Facilitating Access to Historical Documents by Improving Digitisation Results

Thi Tuyet Hai Nguyen

LA ROCHELLE UNIVERSITÉ

École doctorale Euclide

Laboratoire Informatique, Image, Interaction (L3i)

THÈSE présentée par :

Thi Tuyet Hai NGUYEN

soutenance le : **6 avril 2020**

pour obtenir le grade de : **Docteur de La Rochelle Université**

Discipline : **Informatique**

Facilitating Access to Historical Documents by Improving Digitisation Results

---

JURY :

Jean-Christophe BURIE	Professeur, La Rochelle Université, Président
Frédéric BÉCHET	Professeur, Aix Marseille Université, Rapporteur
Laurent HEUTTE	Professeur, Université de Rouen Normandie, Rapporteur
Gaël DIAS	Professeur, Université de Caen Normandie, Examinateur
Laurence LIKFORMAN-SULEM	Maîtresse de Conférence HDR, ParisTech, Université de Paris-Saclay, Examinatrice
Sébastien CRETIN	Digitisation expert, Bibliothèque nationale de France, Examinateur
Antoine DOUCET	Professeur, La Rochelle Université, Directeur de thèse
Adam JATOWT	Associate Professor, Kyoto University, Co-Directeur de thèse

Abstract

Born-analog documents contain enormous knowledge which is valuable to our society. For the purpose of preservation and easy accessibility, several digitisation projects have converted these documents into digital texts by using optical character recognition (OCR) software. Some existing problems of OCR techniques prevent users and further processes from accessing, searching, or retrieving information on these digitised collections, and so limit the benefits of these above projects.

A notable limitation is the fact that certain meaningful structures such as chapters, sections, etc., are not available from OCRed books. Thus, it is not convenient for users to navigate or search information inside books. Another constraint is that the accuracy of modern OCR engines on historical documents substantially decreases. Erroneous OCR output considerably impacts on the performance of search engines and natural language processing systems. This thesis facilitates access to historical digitised documents by addressing such problems.

Several approaches are proposed within this thesis, aiming to reconstruct the logical book structures and to improve the quality of digitised text.

The first contribution is to rebuild the logical book structures. An ensemble method is introduced to extract tables of contents of digitised books. Experimental results show that our approach outperforms the state-of-the-art for both evaluation metrics.

The major contribution of this thesis is to provide methodologies to reduce OCR errors. Common and different features between OCR errors and human misspellings are clarified for better designing post-OCR processing. Normally, a post-processing system detects and corrects remaining errors. However, it is reasonable to treat them separately in some applications which allow to filter out, flag, or selectively reprocess such data. In this thesis, we examine different post-OCR approaches, ones based on error model and language model, and others that involve neural network models. Results reveal that the performance of our proposals is comparable to several strong baselines on English datasets of the two competitions on post-OCR text correction organised in the International Conference on Document Analysis and Recognition in 2017 and 2019.

Keywords: table of contents extraction, book structure extraction, post-OCR processing, post-OCR error correction, post-OCR error detection.

Résumé

Les documents papiers sont à la base de nos connaissances et renferment une myriade d'information dont certaines sont très précieuses pour notre société. Dans un but de préservation et afin de les rendre plus accessibles, de nombreux projets de numérisation visent à convertir ce type de documents en textes numérisés, notamment en utilisant des logiciels de reconnaissance optique de caractères (OCR). Toutefois, certains problèmes inhérents aux techniques actuelles d'OCR rendent difficiles la recherche ou l'accès aux informations présentes dans ces collections numérisées, tant pour les utilisateurs que pour les processus automatiques, et limitent ainsi l'impact de ces efforts de numérisation.

L'une des limitations de la numérisation repose sur le processus même puisque les documents numérisés ne sont pas immédiatement réprésentés sous leur forme logique (partie, chapitre, section, etc.), mais de façon physique. Ainsi, une œuvre sera numérisée page par page, ce qui ne correspondant généralement qu'à une organisation physique et pas à l'intention rédactionnelle des auteurs. La structure logique des documents doit ainsi être extraite afin de permettre aux utilisateurs de naviguer dans les collections ou même de trouver des informations au sein d'un ouvrage.

Un second verrou du processus de numérisation, qui en est également le plus important, correspond aux performances des moteurs d'OCR. En effet, celles-ci sont substantiellement réduites pour les documents patrimoniaux qui ont généralement subis des dégradations. Les erreurs d'OCR que cela induit ont un impact non négligeable sur la performance des outils de recherches et sur les systèmes de traitement du langage naturel puisqu'il faut par exemple apparier des besoins bien écrits à des textes mal reconnus. Cette thèse a pour objectif de faciliter l'accès aux documents historiques numérisés en étudiant les problèmes précédemment mentionnés.

En vue de faciliter l'accès aux documents historiques, plusieurs approches sont proposées, visant à reconstruire les structures logiques des ouvrages et à améliorer la qualité des textes numérisés par OCR.

En ce qui concerne l'extraction de la structure logique, nous avons développé des approches de fusion combinant des méthodes préexistantes afin d'extraire la table des matières d'ouvrages numérisés. Nos expériences ont démontré que cette approche surpasse

l'état de l'art. La contribution majeure de cette thèse fournit, quant à elle, des méthodes pour la détection et la correction des erreurs d'OCR. Les caractéristiques communes et divergentes entre les erreurs d'OCR et celles des utilisateurs sont clarifiées pour mieux concevoir les traitements post-OCR. Normalement, un système de post-traitement détecte et rectifie les erreurs résiduelles. Toutefois, il peut être préférable de gérer ces erreurs séparément grâce à des applications qui permettent de filtrer, d'étiqueter, ou de traiter sélectivement de telles données. Dans cette étude, nous examinons différentes approches post-OCR basées sur la modélisation des erreurs typiques observées, et sur des modèles de réseaux de neurones. Les résultats montrent que les performances de nos méthodes sont comparables à plusieurs méthodes de référence sur des jeux de données en anglais utilisés lors des deux premières éditions de la compétition sur la correction des textes post-OCR organisée durant les conférence ICDAR en 2017 et 2019.

Mots-clés: extraction de la table des matières, extraction de la structure d'un livre, traitements post-OCR, correction des erreurs post-OCR, détection des erreurs post-OCR.

Acknowledgement

First of all, I would like to express my sincere gratitude to my supervisors: Prof. Antoine Doucet, Assoc. Prof. Adam Jatowt, Assoc. Prof. Mickaël Coustaty, and Dr. Nhu-Van Nguyen. They create an open research environment and kindly advise me the way to analyse and solve a problem. Lots of constructive comments and interesting questions are discussed to help me better understand the issue as well as find relevant solutions.

It is my pleasure to thank all members of my PhD defense's committee: Prof. Jean-Christophe Burie, Prof. Frédéric Béchet, Prof. Laurent Heutte, Prof. Gaël Dias, Assoc. Prof. Laurence Likforman-Sulem, and Sébastien Cretin. I gratefully acknowledge the thorough questions from two reviewers: Prof. Frédéric Béchet, Prof. Laurent Heutte.

I am delighted to show the appreciation to all my colleagues, especially my office-mates: Quoc-Bao Dang, Thanh-Nam Le, Made Windu Antara Kesiman, Florian Lardeux, and Lady Viviana Beltrán Beltrán. We have many interesting discussions and pleasant time together over last three years. Furthermore, I am impossible to totally focus on my study without the help of our laboratory administrative staffs: Kathy Theuil, Erlandri Chaigneaud, Muhammad Muzzamil Luquan, Dominique Limousin, and Dominique Besse (Responsable du Pôle Audiovisuel).

I cannot forget all my Vietnamese friends in La Rochelle: Quoc-Bao Dang, Huynh-Le Tran, Thanh-Khoa Nguyen, Vinh-Loc Cu, Anh Thu Phan Ho, Hoang Nam Ho, My-Anh Tran, Nhu-Hoa Nguyen, Thanh-Nam Le, Nhu-Van Nguyen, Thanh Hien Nguyen, and Vietnamese-like Noudéhouénou Lionel Jaderne Houssou. Many thanks for all your helps as well as sweet memories together.

My special gratitude is to Project 911 of Vietnamese Ministry of Education and Training for the financial support. I am pleased to thank Posts and Telecommunications Institute of Technology for giving me a great opportunity to study in France.

Finally, I would like to deeply appreciate my family, especially my parents, for their love and continuous supports. Last but not least, this thesis will be devoted to my husband. Without his encouragement, I will not come here and have unforgettable experiences.

Contents

Abbreviations

2-pass RNN	2-pass Recurrent Neural Network, the name of the competition team
AMELIOCR	The research project of National Library of France
BERT	Bidirectional Encoder Representations from Transformers
BFS	Breadth First Search
BL	British Library
BnF	National Library of France
CAPTCHA	Completely Automated Public Turing test to tell Computers and Humans Apart
CCC	Context-based Character Correction, the name of the competition team
CER	Character Error Rate
Char-SMT/NMT	Character Statistical Machine Translation/ Machine Translation, the name of the competition team
CLAM	Character Level Attention Model, the name of the competition team
CLAWS	Constituent Likelihood Automatic Word-tagging System
COHA	Corpus of Historical American English
CoVe	Context Vectors
CSIITJ	The name of the competition team
EFP	Error Frequency Patterns, the name of the competition team
GAN	Generative Adversarial Nets
GT	Ground Truth
ICDAR	International Conference on Document Analysis and Recognition
ICDAR2017	International Conference on Document Analysis and Recognition organised in 2017
ICDAR2019	International Conference on Document Analysis and Recognition organised in 2019
IMPACT	Digitisation project
LCS	Longest Common Subsequence
LSTM	Long Short Term Memory
MCLCS	Maximal Consecutive Longest Common Subsequence
MDCS	Microsoft Development Center Serbia, the name of the competition team
MLM	Masked Language Model
MMDT	Multi-Modular Domain-Tailored, the name of the competition team

MT	Machine Translation
NER	Named Entity Recognition
NLCS	Normalised Longest Common Subsequence
NLP	Natural Language Processing
NMCLCS	Normalised Maximal Consecutive Longest Common Subsequence
NMT	Neural Machine Translation
NSP	Next Sentence Prediction
OCR	Optical Character Recognition
OOV	Out-Of-Vocabulary
OpenNMT	Open-source toolkit for Neural Machine Translation
OverLC	Overproof Library of Congress
OverNLA	Overproof National Library of Australia
PDF	Portable Document Format
POS	Parts-Of-Speech
RAE	Real Academia Espanola, the name of the competition team
RNN	Recurrent Neural Network
RNN-LM	Recurrent Neural Network based Language Model
SeqGAN	Sequence Generative Adversarial Nets
SLM	Statistical Language Model
SMT	Statistical Machine Translation
ToC	Table of Contents
TPU	Tensor Processing Unit
UVA	The name of the competition team
WER	Word Error Rate
WFST	Weighted Finite-State Transducer
WFST-PostOCR	The name of the competition team
XML	eXtensible Markup Language
XRCE	Xerox Research Centre Europe, the name of the competition team

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Background

Historical documents contain valuable knowledge that gets enormous attention from researchers and libraries around the world. Substantial efforts have been devoted to transform such analog material into electronic text, aiming at better preservation and easier access by a much wider audience. This transformation is known as digitisation [79].

Some notable digitisation projects are Europeana[1], Project Gutenberg[2], Wikisource[3], Google Books[4], and so on. Europeana serves as a portal which allows to access digitised corpus of many European museums, galleries, libraries and archives. Project Gutenberg and Wikisource are non-profit crowd-sourcing projects which create freely available digital documents. Google Books is a massive commercial digitisation project that intends for an online searchable catalog of books.

The digitising process involves the efficient scanning or photographing of documents,

[1] http://europeana.eu/ (accessed 2019-10-30)
[2] http://www.gutenberg.org/ (accessed 2019-10-30)
[3] http://wikisource.org/ (accessed 2019-10-30)
[4] https://support.google.com/books/partner/faq/3396243 (accessed 2019-10-30)

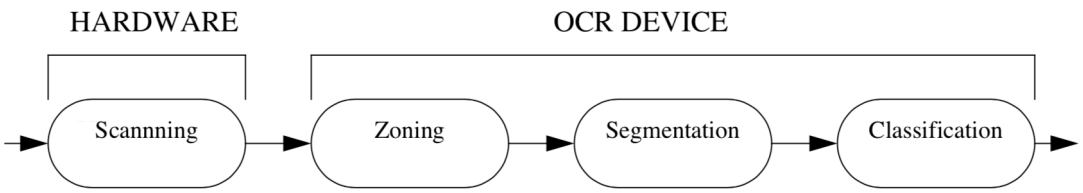


Figure 1.1: Standard steps of OCR procedure [95].

page-by-page, and the conversion of the image of each page into computer-readable text. The selection of digitisation techniques relies on several factors, such as the medium, the type of writing, the language, etc. In any case, the first step is to create digital images of documents by using scanners or digital cameras. When digital objects are available, two common approaches are manual text entry, and OCR software.

Manual text entry typically requires at least two typists to enter text into computers (known as double-keying). The typescripts are compared to highlight mismatches, next, a proofreader makes a choice to correct the transcription. Since double-keying and proofing are very labor-intensive, keying entry is often outsourced to service providers in countries with lower wages than in Europe or America, known as offshore double-keying. However, offshore outsourcing is still expensive and associated with some security issues when sharing information to the third party.

