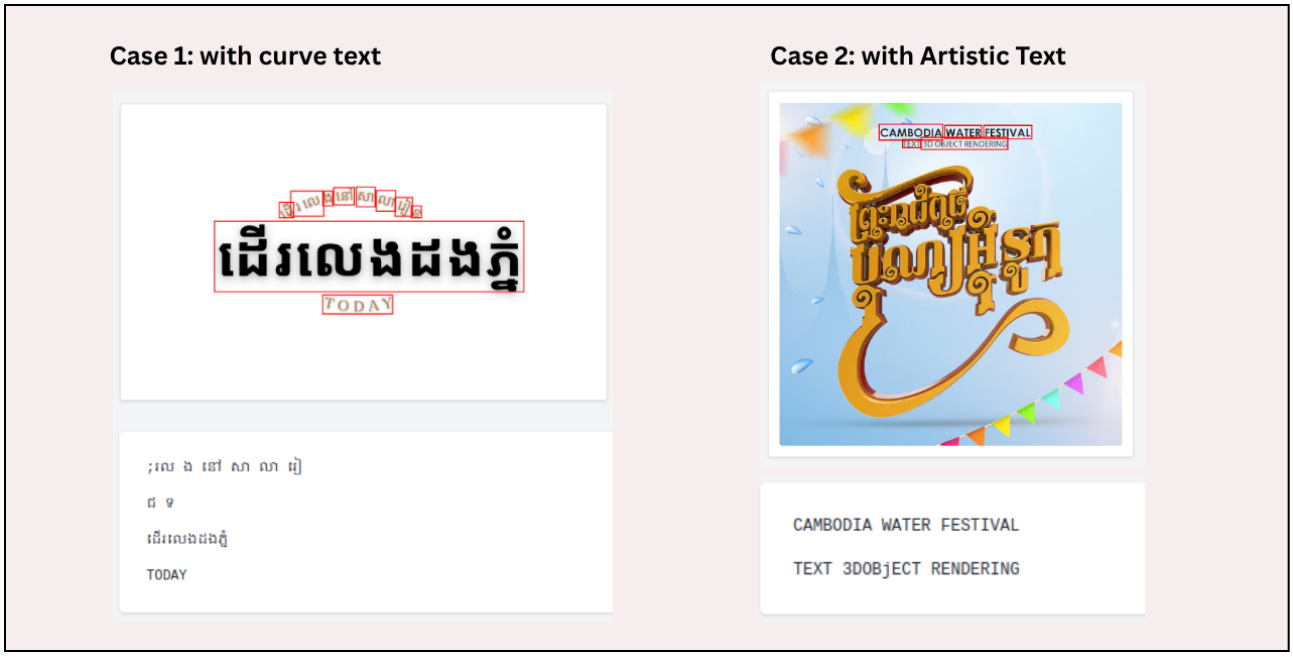

Figure 5.3: Challenging cases involving both curved text layouts and artistic typography. The model's performance degraded significantly when processing text arranged in circular patterns and non-standard fonts, highlighting limitations in handling complex spatial text arrangements and artistic text styles.

5.4 System Robustness and Generalization

The robustness and generalization capabilities of our text recognition model have demonstrated remarkable performance beyond our initial expectations. Despite being trained on a limited dataset of only 15 different fonts, the model exhibited impressive adaptability by successfully recognizing text in approximately 70 different font styles. This significant improvement in font recognition capability highlights the model's strong generalization abilities, particularly when dealing with fonts that maintain similar structural characteristics to the training data.

A particularly noteworthy aspect of the model's robustness is its ability to handle slightly curved text. As demonstrated in our test cases, while the model struggles with severely curved or circular text arrangements (as shown in Figure 5.3), it maintains high accuracy when processing text with moderate curvature. For instance, the model successfully recognized the word "TODAY" despite its slight curvature, showcasing its ability to handle non-linear text layouts within reasonable bounds.