OCR software is a cost-efficient alternative of manual text entry without any security problems. This technique yields good recognition rates on modern documents, and has become one of the widely used and effective methods for the automatic conversion of printed text. Starting from images, generally, the OCR system detects and orders positions and types of all important zones in documents. Zones, then, are segmented into words which are, in turn, divided into characters. The last process classifies them into their corresponding character codes. Figure 1.1 shows the standard conversion steps of OCR procedure.

In principle, once these digitised documents are available online, multiple users can easily search, select, and make use of them simultaneously from various locations at any time. Indeed, people get their information by simply entering keywords into search engines, then the computer system will provide them a list of relevant results. If returned outputs

are books, readers may expect to apply the similar behaviour of reading digital-born books by looking at hyperlinked tables of contents and navigating to their selective pages.

Nevertheless, the digitised historical documents are not error-free due to various reasons. While OCR tools are designed to process text with standard spellings and modern typefaces, spellings of historical text and typefaces in past prints may differ from the current ones. For instance, take a look at spelling variation between a text printed in 1490 and its modernised edition in 2000, 'After dyuerse werkes made translated and achieued hauyng noo werke in hande.' vs. 'After diverse works made, translated and achieved, having no work in hand' [83]. Figure 1.2 illustrates the similarity of some characters that can also pose challenges to the human eye.

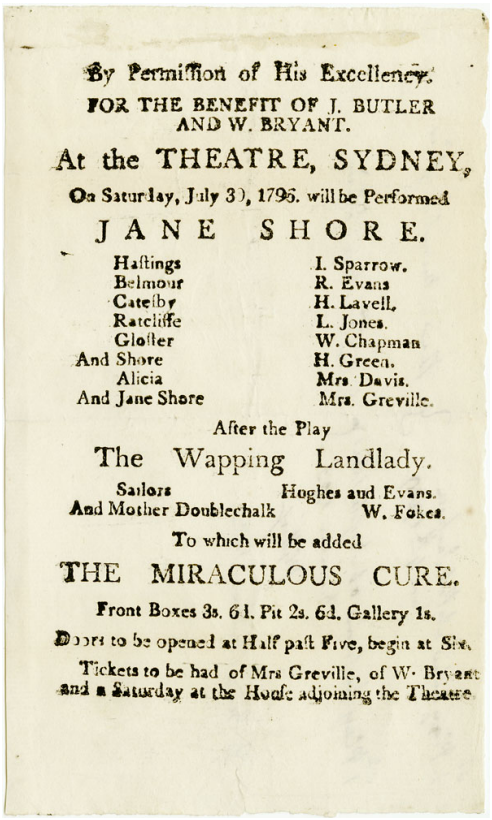
In addition, the physical quality of the original materials, the complicated layouts, etc., also have bad effects on performance of current OCR software. The more highly degraded the input, the higher the error rate. An example of the 18th century printed documents and its corresponding noisy OCRed text are respectively illustrated in Figures 1.3 and 1.4.

Figure 1.2: Gothic characters are frequently confused by OCR system. From left to right, the characters are: s (in its long form), f, u, n, u or n, B, V, R, N [29].

Although OCR engines have been constantly improved, they still lack adequate training data of past documents that is their strict requirement to achieve similar performance on historical text. Their parameters should be adapted to each kind of document, which is not feasible when processing the large number of pages. Furthermore, previously digitised resources processed with outdated OCR software are rarely resent to the state-of-the-art digitisation pipeline, as priority is often given to the masses of newly incoming documents. As a result, digitised historical collections still contain errors.

It is obvious that the noise induced by OCR technologies presents a serious challenge to downstream processes that attempt to make use of such data. Digital documents are

[4] http://pergamentai.mch.mii.lt/DokPranc/indexen.en.htm (accessed on 2019-10-30)



By Permiſſion of His Excellency.
FOR THE BENEFIT OF J. BUTLER
AND W. BRYANT.
At the THEATRE, SYDNEY,
On Saturday, July 3), 1796. will be Performed
JANE SHORE.

Haſtings	I. Sparrow.
Belmour	R. Evans
Cateſby	H. Lavell.
Ratcliffe	L. Jones.
Glofter	W. Chapman
And Shore	H. Green.
Alicia	Mrs Davis.
And Jane Shore	Mrs. Greville.

After the Play
The Wapping Landlady.

Sailors	Hoghes and Evans.
And Mother Doublechalk	W. Fokes.

To which will be added
THE MIRACULOUS CURE.
Front Boxes 3s. 6 l. Pit 2s. 6d. Gallery 1s.
Doors to be opened at Half paſt Five, begin at Six.
Tickets to be had of Mrs Greville, of W. Bryant
and a Saturday at the Houſe adjoining the Theatre.

Figure 1.3: The oldest known document printed in Australia, a theatre playbill 1796 [46].

ply Permifl'ion 0? His Excellenc};
FOR THE BENEFIT 0? J. BUTLER
AND W. BRYANT.
At the THEATR E, SYDNEY,
0n Snurhy. July 3), 1795, will be Performed
JANE SHORE.
' Hafiings I. Sparrow.
Balmour R. Evin:
Cam'bv HA LavelL
Ruciifl'e L. Janet,
Glofler W. Cthmln
And Share H. Green.
Alicia Mn 'D-vil.
And Jane Shore Mu. Gravillc.
Afierlhe Play
The Wapping Landlady.
' Salon Hughes and Evans.
And Modlcr Doublcchalk W. Faku.
To whnch will be added
THE MIRACULOUS CURE.
From Bonus 35. 6 1. Pix 2:. 641. Gillcry It.
Dun to I): opmci I! Hnlfpzft Yule, begin I: Sim
Tickcu m be had of Mrs Gum-vine, of '1- Br} an:
and I hunt-dag 3' 1h: Hoof: whining zbc Thaw

Figure 1.4: OCRed text of the theatre playbill in Figure 1.3.

indexed through their error-prone OCRed version, thus, computing systems may miss some relevant documents in their responses to user queries. Chiron *et al.* [9] estimated the impact of OCR errors on the use of the Gallica digital library from the National Library of France. They indicated that 7% of common search terms, which are queried at least 35 times, are potentially affected by OCR errors. Information retrieval performance remains good for relatively high error rates on long texts [96], but it drops dramatically [13, 70] on short text.

The impact of noisy OCRed text on other natural language processing (NLP) applications has been studied. Performance of named entity recognition tool, which extracts named persons, named locations, named organisations, etc., from text, considerably degrades with the rises of the error rate of OCR output [35, 69]. Text summarisation, which creates a summary representing the most important content of the original, suffers significant degradation on noisy input even with slight increases in the noise level of a document [41]. OCR errors cause negative influences on topic modeling, which discovers the abstract topic occurring in a collection of documents [72], and sentiment analysis if sentiment bearing words are not recognised well.

Digitised documents are not only noisy but also unstructured. Actually, OCR technologies typically produce the full text of digitised books with only the physical structures, such as pages, paragraphs, lines, and words. The absence of logical structures (e.g., chapters, sections) makes readers impossible to navigate to their desirable sub-parts by simply clicking on the corresponding part of the hyperlinked ToC.

Furthermore, another issue of missing logical structure is that users cannot take an advantage from structured information retrieval, which is proved to increase retrieval performance [99], to directly access to relevant parts of their information need in digital libraries. Structured information retrieval is defined as the search over structure documents [5] where a markup language like eXtensible Markup Language (XML) is used to identify document parts along with their meanings.

As a consequence, it is critically essential to improve the existing digitised historical collections for facilitating access to past documents and for further automatic processing.

[5] https://nlp.stanford.edu/IR-book/html/htmledition/xml-retrieval-1.html (accessed on 2020-02-01)

This can be carried out by fixing remaining errors of OCR results (named as Post-OCR processing) as well as re-establishing logical structures (named as Table of Contents extraction) defined by book authors (e.g., chapters, sections, etc.).

Post-OCR processing is in charge of detecting and correcting erroneous OCRed tokens. Post-OCR error detection is meant for identifying positions of incorrect tokens by using dictionaries or other NLP resources. On one hand, the error detection supports human assistants to quickly rectify errors by locating their positions, and on the other hand, it enables to filter out, flag, or selectively reprocess such data if necessary. Furthermore, the error detection produces the input of post-OCR error correction which is intended for rectifying invalid tokens by different techniques.

Table of contents (ToC) extraction is responsible for reconstructing logical book structures. Depending on whether books contain ToC pages or not, ToC can be recognised from such pages and/or derived from the entire book content. ToC extraction approaches will generate a hyperlinked table of contents of each digitised book. Such hyperlinked table contains title, link, deep level for each ToC entry. For example, given a ToC entry as a book chapter, its title is the chapter title, its link is the physical page number, and its depth level is the depth at which the chapter is found in the ToC tree.

This dissertation concentrates on facilitating access to historical documents for users and further processes by rebuilding logical document structures and providing higher quality of OCRed text. Indeed, readers easily search and navigate inside documents with the help of structural information and less erroneous text. Search engines and natural language processing applications like text information extraction, text summarisation, topic modeling, sentiment analysis, part of speech tagging, etc., can yield higher performance with cleaner digitised texts.

1.2 Research problems and main contributions

Limitations of modern OCR technologies in handling historical documents lead to difficulties in reading, retrieving as well as further processes on digitised collections. In other words, they have partly reduced the benefits of digitisation projects by preventing readers

from acquiring knowledge from past documents. It is necessary to minimise the influences of such problems of OCR technologies. This, as a result, is the main purpose of the thesis, which complements OCR techniques by providing logical structural information and improving quality of digitised text.

This thesis presents four key contributions which support the usage of historical documents by rebuilding logical book structures, detecting and correcting invalid OCRed tokens. These results are listed below.

1. Digitisation has constantly increased the number of historical digitised books. However, most of them are in a full text mode with some simple structural information. More complicated logical book structures are often missing, which results in difficulties for users to search and browse inside a book. The first part of this work is to enhance OCR methods by extracting these sophisticated structures for digitised books. Such information can then be used to aid user navigation as well as to improve search performance. In Chapter 4, we introduce an ensemble approach [73] which combines prior book structure extraction approaches based on carefully taking advantages of each method.

2. The accuracy of OCR technologies remarkably affects further processes on digital documents. Most parts of the thesis focus on post-processing approaches to improve the quality of OCRed texts. Naturally, a deep understanding of OCRed text, of course, leads to better designing post-processing approaches. Therefore, in Chapter 5, we study characteristics of OCR output and compare them to human generated misspellings [76]. Our analyses, thus, reveal some clues to guide post-OCR approaches.

3. Typically, a post-processing system detects and corrects remaining errors. However, in some scenarios, decoupling these tasks is a wise solution. In mass digitisation projects, source materials may vary regarding different levels of conservation, language, domains, ect. Quality of the resulting OCRed text is consequently diverse as well. In such case, people may prefer to figure out anomalous text regions, and decide further operations on such data rather than automatically rectifying OCR

output. In this thesis, as a result, we consider them as two separate tasks.

Post-OCR error detection is to identify potential erroneous OCRed tokens. We present two novel error detection approaches in Chapter 6. One of them applies binary classification with several new features extracted from a list of plausible candidates for each OCRed token [77]. The other fine-tunes Bidirectional Encoder Representations from Transformers (BERT) models to detect erroneous tokens.

4. Post-OCR error correction is to fix errors with a given list of error positions. Two correction approaches are proposed and reported in Chapter 7. The first one utilises an adaptive edit distance and some other important features in regression model [74]. The second one transforms OCRed text into corrected text based on neural machine translation models with some variations.

1.3 Method overview

In this thesis, some approaches are suggested to enrich access to historical text, which are divided into three tasks, including ToC extraction, post-OCR error detection, and post-OCR error correction.

In terms of ToC extraction, an ensemble method is designed to carefully combine some existing approaches. Since a large portion of books are integrated with physical tables of contents indicating their logical organisations, therefore, most of methods depend on the detection and the analysis on ToC areas to find out and link ToC entries with their corresponding pages. Some others process all book content to extract logical structures, without considering ToC pages. Hybrid ones handle books with or without ToC pages, which recreate book structures by analysing either the ToC areas or the full book content. Performances of such hybrid methods are inferior to those of the method exploiting ToC areas [22, 23, 25]. Our aggregation proposal is different from these methods, we fully combine the analysis on both ToC areas and book content over the same document.

In terms of the post-OCR error detection, two proposed approaches are implemented. The first one employs statistical features extracted from lexicon, error channel, language model to detect errors. Each OCRed token needs to prove itself to be a correct one

among its replaceable candidates through a set of different features from both character and word levels. Inspired by the winner of the competition on post-OCR text correction in ICDAR2019 [89], the second error detection approach exploits neural network based language model to identify incorrect OCRed tokens. BERT is well-known as deep bidirectional language model, which can be fine-tuned to create state-of-the-art models for a wide range of NLP tasks. In our case, BERT model is trained with some modifications in order to find error positions.

Regarding the post-OCR error correction, the statistical and neural machine translation based approaches are developed. The statistical one applies regression model to select the top-matching correction candidate with features related to error channel, and a statistical or neural network based language model. Machine translation (MT) system is adapted to correct errors by translating OCRed text into corrected text. Given a list of error positions or a list of detected errors, the input and output of MT system are character sequences of word ngrams related to errors of OCRed text and those of corresponding ngrams from ground truth (GT) text, respectively.

1.4 Thesis outline

This manuscript is structured as follows:

- Chapter 2 surveys the state-of-the-art work on extracting book structure information (e.g., chapter, section, etc.) and on post-OCR processing. Their advantages and disadvantages are highlighted.

- In Chapter 3, we provide an overview of datasets and measures used to evaluate our models. All of datasets are open access, and come from two competitions of the International Conference on Document Analysis and Recognition on book structure extraction and on post-OCR text correction. Similarly, evaluation metrics are official ones of these two competitions.

- Chapter 4 introduces an aggregation-based method to enhance ToC extraction using system submissions from the ICDAR Book structure extraction competitions (2009,

2011, and 2013). The logical structure information is very helpful for reader to search or browse inside a book.

- In Chapter 5, we study OCR process and provide a detailed comparison between OCR errors and human spelling mistakes. The deep analysis on invalid OCRed tokens is included towards developing effective post-processing approaches.

- Two error detectors are reported in detail in Chapter 6. The first approach employs different features of both character and word levels to find erroneous tokens. The second one adjusts the well-known contextual language model and static word embeddings to detect errors.

- In Chapter 7, two correction approaches are presented. One of them explores a modified way of candidate generating and candidate scoring to select the best candidate for each error. The other one adapts machine translation technique to transform OCR errors to corrected tokens with some extra features.

- Finally, the thesis concludes in Chapter 8 and gives an outlook on future work.

1.5 List of publications

Results presented in this thesis were published by the author, as listed below:

- Thi Tuyet Hai Nguyen, Adam Jatowt, Nhu-Van Nguyen, Mickaël Coustaty, and Antoine Doucet. Neural Machine Translation with BERT for Post-OCR Error Detection and Correction. In 20th ACM/IEEE Joint Conference on Digital Libraries, JCDL 2020, **Accepted**. [Chapters 6 and 7].

- Thi Tuyet Hai Nguyen, Adam Jatowt, Mickaël Coustaty, Nhu-Van Nguyen, and Antoine Doucet. Post-OCR error detection by generating plausible candidates. In 15th IAPR International Conference on Document Analysis and Recognition, ICDAR 2019, Sydney, Australia, 2019. [Chapter 6].

- Thi Tuyet Hai Nguyen, Adam Jatowt, Mickaël Coustaty, Nhu-Van Nguyen, and Antoine Doucet. Deep statistical analysis of OCR errors for effective post-ocr processing. In 19th ACM/IEEE Joint Conference on Digital Libraries, JCDL 2019, Champaign, IL, USA, pages 29-38, 2019. [Chapter 5].

- Thi Tuyet Hai Nguyen, Mickaël Coustaty, Antoine Doucet, Adam Jatowt, and Nhu-Van Nguyen. Adaptive edit-distance and regression approach for post-OCR text correction. In Maturity and Innovation in Digital Libraries - 20th International Conference on Asia-Pacific Digital Libraries, ICADL 2018, Hamilton, New Zealand, pages 278-289, 2018. [Chapter 7].

- Thi Tuyet Hai Nguyen, Antoine Doucet, and Mickaël Coustaty. Enhancing table of contents extraction by system aggregation. In 14th IAPR International Conference on Document Analysisand Recognition, ICDAR 2017, Kyoto, Japan, pages 242-247, 2017. [Chapter 4].

CHAPTER 2

Related work

Results of OCR technologies are missing high-level structures of document contents (e.g., chapter, section) and still erroneous. A lot of efforts are dedicated to overcome such limitations aiming at facilitating the usage of digitised historical texts. Several methods are suggested to rebuild logical book structures which not only enrich user navigation experiences but also structure digitised documents for better structure retrieval performance. Likewise, there are a wide range of post-OCR approaches that improve the quality of OCRed text by identifying and rectifying remaining erroneous tokens. In this chapter, we give a detailed overview of multiple approaches on reconstructing high-level book structures and those on post-processing.

2.1 Table of contents extraction

Along with the development of digital libraries, several digitised historical books have been accessible via the Internet. Many approaches have been proposed to rebuild book structures, aiming to provide users a convenient way to locate or browse their content of interest. There are some challenges in dealing with extracting table of contents that is a

list of ToC entries each of which includes three elements: title, page number and level of the title. On one hand, the limitation of OCR technologies on historical documents causes troubles for structuring analysis, especially when some keywords like 'chapter', 'section' are wrongly recognised. On the other hand, historical books have various layout formats. Figures 2.1, 2.2 and 2.3 illustrate some different ToC types such as hierarchical ToC, ToC with multiple-line entries or ToC in double-column layout.

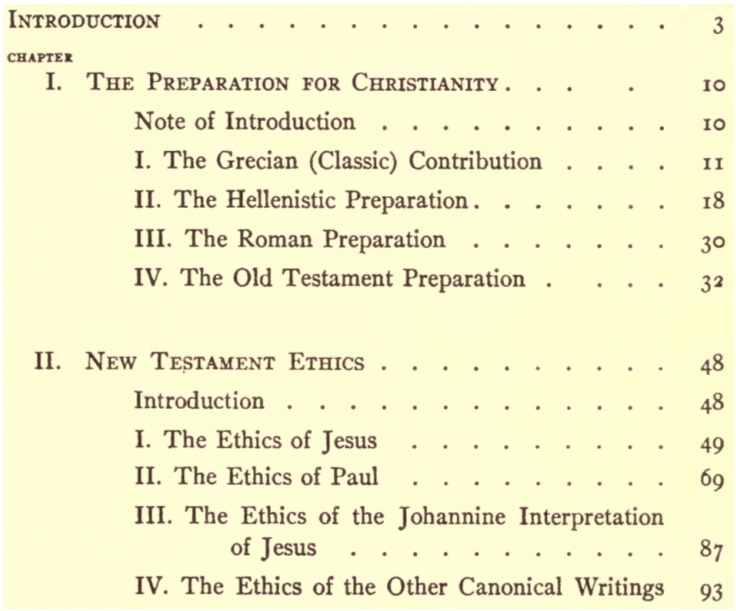
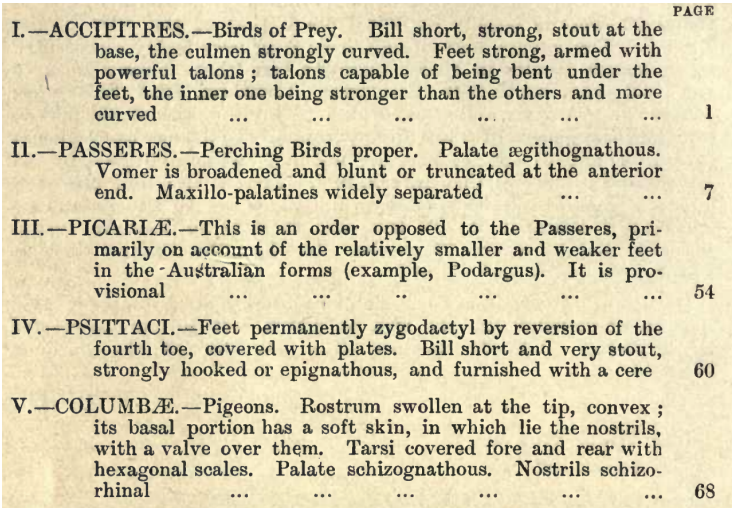


Figure 2.1: Hierarchical ToC from 'History of ethics within organized Christianity', Thomas Cuming Hall, 1910.

Book structure extraction approaches can be classified into three types: methods based on the recognition and analysis of pages containing a table of contents, methods analysing the full content of the book and notably looking for internal titles, and hybrid methods [21].

ToC-recognition-based type. The first approach type relies on the detection of ToC pages. Typically, approaches of this type concentrate on detecting ToC pages within a book, then extracting all ToC entries from such pages. Next, the remaining book content is only processed for identifying links between titles and pages.

The best performing approach (named as MDCS) of the three competitions belongs to this type, and was developed by Dresevic *et al.* [26]. The approach recognised ToC pages and assigned each physical page with a logical page number. After that, each ToC page



PAGE

I.—ACCIPITRES.—Birds of Prey. Bill short, strong, stout at the base, the culmen strongly curved. Feet strong, armed with powerful talons; talons capable of being bent under the feet, the inner one being stronger than the others and more curved 1

II.—PASSERES.—Perching Birds proper. Palate ægithognathous. Vomer is broadened and blunt or truncated at the anterior end. Maxillo-palatines widely separated 7

III.—PICARIÆ.—This is an order opposed to the Passeres, primarily on account of the relatively smaller and weaker feet in the Australian forms (example, Podargus). It is provisional 54

IV.—PSITTACI.—Feet permanently zygodactyl by reversion of the fourth toe, covered with plates. Bill short and very stout, strongly hooked or epignathous, and furnished with a cere 60

V.—COLUMBÆ.—Pigeons. Rostrum swollen at the tip, convex; its basal portion has a soft skin, in which lie the nostrils, with a valve over them. Tarsi covered fore and rear with hexagonal scales. Palate schizognathous. Nostrils schizorhinal 68

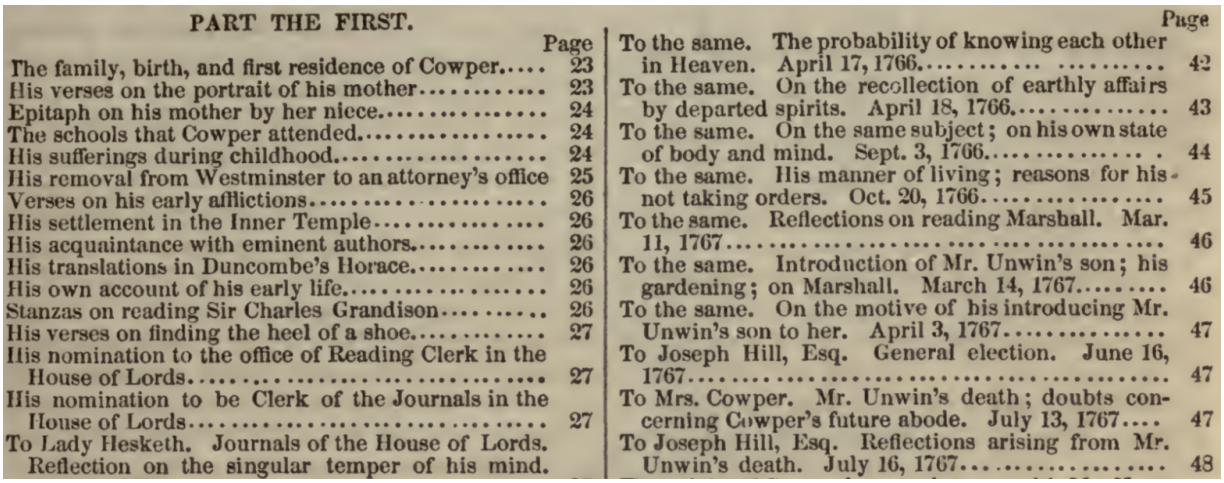
Figure 2.2: Multiple-line ToC entries from 'A key to the birds of Australia : with their geographical distribution', Robert W. Hall, 1906.

was analysed for ToC sections whose important parts were processed to detect titles and corresponding page numbers. In the next step, a fuzzy search technique was applied to identify links between titles and page numbers. All parts of this ToC extraction engine were based on pattern occurrences obtained from their training datasets.

Several methods of this type used ad-hoc rules derived from limited datasets to locate and parse ToC entries [57, 98]. Instead of applying the same rules for all ToC layouts, Wu *et al.* [103] studied the detail content of ToC pages and took the diversity of ToC layouts into consideration. They introduced three basic ToC layout styles, namely, 'flat', 'ordered', and 'divided'. They further designed three corresponding rule-based techniques for processing each of these styles. The ToC layout style classification was used as a complement step before the extraction of ToC entries.

The Epita approach [56] relied on the text boxes provided in portable document format (PDF) documents. It first searched the ToC areas and then reconstructed ToC entries based on the features of text boxes: alignments, lines ending with numbers or not, etc. Page linking was found using the difference between the page number of a page in the middle of the book and its page number written in the book.

The disadvantage of this kind of approach is that it mainly relies on ToC pages to extract ToC entries, therefore its performance can be significantly decreased in case of



PART THE FIRST.

Figure 2.3: ToC in double-column layout, from 'The works of William Cowper : his life, letters, and poems', William Cowper, Thomas Shuttleworth Grimshawe, John William Cunningham, 1857.

books without ToC pages, or whenever the physical or digitised version of the ToC pages is damaged or altered.

Book-content-based type. To overcome the problems of the first approach type, the second type focuses on the analysis of the entire book content instead of focusing on its ToC pages. The representative approach of this type (named as Greyc) was presented by Giguet *et al.* [31, 32]. They used a four-page window to search a large white-space which was considered as a strong indicator of the ending of a chapter and the beginning of a new one. After finding the relevant pages, they extracted entry titles from the third page of the sliding window.