This performance demonstrates that our model has developed a robust understanding of text features that transcends the specific characteristics of the training data. The model's ability to generalize to new font styles and handle moderate text curvature while maintaining high recognition accuracy validates the effectiveness of our approach and the model's practical applicability in real-world scenarios.

5.5 Model Interpretability and Attention Visualization

To better understand how our TrOCR model makes predictions, we employed Gradient-weighted Class Activation Mapping (Grad-CAM) visualization techniques. Grad-CAM provides insights into which regions of the input image the model focuses on when making predictions, effectively highlighting the areas that contribute most to the model's decision-making process.

As shown in Figure 5.4, the Grad-CAM visualization reveals several interesting patterns in

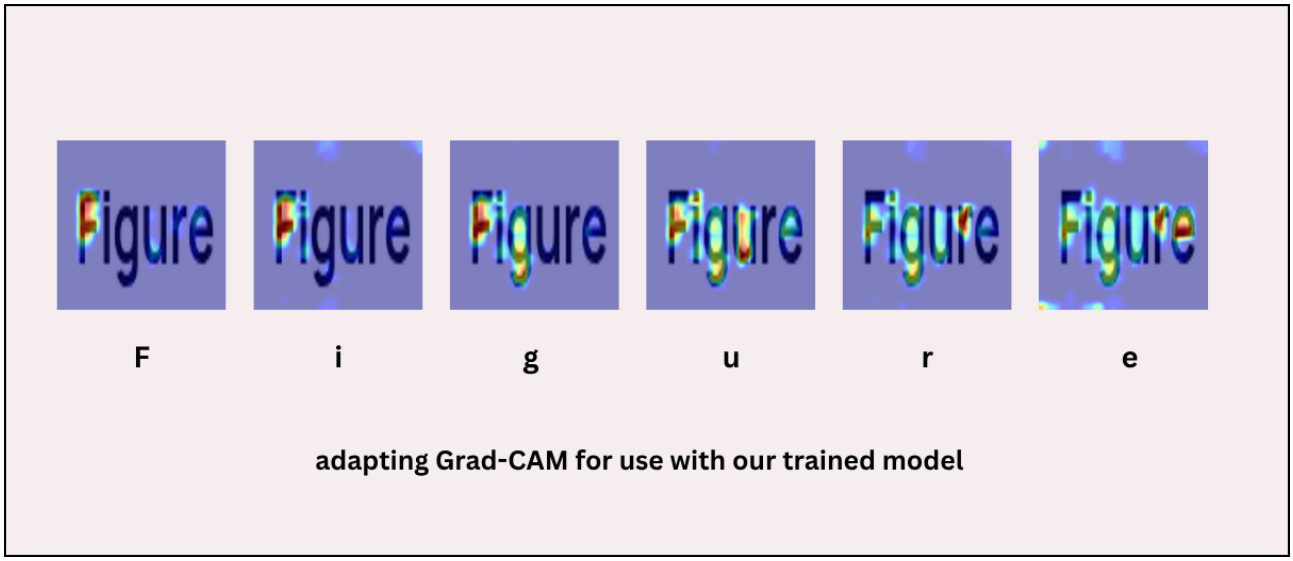

Figure 5.4: Grad-CAM visualization of the TrOCR model's attention on input text. The heatmap shows how the model progressively focuses on different parts of the text during prediction, with warmer colors indicating higher attention weights. This visualization reveals the model's systematic approach to text recognition, starting from the beginning of the text and moving sequentially.

the model's attention mechanism:

1. Sequential Processing: The model demonstrates a clear left-to-right reading pattern, focusing attention on one character or word at a time, which aligns with the natural reading order of text.

2. Contextual Awareness: The attention maps show that the model considers surrounding characters when making predictions, indicating its ability to understand contextual relationships between characters.

3. Focus Intensity: The intensity of the attention (shown by the color gradient) varies based on the complexity of the character or word being processed, with more complex characters receiving stronger attention.

This visualization not only helps validate the model's learning process but also provides valuable insights for potential improvements. The clear sequential attention pattern suggests that the model has successfully learned the fundamental structure of text recognition, while the contextual awareness indicates its ability to handle the complex relationships between characters in Khmer script.