Similarly, Déjean *et al.* [16] exploited page breaks to identify book parts. The breaks associating with high-level book structures (part, chapter) create white-spaces on top of page (leading pages) or bottom of pages (trailing pages). They reported that their approach achieved very good precision and lower recall since other structures unmarked a page break are not detected.

In our point of view, approaches of this type are totally unsupervised and language-independent. They are capable of handling documents with or without ToC as well as are not affected by erroneous ToC pages. However, they require a large memory for processing the whole document even in the case of a book with clear and exhaustive ToC areas.

Hybrid type. In out-of-copyright books, it is observed that as many as 20% books do not contain any ToC [24]. It, thus, seems necessary to use approaches of the hybrid type. In particular, they enable to extract logical book structures from books with and without ToC pages.

The method of Liu *et al.* [59] (named as Nankai) considered whether a book has ToC pages or not, then applied the appropriate approach. A rule-based method was designed for books with ToC pages while machine learning was used to deal with books without ToC pages.

Different from approaches applying traditional rule-based method and classical boolean logic, Gander *et al.* [30] utilised the power of rule-based method with the flexibility of the fuzzy logic, aiming to better handle several OCR flaws as well as variations in the book structure styles. Additionally, results were carefully refined by a grammar-based method in the final step. Their method was called as Innsbruck in the competition.

Another approach (named as XRCE) [17, 18] combined a rule-based method, a supervised one and similar strong indicators Déjean *et al.* [16] in order to extract the ToC entries. Four techniques were applied in their suggestion. The first and second ones used a rule-based technique to parse ToC pages and index pages. The supervised method relying on five generic properties (contiguity, textual similarity, ordering, optional elements, no self-reference) and on some document layout specificities was the core of the third method. The last one relied on the leading and trailing page white-spaces.

This type of approach is promising in that it could properly handle all books, with or without ToC pages. However, it still underperforms the MDCS approach in all the three competitions on book structure extraction.

Conclusion. In summary, no approach has fully combined the features from the ToC pages and those from the book content, even in the case of the hybrid methods. The latter underperforms the best ToC-recognition-based approaches according to the official results of the three ICDAR book structure extraction competitions [23–25].

Our analysis of the submissions to the three competitions indicates that the MDCS approach always obtains the best performance on 1,653 books with ToC pages. The XRCE

[1]Two evaluation metrics (i.e., title-based, link-based) are described in detail in Section 3.2.1.

Table 2.1: Detail performance scores over the ICDAR 2009 competition dataset on title-based and link-based measures[1].

Method	F-measure			
	Books with ToC		Books without ToC	
	Title-based	Link-based	Title-based	Link-based
Greyc [32]	0.07	1.5	0.13	0.5
Noopsis [23]	10	47.7	0.87	2.8
XRCE [17, 18]	33.17	72.4	**7.81**	**22.6**
MDCS [26]	**50.84**	**78.8**	0.13	7.4

Table 2.2: Detail performance scores over the ICDAR 2011 competition dataset on title-based and link-based measures.

Method	F-measure			
	Books with ToC		Books without ToC	
	Title-based	Link-based	Title-based	Link-based
Greyc [32]	9.47	52.5	6.9	42.5
XRCE [17, 18]	19.02	58.4	**26.32**	**53.8**
Nankai [59]	38.85	71.6	7.93	26.9
MDCS [26]	**48.96**	**78.2**	5.12	8.6

approach achieves the highest performance on the 187 books without ToC pages of the competition datasets in ICDAR 2009 and ICDAR 2011. The Greyc approach is the best on the 167 books without ToC pages of the ICDAR 2013 competition dataset. Tables 2.1, 2.2 and 2.3 illustrate the performance scores observed over the three competition datasets.

As a consequence, we propose a novel ensemble approach which builds on previous works. The notable difference between our proposal and past methods is that we combine both types of techniques: those based on ToC pages and those based on book content. In other words, we benefit from both of them to construct a hyperlinked ToC. Our approach is implemented relying on two set operators (the union and the intersection) applied on two properties of ToC entries (title and page number). The full details on this contribution are provided in Chapter 4.

Table 2.3: Detail performance scores over the ICDAR 2013 competition dataset on title-based and link-based measures.

Method	F-measure			
	Books with ToC		Books without ToC	
	Title-based	Link-based	Title-based	Link-based
Greyc [32]	8.74	47	**9.18**	**35.4**
Epita [56]	18.06	41.9	0.07	1.8
Wurzburg [25]	22.13	48.6	7.53	26.5
Innsbruck [30]	36.17	74.2	8.2	33.6
Nankai [59]	42.65	73.8	0.7	7.5
MDCS [26]	**52.67**	**80.1**	0.2	1.9

2.2 Post-OCR processing

Post-OCR processing approaches detect and correct remaining OCR errors for yielding better quality of digitised documents. The literature of post-OCR processing research has a rich family of models. They are grouped into three types: manual approach type which lets human manually review and correct OCRed texts, lexical approach type towards the comparison of source words to dictionary entries, neural network and statistical approach one that utilises information learnt from training data. The insights of each group are discussed in the following sections.

2.2.1 Manual approach type

ReCAPTCHA and crowd-sourcing are some key approaches of this type. CAPTCHAs (Completely Automated Public Turing test to tell Computers and Humans Apart) are widespread security measures on the World Wide Web. Von Ahn *et al.* [100] suggested to benefit from CAPTCHAs to digitise old printed material. They concealed crowd-sourcing effort in OCR correction behind an access system to websites. Users were shown two images; one was known to the system and used for verifying access to a website; another was unknown and its content would be determined by majority vote of contributors. Users did not know which one was known or unknown to the system. The authors reported that

the reCAPTCHA system achieved a word-level accuracy of 99.1% whereas standard OCR approaches on the same set of articles obtained only 83.5%. An example of reCAPTCHA is shown in Figure 2.4.

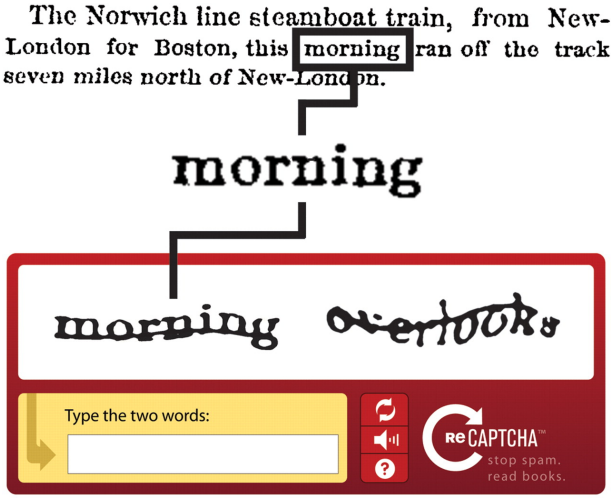


Figure 2.4: An example of reCAPTCHA [100].

Crowd-sourcing approaches for post-correction of OCR output have been successfully applied to several historical text collections. Collaborative correction on a large amount of text requires a easy-to-use user interface, and continuous efforts for keeping the volunteer proofreaders motivated. Several crowd-sourcing systems have been built to profit from the public effort in order to enhance the quality of digitised texts.

One of the first crowd-sourcing approaches was a web-based system called *Trove* [37][2]. This system was developed by the National Library of Australia for crowd correction in historical Australian newspapers. The approach presented full articles to volunteers, and allowed to fix text line by line. A screenshot of the web-based Trove system [37] is demonstrated in Figure 2.5.

Clematide *et al.* [12] reported a crowd-correction platform called *Kokos* to improve the OCRed text quality of the yearbooks of the Swiss Alpine Club (SAC). *Kokos* showed full documents to users and let them correct errors word by word instead of line by line like *Trove*. More than 180,000 characters on about 21,000 pages were corrected by volunteers in about 7 months, word accuracy of 99.7%. Figure 2.6 shows a screenshot of the web-based Kokos system.

[2] https://help.nla.gov.au/trove/for-digitisation-partners (accessed 2019-11-01)

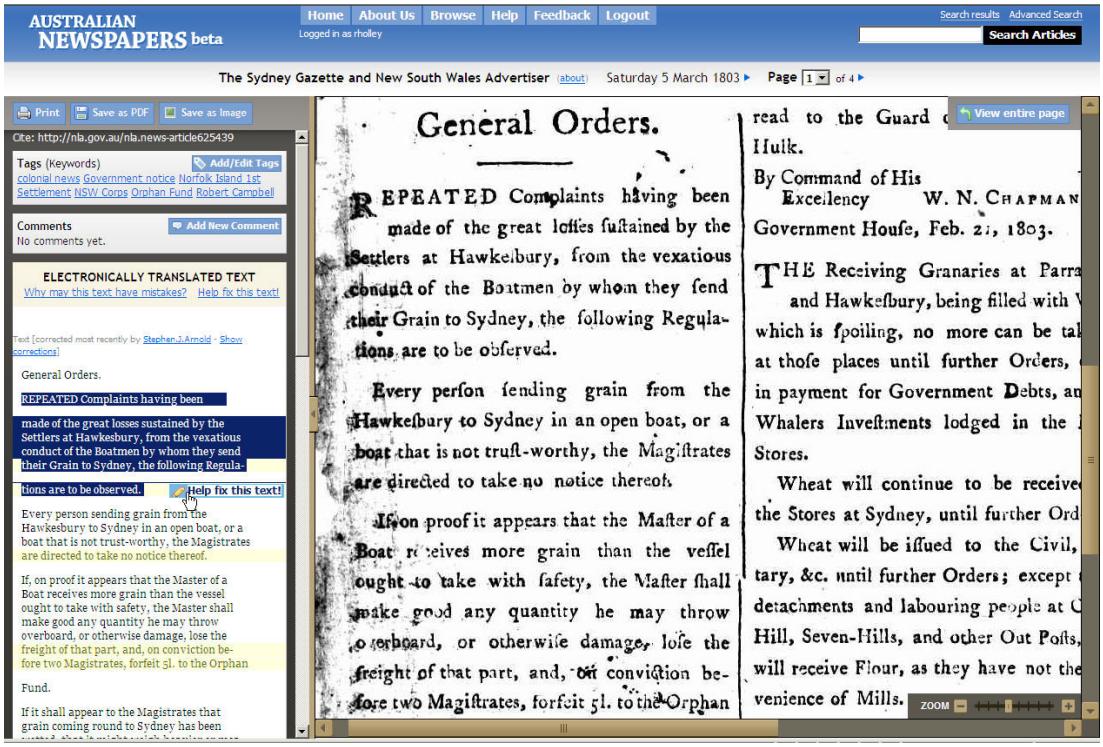


Figure 2.5: User interface of the Trove system [37].

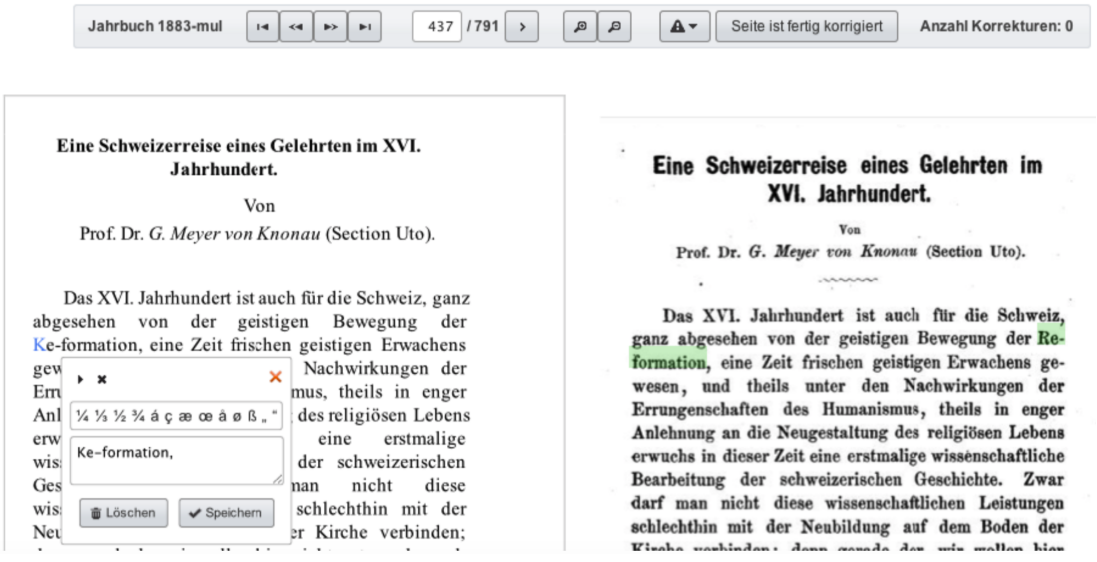


Figure 2.6: A book page in Kokos [12].

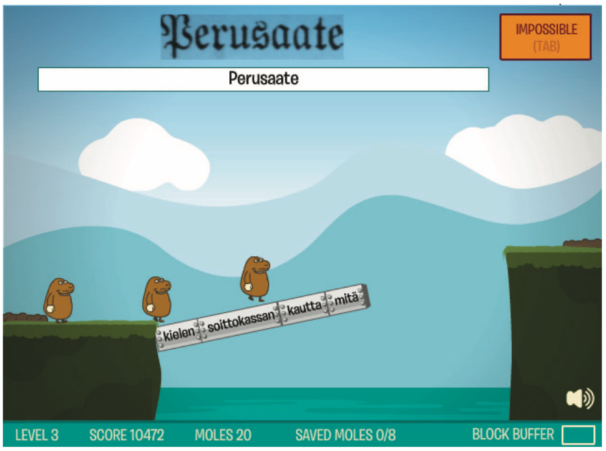


Figure 2.7: An example of Digitalkoot [10].

Instead of showing full articles to users, another system (named *Digitalkoot*) broke the articles into single words without any context and inserted them to simple games. The objective of this model was to attract volunteers to donate their time for correcting erroneous tokens. This approach was developed by Chrons *et al.* [10] for rectifying OCR errors in old Finnish newspapers. An example of Digitalkoot is given in Figure 2.7. The experimental results reported that the quality of corrected text was very high with word accuracy over 99%. However, the system ignored nearby context and only allowed users to interact with single words, which raises a doubt that real-word errors cannot be corrected.

While collaborative OCR correction approaches proved their performance with high accuracy, they also have some limitations. They require the original documents which are often unavailable on some OCRed text corpus. In addition, they heavily depend on volunteer work as well.

2.2.2 Lexical approach type

The approaches of the lexical type typically utilise distance measures between an erroneous word and a lexicon entry to suggest candidates for correcting OCR errors.

Some researchers focus on specialised lexicons to improve the accuracy of Abbyy FineReader on historical documents. Gotscharek *et al.* [33] used a corpus of historical German documents from before 1500 to 1950 to construct different types of lexicons, and evaluated them on three sets of test documents. The authors confirmed that lexicons

helped to reduce error rate of the original OCRed text, with 28.20%, 42.00%, 59.06% on the 16th-century, 17th-century, and 18th-century test set, respectively.

Considering the fact that spellings in historical documents are often not standardised and historical stages of a language often lack complete lexicons, some prior works aim to study the influence of the coverage of a lexicon. Then different ways are suggested to dynamically collect specialised lexicons.

Strohmaier *et al.* [94] argued that conventional dictionaries were short of a considerable number of tokens of a specific thematic area, which would drastically decrease performance of post-OCR processes. They suggested to exploit thematic dictionary to improve results of approaches belonging to the lexical type. They built a dynamic dictionary from collecting vocabularies of Web pages of the input domain, which obtained a higher coverage than static conventional dictionary.

Ringlstetter *et al.* [90] emphasised that approaches of this type only achieved a high performance if dictionary was sensitive to the document domain. They refined the crawl strategy employed in the previous approach [94] to produce smaller dictionaries with high coverage. In particular, the similarity between crawled pages and the given input text was controlled based on the normalised cosine distance. Web pages with many orthographic errors were removed from dictionary construction.

Instead of building a dictionary, Bassil *et al.* [5] profited Google's massive indexed data for post-processing OCR output. They sent OCRed tokens to Google search engine as search queries. If the query contained errors, the search engine would suggest some replaceable words for misspellings. These suggestions were considered as corrections for OCR errors. One of competition participants (named as EFP) [8] also explored lexicon look-up techniques and regular expressions to detect and correct errors.

Lexical approach type is easy to apply, however, it also goes together with some difficulties. Historical documents do not follow a standard spellings like modern texts and often lack complete lexicons. Moreover, the approaches of this type only concentrate on single words so that they cannot handle real-word errors which are valid lexicon items but occur in wrong context.

2.2.3 Neural network and statistical approach type

Most of post-processing approaches are statistical, which enable to model specific distributions of the target domain from available training data. Some methods of this type incorporate different digitised outputs of the same paper-based document to benefit from each other. Some approaches exploit error model and language model in different ways to detect and correct remaining erroneous tokens. Some others profit from machine translation techniques in order to transform OCRed text into corrected one.

Merging OCR outputs

One direction of work combines multiple OCR outputs, which typically includes three steps. OCR texts are firstly created from multiple OCR engines on the same input or from the same OCR engine on different digital versions of the original document. Secondly, alignment algorithms are applied to align the OCR output. Finally, different decision methods are explored to select the final output.

In the first step, approaches obtain multiple outputs by different ways. Lopresti *et al.* [61] employed the same OCR engine on several scans of the same document. Instead of using different scans of the same document, Lund *et al.* [62] adjusted the binarisation threshold to create multiple versions of the original documents. Otherwise, Lin [58] and Boschetti *et al.* [6] used different OCR engines to digitise the same document.

Next, some alignment methods have been developed to align multiple OCR output sequences. Lund *et al.* [63] introduced an efficient algorithm to align the output of multiple OCR engines and then to take advantages of the differences between them.

In the last step, several techniques were applied to choose the best sequence. Lopresti *et al.* [61], Lin [58], and Lund *et al.* [62] utilised voting policy to decide the best sequence. Boschetti *et al.* [6] selected characters using Naive Bayes classifier. Some kinds of features (voting, number, dictionary, gazetteer, and spelling checker) were used in Maximum entropy classification methods to choose the best possible correction by Lund *et al.* [64].

These approaches of this type proved their benefit with lower word error rate than the individual OCR engine. However, they limit candidate suggestions from the recognition output of OCR engines. In addition, the approaches do not consider any contextual infor-

mation, thus, real-word errors are impossibly corrected. Furthermore, they require some additional efforts of multiple OCR processing and the presence of the original OCR input which are not always available in some datasets of post-OCR processing task. Collections of the two competitions on Post-OCR Text Correction [8, 89] are some of typical examples.

Error model and language model

Several supervised and unsupervised approaches exploit error model and language model to deal with post-OCR processing problems.

Some approaches mainly investigate the error model and ignore context information. Given an OCR error, CSIITJ - a competition team [89] selected a list of candidates based on edit distance between the error and lexicon entries. These candidates were, then, ranked by the error model and suggested as corrections.

Kolak and Resnik [52] considered the post-processing problem as a recognition one and adapted a framework of syntactic pattern recognition to solve the problem. Parameters of their model were estimated by Levenshtein distance and IBM translation models.

Perez-Cortes *et al.* [81] applied the extended version of the Viterbi algorithm to find the lowest cost path through a directed graph associated to the stochastic finite-state automaton and to the input string. Their experiments showed some improvements on correcting handwritten Spanish names at character level.

Extending the approach of Perez-Cortes *et al.* [81], Llobet *et al.* [60] built an error model and a language model, then added one more model built from character recognition confidences called hypothesis model. Three models were compiled separately into Weighted Finite-State Transducers (WFSTs), then were composed into the final transducer. The best token was the lowest cost path of this final transducer. However, character recognition confidence is often unavailable in some digitised corpus, such as datasets of three works [8, 27, 89], so it is hard to implement this hypothesis model.

A competition team from Centro de Estudios de la RAE [8, 89] also implemented a WFST based method (denoted as WFST-PostOCR in the first competition, RAE in the second one). The RAE team compiled probabilistic character error models into WFST. Ngram language models and the lattice of candidates generated by the error model were

used to decide the best token sequence. This approach obtained the best performance on detection task of the competition ICDAR2017, and improved the quality of OCRed text in the two competitions.

Another competition participant (called as 2-pass RNN) [8] examined neural language model instead of statistical one. Erroneous tokens were detected based on two recurrent neural network (RNN) models. Features of character-level model were used as the input of the word-level model.

Some approaches apply the error model along with the language model to deal with both non-word and real-word errors. Tong and Evans [97] implemented a context sensitive correction system. The approach took advantages of information from multiple sources, including letter ngrams, character confusion probabilities, and word-bigram probabilities. Candidates were selected by comparing OCRed token ngrams with lexicon-entry ngrams, then ranked by the conditional probability of the token being recognised as matches. Finally, statistical language modeling was used to determine the best scoring word sequence.

Taghva *et al.* [95] generated candidates by confusion lists, then scored them using Bayesian function on frequencies of word pairs and character ngrams. In addition, a heuristic was designed to create candidates for words containing unrecognized characters. The ranked list of candidates was then recommended to users.

Evershed *et al.* [27] introduced post-processing approaches applying both error model and word language model. They carefully generated candidates at character level using the error model and at word level using word trigram and 'gap' trigram. The error model used confusion matrix and the novel reverse OCR derived estimation. Suggestions were ranked by the confusion cost from the error model, and trigram language cost. In our opinion, more clues can be employed to select the best matching candidate, for example, candidate frequency, costs related to skip-grams, or part-of-speech tagger.

Some unsupervised post-OCR approaches have been developed. Reynaert [85, 86] introduced an unsupervised method to solve problems of spelling variation. The method exploited a hash table and a hash function to produce a large number for identifying words (anagrams) which have the same characters in common. The main characteristic of this hash function is that each character or character sequence of a word can be calculated

separately. This feature enabled to retrieve similar words for a given word by inserting, deleting, substituting or transposing characters. The results showed that a large amount of non-word errors were detected and corrected.

Anagram hash algorithm was then applied in another unsupervised approach. Niklas [78] combined this hash algorithm and a new OCR adaptive method in order to search the best matching proposals for OCR erroneous tokens. The proposed method first classified characters into equivalence classes based on their similar shapes. Characters of the same class would share the same OCR-key. The key of the given word was utilised to retrieve all words which had the same key in the dictionary. In addition, context information (word bigrams) was also applied to score suggested words.

Regression model

Machine learning approaches learn from different features, which enable more robust candidate selection. These approaches explore multiple sources to generate candidates, extracted features and then rank them using a statistical model.

Kissos and Dershowitz [48] computed feature values for each input, including confusion weight, OCR confidence, frequencies (unigram frequency, related bigram frequency, term frequency in OCRed document). These features were used to decide whether the OCRed word should be replaced by its highest ranked correction-candidate. However, it should be reminded that OCR confidence is not always available in OCRed text collections, therefore that model cannot fully be implemented. Furthermore, their features were not designed to deal with segmentation errors. For example, frequency of unigram does not take run-on errors into account (e.g., there is not frequency of unigram if an error is as 'doubtfud.of', its candidate is as 'doubtfull of'). In addition, that approach did not consider an important feature employed in error correction [39], which is the similarity between an error and its candidate.

Mei *et al.* [66] argued that the above method made use of solely ngram frequencies without knowing the characteristics of OCR errors. They identified errors relying on frequencies of word and word ngrams. A token was viewed as an error if its frequency or its ngram frequencies in the same document were less than a threshold. They suggested a sim-

ilar approach with some additional features, such as similarity metrics (Levenshtein edit distance, longest common subsequence), contextual information (word ngram frequency, skip-gram frequency). However, they ignored another important feature - confusion probability which was used in several successful post-processing approaches [52, 60, 81].

Khirbat [45] classified a token as being incorrect or correct by using three following features: the presence of non alpha-numeric text in the token, the occurrence of the token and its context in other places of the same document, the comparison between its word bigram frequency and a threshold. They rectified the errors by employing similar features, which were exploited in the work of Mei *et al.* [66], to score candidates based on simulated annealing [47].

Machine translation model

Along with the development of machine translation techniques, some approaches considered OCR post-processing as machine translation (MT) task, which transforms OCRed text into the corrected one in the same language.

Afli *et al.* [1] successfully trained statistical machine translation (SMT) system to reduce OCR errors of historical French texts. They concluded that word-level SMT systems performed slightly better than character-level systems for OCR post-correction [1], and the word-level systems outperformed language model techniques [2]. However, it should be mentioned that they had a large training set of over 60 million tokens while some datasets were much smaller.

Schulz and Kuhn [91] presented a complex architecture named Multi-Modular Domain-Tailored (MMDT) for OCR post-correction of historical texts. This approach combined many modules from word level (e.g., original words, spell checker, compounder, word splitter, text-internal vocabulary) to sentence level (i.e., SMT) for candidate suggestion. Then, the decision module of Moses decoder [50] was used to rank candidates.

Following the success of translation task in post-OCR, some competition teams (Char-SMT/NMT, CLAM, CCC, UVA) employed methods of character-based machine translation to correct OCR errors.

CLAM and UVA depended on neural machine translation while Char-SMT/NMT [3]

combined neural machine translation and statistical machine translation. Amrhein and Clematide [3] reported that SMT systems outperformed NMT systems in error correction, while NMT systems obtained higher performance in error detection. Their complicated ensemble models achieved the best result in the correction task of the first competition on post-OCR text correction [8].

The Context-based Character Correction (CCC) method is the winner of the second competition [89]. It fine-tuned the pretrained language model BERT [20] with some convolutional layers and fully-connected layers to identify OCR errors. Their correction model is an attention sequence to sequence model with fine-tuning BERT.

It is obvious that participants applied various methods to detect and correct OCR errors in the first two competitions on post-OCR text correction [8, 89]. Both of winners (Char-NMT/SMT, CCC) utilised character-level machine translation techniques with some additional features. Their methods outperformed most of other methods, such as approaches based on error model and language model (such as weighted-finite-state-transducer, anagram hashing) and lexical approaches.