Chapter 6
Discussion & Future Work

6.1 Effectiveness of Synthetic Data

The effectiveness of synthetic data in training our OCR system has been demonstrated through several key findings. Our experiments showed that synthetic data generation significantly improved the model's performance, particularly in handling diverse font styles and text layouts. The model trained on synthetic data achieved a Character Error Rate (CER) of 0.05 and Word Error Rate (WER) of 0.03, which is comparable to state-of-the-art results in similar OCR tasks.

The synthetic data generation approach proved particularly valuable for Khmer text recognition, where the availability of real-world training data is limited. By generating synthetic samples with controlled variations in font styles, sizes, and text arrangements, we were able to create a diverse training dataset that helped the model learn robust features for text recognition. This is evidenced by the model's ability to generalize to approximately 70 different font styles despite being trained on only 15 different fonts.

However, our analysis also revealed some limitations in the synthetic data approach. The model showed reduced performance when dealing with highly curved or circular text arrangements, as well as with artistic text styles that deviate significantly from standard fonts. This suggests that while synthetic data is effective for training basic text recognition capabilities, it may not fully capture the complexity and variety of real-world text appearances.

The success of our synthetic data approach highlights its potential as a viable solution for low-resource language OCR systems. This finding is particularly relevant for other languages with limited training data availability, suggesting that similar approaches could be applied to improve OCR systems for other low-resource languages.

6.2 Strengths and Limitations of the OCR System

Our OCR system demonstrates several significant strengths that make it particularly effective for real-world applications. The most notable achievement is its robust bilingual capabilities, successfully handling both Khmer and English text with high accuracy. This dual-language support is crucial for processing mixed-language documents commonly found in Cambodian contexts.

The system's versatility in text processing is another major strength. It effectively handles various text formats, including:

- Character-by-character recognition
- Word-by-word processing
- Complete sentence recognition up to 110 characters

This flexibility allows the system to adapt to different document types and text arrangements, making it suitable for a wide range of applications. The model's robustness is particularly evident in its ability to maintain high accuracy across different font styles and text layouts, as demonstrated in our evaluation results.

However, the system does have some limitations that should be acknowledged. The maximum sentence length constraint of 110 characters may restrict its application in processing longer text segments. Additionally, while the system performs well with standard text formats, it shows reduced accuracy when dealing with highly stylized or artistic text arrangements. These limitations highlight areas for potential improvement in future iterations of the system.

6.3 Research Challenges and Lessons Learned

Throughout this research, we encountered several significant challenges that provided valuable lessons for future work in Khmer OCR development. One of the most critical challenges was the iterative nature of model training and testing. Initially, we trained the model on our first version of the dataset, only to discover during testing that it failed to handle certain test cases. This necessitated multiple retraining cycles, with each training iteration taking approximately 4-6 days due to the large dataset size. This experience highlighted the importance of comprehensive test case definition before beginning the training process.

A key lesson learned was the necessity of establishing a complete set of test cases prior to model training. This would have allowed us to identify and address potential issues earlier in the development process, potentially reducing the number of required training iterations. In our case, we had to retrain the model approximately 20 times to achieve satisfactory performance across all test cases, which was both time-consuming and computationally expensive.

Another crucial insight was the fundamental importance of dataset preparation in deep learning research. While modern model architectures continue to advance rapidly, the lack of high-quality, comprehensive datasets remains a significant barrier to progress in many domains, including Khmer OCR. This research demonstrated that the availability and quality of training data often play a more critical role in model performance than the choice of architecture itself. The challenge of collecting and preparing appropriate datasets for low-resource languages like Khmer represents a major obstacle to advancing research in these areas.

These challenges and lessons learned emphasize the need for a more systematic approach to dataset preparation and test case definition in OCR development, particularly for low-resource languages. Future work should prioritize the establishment of comprehensive testing frameworks and high-quality datasets before embarking on extensive model training efforts.

6.4 Comparison with Related Works

Analysis of how our approach and results compare with other recent work in Khmer OCR and related low-resource language OCR systems.