Conclusion

After studying a wide range of post-OCR processing approaches, we offer two important conclusions below:

1. Most of the above-discussed post-processing approaches focus on the correction part rather than the detection part. Nonetheless, it does not mean that the error detection part is not important as naturally one cannot correct errors without knowing their positions. There are a few separate detection methods relying on lexicon (e.g., EFP [8]), or on word ngram frequencies (e.g., [66]), or on feature-based classification model (e.g., [45]), or on BERT model (e.g., [89]). Some others usually identify incorrect tokens based on the most-matching candidates from the correction part. If the best alternative differs from the OCRed token, then the token is erroneous and replaced by this candidate. The advantage of these approaches is to detect and correct errors at the same time, but the performance of the detection task depends on that of a more difficult task - the correction.

2. Neural machine translation based techniques have been the-state-of-the-art according to the results of the competition on post-OCR text correction. Nevertheless, it should be clarified that there is no clear performance comparison between BERT and feature-based classification models on the post-OCR error detection. Similarly, we still lack obvious evaluations between MT and regression models [45] on post-OCR error correction.

As a consequence, we focus on BERT and feature-based classification models to locate error positions. Likewise, regression and MT models are employed to correct OCR errors. Furthermore, by mainly exploiting natural language processing resources, our post-OCR approaches can handle different digitised documents that created by varying digitisation processes and out-of-date OCR algorithms.

In terms of the post-OCR error detection, the first approach exploits various character-level as well as word-level features to identify whether the OCRed token is incorrect or correct via binary classification. An OCRed token needs to prove to be a valid word via feature values computed from its plausible candidate set. This method differs from the prior work of classification model [45] since we explore more features of both character and word levels. In addition, feature values are computed based on the possible alternative set of each OCRed token instead of only relying on the erroneous token.

The second detection approach adapts BERT model on the named entity recognition task to identify errors. Our proposal is similar to the best-performing approach in the competition ICDAR2019, but we simplify the model with only one fully-connected layer on the top of the hidden-states output. In addition, our model applies Fasttext [68], Glove [40] as the initial embeddings of our model rather than using randomly numbers.

In terms of the post-OCR error correction, our first method makes use of confusion probability obtained from the noisy channel model of single or multiple edits, and context probability given by language model. Then, these two features and some essential features suggested by related works [48, 66] are used to predict the confidence of each candidate becoming a correction via regression model. Our second correction approach which applies NMT techniques on contextual input data and some additional features (e.g., our character embedding, candidate filter based on length difference) is promising to reduce OCR errors.

This chapter surveys the prior works on book structure extraction and post-processing OCRed text, which provides a general viewpoint about what was done in the state of the art as well as the advantages and disadvantages of each method. Based on such background, we design and implement our novel approaches for aiding access to digitised document collections.

To evaluate these contributions, we need to identify adequate datasets and evaluation metrics, which are the focus of the next chapter.

CHAPTER 3

Evaluation datasets and metrics

After considering the state of the art, we will now detail the way that we evaluate our works aimed at extracting tables of contents from digitised books as well as detecting and correcting OCR errors. Evaluation metrics and datasets play a crucial role in testing the performance of proposed methods. Shared tasks are great opportunities to compare approaches in a controlled setting where all are evaluated using the same measures and the same datasets. That is the reason why we select the official metrics and datasets of two competitions to assess our proposals. This chapter introduces adequate benchmark datasets and associated metrics.

3.1 Datasets

Five datasets are applied to evaluate our approaches on table of contents extraction and post-OCR text correction. All of them are public datasets of competitions from the international conference on document analysis and recognition. In this chapter, five of them are discussed in detail.

Table 3.1: Datasets of the three ICDAR competitions on book structure extraction.

Dataset	Total books	# Books with ToC	# Books without ToC	Participants
ICDAR 2009	527	436	91	MDCS, XRCE, Noopsis, Greyc
ICDAR 2011	513	417	96	MDCS, Nankai, XRCE, Greyc
ICDAR 2013	967	800	167	MDCS, Nankai, Innsbruck, Wurzburg, Epita, Greyc

3.1.1 Table of contents extraction

The reference datasets and metrics for the extraction of tables of contents were defined in the context of the ICDAR conference's series of Book Structure Extraction competitions [24]. We use the three datasets created in the context of the 2009 [23], 2011 [22] and 2013 [25] Book Structure Extraction competitions. Different datasets have been annotated and used for each of the competition, but all of them are composed of books selected from the corpus of the INEX book search track [43] which contains 50,239 books. This book collection is provided by Microsoft Research and the Internet Archive [44].

The three subsets preserve the diversity of the large book collection in both book genre and the observed ratio of books with and without a physical table of contents pages (80:20). Details on the three datasets of the competitions are given in Table 3.1, together with the corresponding lists of participants.

Each book of these datasets is provided in two different formats [23]. Portable document format is provided to participants to give them access to original image files, while DjVu XML is provided as the output of an OCR process including OCRed text and basic structure provided in a simple markup format illustrated and described as follows:

```
<DjVuXML>
 <BODY>
  <OBJECT data="file ..."  [...] >
   <PARAM name="PAGE"  value="[...]" >
   [...]
   <REGION>
```

```
      <PARAGRAPH>
       <LINE>
        <WORD coords="[...]"> Moby </WORD>
        <WORD coords="[...]"> Dick </WORD>
        <WORD coords="[...]"> Herman </WORD>
        <WORD coords="[...]"> Melville </WORD>
         [...]
       </LINE>
        [...]
      </PARAGRAPH>
     </REGION>
      [...]
   </OBJECT>
    [...]
  </BODY>
</DjVuXML>
```

An <OBJECT> element corresponds to a page in a digitised book. A physical page number is given as the attribute @value of the <PARAM> element. Inside a page, each paragraph is marked up. Each paragraph element includes line elements, in which each word is showed separately. Coordinates of a rectangle surrounding a word are given as attributes of word elements.

3.1.2 Post-OCR processing

As we saw in Chapter 2, post-OCR processing has been a long standing problem. However, few public evaluation benchmarks exist. The competition on Post-OCR Text Correction organised in 2017 and 2019 in the context of the ICDAR conference [8, 89] therefore attracted strong interest from the community, with around 70 registrations in total. This shared task provides a good opportunity to compare techniques for the detection and correction of OCR errors. Therefore, we rely on the datasets of this competition to evaluate

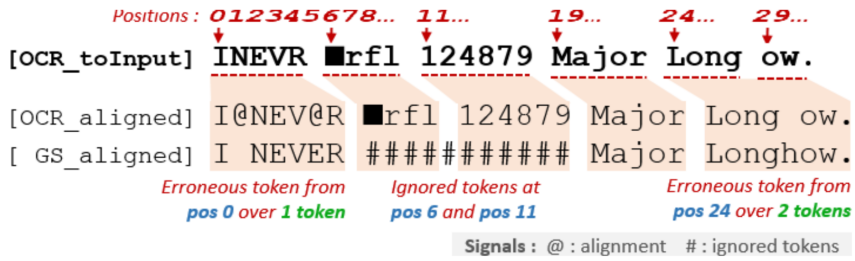


Figure 3.1: A sample of training data in the 2017 competition dataset [8].

the performance of our proposed methods.

The dataset of the 2017 competition was built within the AMELIOCR project [1]. It contains OCRed text from historical documents in English and French taken from different digital collections available at the National Library of France (BnF) and the British Library (BL). The corresponding ground truth was created through different projects such as Europeana Newspapers, IMPACT, Project Gutenberg, Perseus and Wikisource. There is no detailed information about which OCR engines were used to generate the OCRed text of the competition dataset. The full dataset of the first competition includes around 12 million OCRed characters along with the corresponding ground truth, with an equal split of English and French data. The dataset is distributed as a training set of 10 million characters and an evaluation set of 2 million characters.

The data is split into multiple files with three lines of text for each file. The first line contains the original output of the OCR system, while the OCRed text is aligned with the ground truth on the second line. The last line contains the aligned ground truth. The '@' character is used as a padding symbol in the aligned sequences. Any text that could not be identified with certainty in the original image is aligned with the '#' character. One example of the training data is illustrated in Figure 3.1.

The dataset of the second competition, organised in 2019, consists of 22 million OCRed characters (754,025 tokens) along with the corresponding ground truth, covering 10 European languages. The OCRed text includes documents originating from digitised collec-

[1]The project is in the collaboration between the National Library of France (department of preservation and conservation) and the L3i laboratory of the university of La Rochelle.

tions. The GT comes from different sources such as HIMANIS [2], IMPACT [3], IMPRESSO [4], Open data of National Library of Finland [5], GT4HistOCR [93] and RECEIPT [4].

Similar to the dataset of the first competition, there is no information about the OCR engines and configurations that produced the OCRed text. 80% of the dataset is given to participants as training set, while the rest is utilised for evaluation.

This thesis uses the English OCRed texts of both competitions. There are 813 files written in English in the dataset of the first competition. All these files are published either in periodicals or monographs, and the competition organisers divide them into two datasets: Monograph and Periodical. The dataset of the second one contains 200 files in English, which are stemming from IMPACT project. Details of our evaluation datasets are shown in Table 3.2.

Table 3.2: Description of the datasets used for post-OCR processing evaluation; CER denotes character error rate[6], # Files denotes a number of files, # Chars denotes a number of characters; '-' denotes no information.

Dataset	Source	Type	Dates	CER(%)	# Files	# Chars
Monograph	BL Monog	monographs	1858 - 1891	1	667	1.2M
	GT BnF Eng	monographs	1802 - 1911	2		3M
Periodical	BL Euro NP	periodicals	1744 - 1894	4	59	1.8M
Comp2019	IMPACT	-	-	21.28	200	0.24M

3.2 Metrics

To ease comparison with state of the art, and because our research problems are aligned with the settings of the given competitions, we apply the same evaluation metrics as those competitions did. The following sections describe each measure in detail.

[2] http://www.himanis.org (accessed 2019-12-02)
[3] https://www.digitisation.eu (accessed 2019-12-02)
[4] https://impresso-project.ch (accessed 2019-12-02)
[5] https://digi.kansalliskirjasto.fi/opendata (accessed 2019-12-02)
[6] CER is computed as Levenshtein distance that is the minimum number of operations required to transform the reference text into the output.

3.2.1 Table of contents extraction

The key concepts used in the evaluation of ToC extraction are defined as follows [23]. Atomic units that make up a ToC are considered as ToC entries. A ToC entry has three properties: *title*, *link*, and *depth level*. Given a ToC entry corresponding to a book chapter, *title* is the chapter title, *link* is the physical page number at which the chapter starts in the book, *depth level* is the depth at which the chapter is found in the ToC tree with the book as the root.

Matching titles. Two titles match if they are 'sufficiently similar', where a similarity is computed based on a modified version of the Levenshtein distance. In particular, they give different costs on modification operations involving a type of a changed character: the cost of alphanumeric modification is 10; the cost of non-alphanumeric one remains 1.

Two strings A and B are 'sufficiently similar' if their distance D is less than 20% and if the distance between their first and last five characters (or less if the string length is small) is lower than 60%. The distance D between strings A and B is computed as follows:

$$D(A,B) = \frac{LevenshteinDist * 10}{min(length(A), length(B))} \tag{3.1}$$

where $LevenshteinDist$ is the modified Levenshtein distance, $length(A)$ is the number of characters of strings A.

This loose definition of similarity was designed to match and consider correct both the ToC entry title that may be found in the book's ToC and the one that may be found at the corresponding page (e.g., sometimes the titles have variations such as being preceded by the word 'Chapter' or not).

Matching links. A link is correctly recognised if an entry has *matching title* linking to the same physical page in the ground truth.

Matching depth levels. A depth level is correct if an entry has *matching title* at the same depth level in the ground truth.

Matching complete ToC entries. A ToC entry is said to be entirely correct if an entry has *matching title*, *matching depth level*, and *matching link*, correspondingly.

Two complementary metrics are yielded to evaluate ToC entries, including a title-

based measure [23] and a link-based measure [19]. The main difference is that the former primarily matches ToC entries based on the similarity of titles, while the latter directly takes into account the quality of the links.

In the title-based measure, ToC entries are firstly assessed on whether their titles are similar to any available titles of the ground truth according to a distance measure mentioned in Equation 3.1, then the links and the depth levels are considered.

Concerning the link-based measure, first of all, ToC entries are tested based on whether they link to a page number that truly matches an existing ToC entry. After that, the similarity of the titles is computed by Equation 3.2, and the depth levels are tested.

$$simil(s_1, s_2) = 1 - \frac{weightedLevenshtein(s_1, s_2)}{max(weight(s_1), weight(s_2))} \tag{3.2}$$

where $weightedLevenshtein$ is similar to $LevenshteinDist$ mentioned in the Equation 3.1, and $weight(s)$ is the sum of each character's weight in the string s (if a character is a letter or a number then its weight is 10, otherwise, its weight is 1).

After considering whether ToC entries match the ground truth or not, three common metrics including Precision, Recall, and F-measure values are computed for each property separately, and for complete entries. Each of three measures is calculated for each book and then averaged over the total number of books (macro-average). Precision, Recall, and F-measure are formulated in Equations 3.3, 3.4, and 3.5, respectively.

$$Precision = \frac{TP}{TP + FP} \tag{3.3}$$

where TP is True Positive, FP is False Positive

$$Recall = \frac{TP}{TP + FN} \tag{3.4}$$

where FN is False Negative

$$F - measure = \frac{2 * Precision * Recall}{Precision + Recall} \tag{3.5}$$

3.2.2 Post-OCR processing

As for post-OCR processing, we employ the same evaluation metrics used in the competition. In the error detection task, the goal is to identify tokens as being either erroneous or not. Therefore, the organisers used the three popular metrics Precision, Recall and F-measure as defined in Equations 3.3, 3.4, and 3.5, respectively.

In the correction task, this official metric is the relative improvement ($impr$) between the original Levenshtein distance ($origDist$) and the corrected distance ($avgDist$).

$$impr = \frac{origDist - avgDist}{origDist} * 100 \tag{3.6}$$

The $origDist$ distance is calculated based on Levenshtein distance between the raw OCRed text (ocr) and its GT (gt) as below:

$$origDist(ocr, gt) = \frac{\sum_{i=1}^{n} LVdistance(ocr_i, gt_i)}{N} \tag{3.7}$$

where ocr_i is an OCRed token, gt_i is a GT token, n is a number of tokens, N is the total number of characters in GT, and $LVdistance$ is the Levenshtein distance.

The $avgDist$ distance relies on Levenshtein distance between the corrected text ($corr$) and the corresponding GT, and the likelihood of each candidate to be the correction in case there are many candidates for one error. It is computed by the following formula:

$$avgDist(corr, gt) = \frac{\sum_{i=1}^{n} \sum_{j=1}^{m} w_{ij} * LVdistance(c_{ij}, gt_i)}{N} \tag{3.8}$$

where w_{ij} is the likelihood of candidate c_{ij} to be a GT word gt_i, m is the number of candidates of an OCRed token corresponding to gt_i.

CHAPTER 4

Table of contents extraction

Digitised books inherently lack logical structure information, such as chapters, sections. To enrich the navigation experience of users, several approaches have been proposed to extract tables of contents (ToC) from digitised books. In this chapter, we introduce and evaluate our hybrid approach designed to combine approaches focused on the identification and analysis of ToC pages, and those that build ToCs by searching for structure throughout the whole book. To do this, we present and apply an aggregation-based method to enhance ToC extraction and use as simulation of systems to be combined the output of submissions to the ICDAR Book structure extraction competitions held in 2009, 2011, and 2013.

4.1 ToC extraction approach

Most multi-page documents come with a built-in table of contents, which naturally reflects the logical structure of the entire document. Consequently, most of the prior work concentrate on recognising ToC pages and extracting ToC entries along with their corresponding page numbers within such ToC area. These approaches are incapable to work in case of books without ToC pages or with too degraded ToC areas. Some other methods

process entire book content to detect ToC entries, which is typically based on leading and trailing page white-spaces. In our opinion, results extracted from ToC pages can complement those extracted from the book content and vice versa. Therefore, we suggest an aggregation method which fully combines the analyses of both ToC pages and book content.

Our ensemble method is built on the top of existing approaches. Specifically, we employ extracted ToCs from submissions of the ICDAR book structure extraction competitions in 2009 [23], 2011 [22], and 2013 [25]. Each submission may be considered as a set of ToCs of books each identified by *bookid*, with for each book a set of ToC entries defined by page number and title. An example of a ToC page and the corresponding submission are shown in Figures 4.1, and 4.2, respectively.

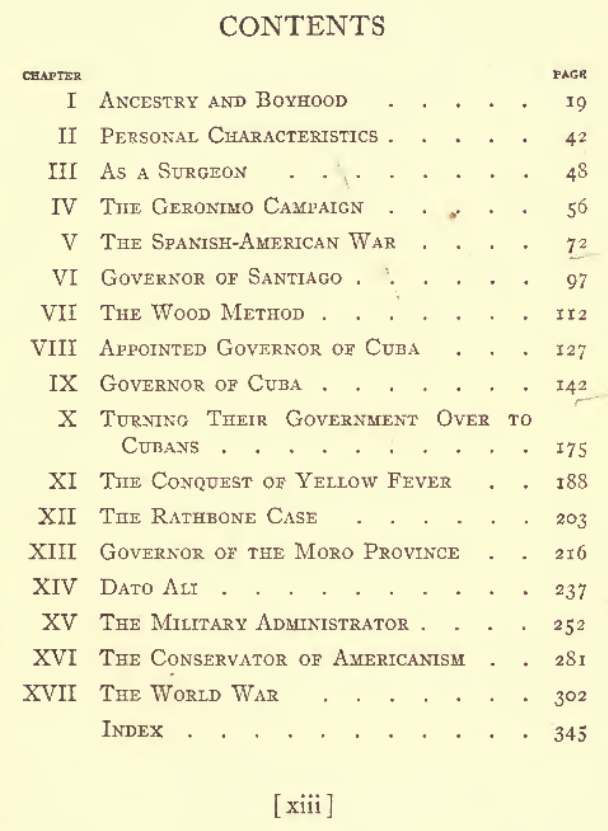
We aggregate ToC entries obtained from each pair of available submissions for each book in a collection by using two common set operators (i.e. union, and intersection). Our purpose is to evaluate the performance of an aggregation submission which only contains the common entities of two submissions or all the entities extracted by these submissions.

It is simple to apply set operators on primitive sets such as integers, floats or strings. However, a ToC submission is a derived set which consists of ToCs of books. A ToC of a book has a unique *bookid* and a list of ToC entries, that each includes three properties (i.e. title, page number, and depth level).

For each *bookid* in the corpus, we propose to aggregate ToC entries of two submissions by applying each set operator to each property which belongs to a primitive set. The difficulty in choosing appropriate properties is supported by reading-behaviours of users. When reading a book, typically, users first pay attention on the ToC to find contents and identify the corresponding pages. Hence, we consider two properties of ToC entries, title and page number, for combining submissions. Details of the set operators as well as the properties of ToC entries are introduced in the following sections.

4.1.1 The properties

It is not difficult to identify whether two page numbers are the same or not, but checking the matching of two entity titles is non-trivial because, in addition to possible OCR errors,



CONTENTS

[xiii]

Figure 4.1: ToC with single-line entries.

```
<book>
 <bookid>8CD865E34C45AA86</bookid>
   <toc-entry page="25" title="I ANCESTRY AND BOYHOOD"/>
   <toc-entry page="50" title="II PERSONAL CHARACTERISTICS"/>
   <toc-entry page="56" title="III As A SURGEON"/>
   <toc-entry page="64" title="IV THE GERONIMO CAMPAIGN . . v"/>
   <toc-entry page="84" title="V THE SPANISH- AMERICAN WAR"/>
   <toc-entry page="111" title="VI GOVERNOR OF SANTIAGO"/>
   <toc-entry page="126" title="VII THE WOOD METHOD"/>
   <toc-entry page="141" title="VIII APPOINTED GOVERNOR OF CUBA"/>
   <toc-entry page="156" title="IX GOVERNOR OF CUBA"/>
   <toc-entry page="193" title="X TURNING THEIR GOVERNMENT OVER TO CUBANS"/>
   <toc-entry page="206" title="XI THE CONQUEST OF YELLOW FEVER"/>
   <toc-entry page="221" title="XII THE RATHBONE CASE"/>
   <toc-entry page="234" title="XIII GOVERNOR OF THE MORO PROVINCE"/>
   <toc-entry page="259" title="XIV DATO ALI"/>
   <toc-entry page="274" title="XV THE MILITARY ADMINISTRATOR"/>
   <toc-entry page="303" title="XVI THE CONSERVATOR OF AMERICANISM"/>
   <toc-entry page="328" title="XVII THE WORLD WAR"/>
   <toc-entry page="371" title="INDEX"/>
   <toc-entry page="371" title="[xiii]"/>
</book>
```

Figure 4.2: Submission example of ToC in Figure 4.1.

the title of the same entry in the ToC page and the actual book content may slightly differ. However, the competition organisers defined a strategy to deal with this and compare titles. The modified Levenshtein edit distance D used in the competition on book structure extraction [22] is applied to decide whether two strings A and B are similar or not. If the distance D (computed in Equation 3.1) between string A and B is lower than 20% and the distance between their first and last five characters (or less if the string is shorter) is lower than 60%, then two strings are similar.

4.1.2 The operators

In set theory, the intersection (AND) of set A and set B is the set which contains all elements of A that also belong to B. The mathematical formulation of set intersection is provided by Equation 4.1.

$$A \cap B = \{x : x \in A \wedge x \in B\} \tag{4.1}$$

As to the union (OR), the union of set A and set B is the set of all elements in two sets, which are in A, in B, or in both A and B. In other words, this set contains all elements of set A and some elements of set B which are different from set A. Equation 4.2 formulates the union operator as follows:

$$A \cup B = \{x : x \in A \vee x \in B\} \tag{4.2}$$

For each *bookid* in the collection, we study 8 possible combinations, including *AND pages, OR pages, AND titles, OR titles, AND pages AND titles, OR pages OR titles, OR pages AND titles*, and *OR titles AND pages*. Each of them will be carefully discussed in the following two sub-sections, organising the combinations in two types, based on whether they use a single or double operators.

Before examining each combination in detail, let us take two simple example submissions shown in Figures 4.3 and 4.4 to illustrate these combinations. Assuming that submission 1 is of participant ID1 and submission 2 is of participant ID2, for the book

```xml
<?xml version="1.0" encoding="UTF-8"?>
<bs-submission participant-id="ID1">
    <book>
        <bookid>1</bookid>
        <toc-entry title="CLARISINE THE COUNTESS" page="32"/>
        <toc-entry title="THE BALLAD OF BLOODY ROCK" page="39"/>
        <toc-entry title="WHEN FIRST LOVE COMES"  page="46"/>
        <toc-entry title="THE SWALLOW" page="50"/>
    </book>
</bs-submission>
```

Figure 4.3: Example submission of participant ID_1.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<bs-submission participant-id="ID2">
    <book>
        <bookid>1</bookid>
        <toc-entry title="CLARISINE COUNTESS;" page="32"/>
        <toc-entry title="1.THE BALLAD BLOODY ROC" page="40"/>
        <toc-entry title="THE POPLARS" page="46"/>
        <toc-entry title="THE FINISHED TASK" page="53"/>
    </book>
</bs-submission>
```

Figure 4.4: Example submission of participant ID_2.

of *bookid* 1, we denote the ToC of submission 1 as ToC_1 and the ToC of submission 2 as ToC_2. It is worth clarifying that according to the distance measure D mentioned in Equation 3.1, the first title of ToC_1 'CLARISINE THE COUNTESS' is similar to that of ToC_2 'CLARISINE COUNTESS;' and the second title of ToC_1 'THE BALLAD OF BLOODY ROCK' is similar to that of ToC_2 '1.THE BALLAD BLOODY ROC'.

Our examples show four cases which can happen between book titles and book pages, including 'similar title and same page' (the first ToC entries), 'similar title and different page' (the second ToC entries), 'different title and same page' (the third ToC entries), and 'different title and different page' (the fourth ToC entries).

Single operator

In this type of combination, we only apply one set operator on one property. With the page number property, the intersection (*AND pages*) and the union of two submissions (*OR pages*) are exploited. The *AND pages* set only consists of ToC entries having the same pages of two submissions, while the *OR pages* set contains all ToC entries of the first submission and those of the second one whose pages are different from the first one.

Given our examples, the *AND pages* set contains two entities sharing the same page

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<bs-submission participant-id="ANDpages">
    <book>
        <bookid>1</bookid>
        <toc-entry page="32" title="CLARISINE THE COUNTESS" />
        <toc-entry page="46" title="WHEN FIRST LOVE COMES" />
    </book>
</bs-submission>
```

Figure 4.5: AND pages set.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<bs-submission participant-id="ORpages">
    <book>
        <bookid>1</bookid>
        <toc-entry page="32" title="CLARISINE THE COUNTESS" />
        <toc-entry page="39" title="THE BALLAD OF BLOODY ROCK" />
        <toc-entry page="46" title="WHEN FIRST LOVE COMES" />
        <toc-entry page="50" title="THE SWALLOW" />
        <toc-entry page="40" title="1.THE BALLAD BLOODY ROC" />
        <toc-entry page="53" title="THE FINISHED TASK" />
    </book>
</bs-submission>
```

Figure 4.6: OR pages set.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<bs-submission participant-id="ANDtitles">
    <book>
        <bookid>1</bookid>
        <toc-entry page="32" title="CLARISINE THE COUNTESS" />
        <toc-entry page="39" title="THE BALLAD OF BLOODY ROCK" />
    </book>
</bs-submission>
```