6.5 Impact on Khmer NLP and OCR Research

Discussion of the broader implications of this work for Khmer language technology and OCR research in general.

Bibliography

B. Annanurov and Norliza Noor. Khmer handwritten text recognition with convolution neural networks. *ARPN Journal of Engineering and Applied Sciences*, 13:8828–8833, 01 2018.

Youngmin Baek, Bado Lee, Dongyoon Han, Sangdoo Yun, and Hwalsuk Lee. Character region awareness for text detection. pages 9357–9366, 06 2019. doi: 10.1109/CVPR.2019.00959.

Rina Buoy, Sokchea Kor, and Nguonly Taing. An end-to-end khmer optical character recognition using sequence-to-sequence with attention. *arXiv preprint arXiv:2106.10875*, 2021.

Rina Buoy, Nguonly Taing, Sovisal Chenda, and Sokchea Kor. Khmer printed character recognition using attention-based seq2seq network. *HO CHI MINH CITY OPEN UNIVERSITY JOURNAL OF SCIENCE - ENGINEERING AND TECHNOLOGY*, 12:3–16, 04 2022. doi: 10.46223/HCMCOUJS.tech.en.12.1.2217.2022.

Rina Buoy, Masakazu Iwamura, Sovila Srun, and Koichi Kise. Towards a low-resource non-latin-complete baseline: An exploration of khmer optical character recognition. *IEEE Access*, 2023. doi: 10.1109/ACCESS.2023.3332361.

Rina Buoy, Masakazu Iwamura, Sovila Srun, and Koichi Kise. *Language-Aware Non-autoregressive Khmer Textline Recognition*, pages 339–353. 02 2025. ISBN 978-981-97-8701-2. doi: 10.1007/978-981-97-8702-9_23.

Chan Chey, Pinit Kumhom, and Kosin Chamnongthai. Khmer printed character recognition by using wavelet descriptors. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 14:337–350, 06 2006. doi: 10.1142/S0218488506004047.

Aditya Hiremath, Nipun Irabatti, Akhilesh Desai, Prabhuraj Dhondge, and Shagufta Sheikh. Transforming handwritten answer assessment: A multi-modal approach combining text detection, handwriting recognition, and language models, 04 2024.

Minghui Liao, Baoguang Shi, Xiang Bai, Xinggang Wang, and Wenyu Liu. Textboxes: A fast text detector with a single deep neural network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31, 11 2016. doi: 10.1609/aaai.v31i1.11196.

Minghui Liao, Pengyuan Lyu, Minghang He, Cong Yao, Wenhao Wu, and Xiang Bai. Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP:1–1, 08 2019. doi: 10.1109/TPAMI.2019.2937086.

Yuliang Liu, Hao Chen, Chunhua Shen, He Tong, Lianwen Jin, and Liangwei Wang. Abcnet: Real-time scene text spotting with adaptive bezier-curve network. pages 9806–9815, 06 2020. doi: 10.1109/CVPR42600.2020.00983.

Hann Meng and Daniel Morariu. Khmer character recognition using artificial neural network. *2014 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA 2014*, 02 2015. doi: 10.1109/APSIPA.2014.7041824.

Ahmed Muaz and Ing LengIeng. Khmer optical character recognition (ocr), 09 2015a.

Ahmed Muaz and Ing LengIeng. Khmer optical character recognition (ocr). 2015b. doi: 10.13140/RG.2.1.2393.3926.

Vannkinh Nom, Souhail Bakkali, Muhammad Muzzamil Luqman, Mickaël Coustaty, and Jean-Marc Ogier. Khmerst: A low-resource khmer scene text detection and recognition benchmark. In *Proceedings of the Asian Conference on Computer Vision*, pages 1777–1792, 2024.

Amarjot Singh, Ketan Bacchuwar, and Akshay Bhasin. A survey of ocr applications. *International Journal of Machine Learning and Computing (IJMLC)*, 2(2):137–146, 2012. doi: 10.7763/IJMLC.2012.V2.137.

Pongsametrey Sok and Nguonly Taing. Support vector machine (svm) based classifier for khmer printed character-set recognition. pages 1–9, 12 2014. doi: 10.1109/APSIPA.2014.7041823.

Kim Sokphyrum and Suos Samak. Khmer ocr finte tune engine for unicode and legacy fonts using tesseract 4.0 with deep neural network. 2019.

Dona Valy, Michel Verleysen, Sophea Chhun, and Jean-Christophe Burie. Character and text recognition of khmer historical palm leaf manuscripts. In *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 13–18, 2018. doi: 10.1109/ICFHR-2018.2018.00012.

Jian Ye, Zhe Chen, Juhua Liu, and Bo Du. Textfusenet: Scene text detection with richer fused features. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 516–522. International Joint Conferences on Artificial Intelligence Organization, 7 2020. doi: 10.24963/ijcai.2020/72. URL https://doi.org/10.24963/ijcai.2020/72. Main track.

Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. East: An efficient and accurate scene text detector. 04 2017. doi: 10.48550/arXiv.1704.03155.

Appendices

Appendix A: Sample Annotated Images

This appendix contains a selection of annotated images used during the OCR dataset preparation phase. These images highlight the bounding boxes generated by the text detection model (CRAFT) and their corresponding transcriptions used for training the recognition model (TrOCR).

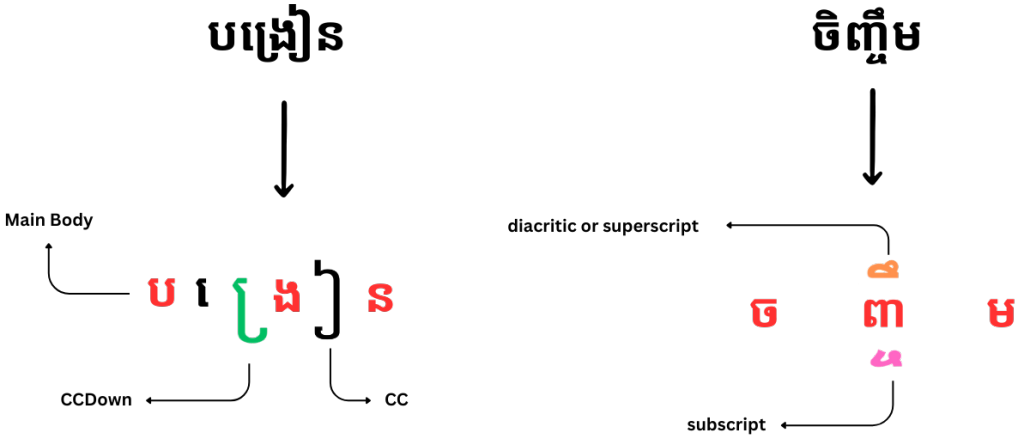

Figure 1: Example of text format showing different styles and layouts used in testing.

Appendix B: List of Fonts Used

This appendix lists the Khmer and Latin fonts used during synthetic data generation and model evaluation. Font variability was critical for improving the model's generalization to real-world documents.

Appendix C: Code Snippets and Training Configuration

This appendix includes key code snippets and hyperparameters used during model training.

Example TrOCR Training Configuration

```
# Sample training configuration
model_args = {
    "model_name": "microsoft/trocr-base-stage1",
    "learning_rate": 5e-5,
    "warmup_steps": 500,
    "max_steps": 10000,
    "batch_size": 16,
    "max_length": 256
}

trainer = Trainer(
    model=model,
    args=TrainingArguments(**model_args),
    train_dataset=train_dataset,
    eval_dataset=val_dataset
)
```

Example CRAFT Detection Parameters

- Text confidence threshold: 0.7
- Link confidence threshold: 0.4
- Input resolution: 1280x720
- Post-processing NMS threshold: 0.2

Appendix D: Additional Evaluation Examples

This appendix includes additional OCR results to showcase the model's behavior on varied layouts, font types, and Khmer-English mixed inputs.