Figure 4.7: AND titles set.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<bs-submission participant-id="ORtitles">
    <book>
        <bookid>1</bookid>
        <toc-entry page="32" title="CLARISINE THE COUNTESS" />
        <toc-entry page="39" title="THE BALLAD OF BLOODY ROCK" />
        <toc-entry page="46" title="WHEN FIRST LOVE COMES" />
        <toc-entry page="50" title="THE SWALLOW" />
        <toc-entry page="46" title="THE POPLARS" />
        <toc-entry page="53" title="THE FINISHED TASK" />
    </book>
</bs-submission>
```

Figure 4.8: OR titles set.

numbers (the entities of pages 32, 46). Similarly, the *OR pages* set consists of six ToC entries, including four from ToC_1 (those of pages 32, 39, 46, 50) and two from ToC_2 which have different page numbers (those of pages 40, 53) from ToC_1. These combination sets of our examples are illustrated in Figures 4.5 and 4.6.

Regarding the title property, the intersection set (*AND titles*) of two submissions includes ToC entries having similar titles of two submissions; and the union (*OR titles*) of two submissions consists of all ToC entities of the first submission and those of the second one whose titles are different from the first one. The *AND titles* set and *OR titles* set of our examples are described in Figures 4.7 and 4.8.

Double operators

This sub-section describes approaches where we rely on two operators on both of properties. Firstly, there are two combinations utilising the same set operator on book properties separately: *AND pages AND titles, OR pages OR titles*. The result of the combination (*AND pages AND titles*) is the set that only contains ToC entries which have the same book pages and similar book titles. The output set of the combination (*OR pages OR titles*) includes all entries of the first submission and those of the second one having different pages and different titles from the first one.

Give our examples, the *AND pages AND titles* set only contains the first entries of ToC_1 and ToC_2 (the entries of page 32). The *OR pages OR titles* set includes all ToC_1 entries and the ToC_2 entries whose page numbers and titles are different from those in ToC_1 (only the entry of page 53 is added for that reason, while the entry of page 40 will be ignored because its title is similar to that of the entry of page 39). Figures 4.9 and 4.10 demonstrate these double combinations.

Secondly, two combinations are utilising different set operators on two properties: *OR pages AND titles*, and *OR titles AND pages*. The *OR pages AND titles* set is a subset of the *OR pages* set, since we can get it by removing the ToC entries which have different titles from the ones in the *AND titles* set. Similarly, the *OR titles AND pages* set is a subset of the *OR titles* set, obtained by deleting the ToC entries having different page numbers from those in the *AND pages* set.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<bs-submission participant-id="ANDpagesANDtitles">
    <book>
        <bookid>1</bookid>
        <toc-entry page="32" title="CLARISINE THE COUNTESS" />
    </book>
</bs-submission>
```

Figure 4.9: AND pages AND titles set.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<bs-submission participant-id="ORpagesORtitles">
    <book>
        <bookid>1</bookid>
        <toc-entry page="32" title="CLARISINE THE COUNTESS" />
        <toc-entry page="39" title="THE BALLAD OF BLOODY ROCK" />
        <toc-entry page="46" title="WHEN FIRST LOVE COMES" />
        <toc-entry page="50" title="THE SWALLOW" />
        <toc-entry page="53" title="THE FINISHED TASK" />
    </book>
</bs-submission>
```

Figure 4.10: OR pages OR titles set.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<bs-submission participant-id="ANDtitlesORpages">
    <book>
        <bookid>1</bookid>
        <toc-entry page="32" title="CLARISINE THE COUNTESS" />
        <toc-entry page="39" title="THE BALLAD OF BLOODY ROCK" />
        <toc-entry page="40" title="1.THE BALLAD BLOODY ROC" />
    </book>
</bs-submission>
```

Figure 4.11: OR pages AND titles set.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<bs-submission participant-id="ORtitlesANDpages">
    <book>
        <bookid>1</bookid>
        <toc-entry page="32" title="CLARISINE THE COUNTESS" />
        <toc-entry page="46" title="WHEN FIRST LOVE COMES" />
        <toc-entry page="46" title="THE POPLARS" />
    </book>
</bs-submission>
```

Figure 4.12: OR titles AND pages set.

In our examples, the *OR pages AND titles* set is created by removing from the *OR pages* set the ToC entries of pages (46, 50, 53) since their titles are different from the ones in the *AND titles* set. Likewise, the *OR titles AND pages* set is generated by deleting from the *OR titles* set the ToC entries of pages (39, 50, 53) that are unlike those in the *AND pages* set. These combinations are illustrated in Figures 4.11 and 4.12.

4.2 Experimental Results

4.2.1 Evaluation

As mentioned in Section 3.2.1, in order to assess the quality of the results and compare our results to the methods proposed in the competition, the official and alternative metrics of the competition are used: a title-based measure [24] and a link-based measure [19]. After checking whether ToC entries match or not, three popular metrics including Precision, Recall, F-measure are used to get an overall evaluation.

4.2.2 Results

The global performances of our systems, computed on these three competition datasets, are presented in Tables 4.1, 4.2, and 4.3. It should be noted that we combine ToC entries from each pair of any available submissions. These tables only report the best result of each combination for the compactness reason. Each table is horizontally split into 3 blocks of information. The first block evaluates the two best approaches (denoted as Best appr.) from the competition. The two next blocks correspond to the results obtained with the single operators (denoted as Single) and the double ones (denoted as Double) presented in Sections 4.1.2 and 4.1.2.

Our results show that the union operator applied to one property outperforms the sole state-of-the-art approach (MDCS) on both the title-based and the link-based evaluation measures. In terms of the link-based measure, the aggregation of two best competition approaches using *OR pages* always gets higher performance than the MDCS approach, with 4.0%, 9.9% and 6.6% improvements over the competition datasets from 2009, 2011,

Table 4.1: Performance scores over the ICDAR 2009 competition dataset.

	Method	Precision		Recall		F-measure	
		Title-based	**Link-based**	**Title-based**	**Link-based**	**Title-based**	**Link-based**
Best appr.	Books with ToC pages (MDCS)	41.33	65.90	42.83	70.30	41.51	66.40
	Books without ToC pages (XRCE)	30.28	69.20	28.36	64.80	28.47	63.80
Single	AND pages (MDCS-XRCE)	42.94	66.80	34.68	52.60	36.90	56.20
	AND titles (MDCS-XRCE)	38.51	54.20	24.38	33.70	27.40	38.00
	OR pages (MDCS-XRCE)	41.01	70.20	44.63	77.50	**41.70**	**70.40**
	OR titles (MDCS-XRCE)	36.12	59.70	46.08	76.70	39.05	63.40
Double	OR pages AND titles (MDCS-XRCE)	37.27	55.60	24.45	34.80	27.11	38.80
	OR titles AND pages (MDCS-XRCE)	35.53	56.10	36.17	54.30	34.11	51.40
	AND pages AND titles (MDCS-XRCE)	38.44	54.30	23.24	31.70	26.54	36.50
	OR pages OR titles (MDCS-XRCE)	41.75	70.70	44.59	76.20	**42.11**	**70.10**

Table 4.2: Performance scores over the ICDAR 2011 competition dataset.

	Method	Precision		Recall		F-measure	
		Title-based	**Link-based**	**Title-based**	**Link-based**	**Title-based**	**Link-based**
Best appr.	Books with ToC pages (MDCS)	40.40	64.50	43.17	70.20	40.75	65.10
	Books without ToC pages (XRCE)	27.39	79.30	18.69	52.50	20.38	57.60
Single	AND pages (MDCS-Nankai)	39.72	64.10	34.14	54.40	34.96	55.60
	AND titles (MDCS-Nankai)	38.48	58.90	27.60	39.80	30.00	43.80
	OR pages (MDCS-XRCE)	43.52	75.00	48.82	83.20	**44.50**	**75.00**
	OR titles (MDCS-XRCE)	39.55	63.50	51.86	79.60	42.25	65.00
Double	OR pages AND titles (MDCS-Nankai)	36.14	56.60	27.88	41.30	29.44	44.00
	OR titles AND pages (MDCS-Nankai)	35.59	56.30	35.37	55.60	33.79	52.30
	AND pages AND titles (MDCS-Nankai)	37.86	58.10	25.76	36.80	28.53	41.40
	OR pages OR titles (MDCS-XRCE)	43.96	74.90	47.37	79.60	**43.64**	**72.50**

Table 4.3: Performance scores over the ICDAR 2013 competition dataset.

	Method	Precision		Recall		F-measure	
		Title-based	**Link-based**	**Title-based**	**Link-based**	**Title-based**	**Link-based**
Best appr.	Book with ToC pages (MDCS)	42.77	64.90	45.92	71.50	43.61	66.60
	Book without ToC pages (Innsbruck)	33.63	75.70	32.14	68.90	31.34	67.20
Single	AND pages (MDCS-Nankai)	43.87	65.50	37.49	54.80	38.85	56.50
	AND titles (MDCS-Nankai)	42.12	60.50	30.12	40.30	32.94	44.50
	OR pages (MDCS-Innsbruck)	41.97	69.50	49.69	85.40	**44.07**	**73.20**
	OR titles (MDCS-Innsbruck)	38.36	61.50	50.45	83.40	42.04	67.40
Double	OR pages AND titles (MDCS-Nankai)	39.67	59.60	30.21	41.80	32.07	44.70
	OR titles AND pages (MDCS-Nankai)	38.80	58.40	38.95	56.70	37.29	53.80
	AND pages AND titles (MDCS-Nankai)	42.49	60.00	28.27	37.00	31.69	41.80
	OR pages OR titles (MDCS-Innsbruck)	42.86	69.70	49.42	82.40	**44.53**	**72.20**

and 2013, respectively.

In terms of the title-based measure, the *OR pages* aggregation is 3.75% higher than MDCS for the 2011 competition. With the 2009 and 2013 datasets, the *OR pages OR titles* aggregation achieves better results than MDCS, by 0.6% and 0.92%, respectively.

The union operator outperforms other set operators because it combines the best of two worlds, by integrating results from a) methods that are good at extracting ToC entries from books with ToC pages, and b) methods that are good over books without ToC pages. This confirms our initial hypothesis that both types of approaches are complementary. Indeed, the main F-measure improvement is due to strong recall improvement while precision remains stable.

Significance of our results. To determine whether our results are statistically conclusive, we compute the student's t-test to compare the distributions of our best combinations to the best-performing methods over each of the three competition datasets. These detail p-values are demonstrated in Table 4.4.

Regarding linked-based measure, this shows clear significance over all competition subsets ($p<0.001$), demonstrating the added-value of our approach over the state of the

art. Concerning the title-based measure, statistical significance is obtained for the 2011 datasets ($p<0.001$), but not for the 2009 and 2013 datasets.

As described in Section 3.2.1, the title-based measure checks the similarity between titles before other properties. We think that there are more books (without printed ToC pages) having long and degraded titles in the 2009 and 2013 datasets than those in the 2011 dataset, which cause low-quality extracted titles and reduces the performance of ToC approaches that focus on the analysis of the full content. As a consequence, our aggregation approach cannot reach similar performance improvements as it does with the 2011 dataset.

The official results on the title-based measure of the competition confirm our assumption. While the performance of the method based on ToC pages (MDCS) remains stable, that of the two approaches considering the entire book content in our best combination (XRCE, Innsbruck) is reduced dramatically on the 2009 and 2013 datasets (XRCE reaches only 7.81% on the 2009 dataset, and Innsbruck 8.2% on the 2013 dataset, while XRCE reaches 26.32% on the 2011 dataset).

Table 4.4: Student's t-test over three competition datasets.

Dataset	**P-value**	
	Title-based	**Link-based**
2009	0.1567	1.225E-10
2011	1.109E-05	1.835E-14
2013	0.0018	1.159E-26

4.3 Conclusions

This chapter presents our aggregation approach using two set operators on two properties of ToC entries in order to combine the output of top-performing methods in book structure extraction. By combining the output of systems that are focused on the detection and analysis of ToC pages and systems that are focusing on inner book contents, it manages to perform statistically significant improvement over the state of the art in extraction of

table of contents. Our experimental results indeed demonstrate that the union operator applied on ToC entries' properties performs better than the top-performing methods for both title-based and link-based evaluation.

We have now presented a way to ease information access by providing logical structure to enter and browse documents. The remainder of this dissertation will focus on the amelioration of the textual contents of documents.

CHAPTER 5

Comparison of OCR errors and human misspellings

OCR errors share some common features with spelling errors, but, OCR errors have their own special characteristics as they are created by different processes than spelling errors. Various characteristics of OCR errors on popular public datasets are analysed and compared with misspellings in order to design better post-OCR approaches. This chapter gives an overview of OCR process and the results of the analyses.

5.1 OCR process

Historical digital texts hold some special characteristics and differ from modern electronic text. Most of problems of OCRed texts involve in this conversion process, which consists of four standard steps: scanning, zoning, segmentation, and classification [95].

- Scanning is the first step of digitisation process, which produces digital version of paper-based documents. The outcome of this step heavily depends on the degradation level of the original document, scanner, or a way of scanning.
- Zoning automatically orders the text regions in the documents. Zoning errors greatly affect the word order of the scanned material and produce an incoherent document.

Some of real-word errors of OCRed text are zoning errors. In fact, wrongly ordering one sentence might cause all correctly recognised words of that sentence to be real-word errors. Zoning errors are very hard to be corrected by post-processing using NLP techniques.

- Segmentation breaks zones into words, and decomposes words into characters. Incorrect segmentation results in errors involving incorrect split and concatenation of words (e.g., 'sea' vs. 'se a', 'blue sea' vs. 'bluesea'), or those related to multiple substitutions (m characters are wrongly recognised as n characters, e.g '**m**ain' vs. '**lii**ain', '**cl**ient' vs. '**d**ient').

- Classification process classifies characters into their corresponding ASCII characters. OCR devices recognise characters primarily by their shapes. With some noises, it is easy to mis-recognise symbols of similar shapes. In other words, a correct character is replaced by an invalid character, e.g., '**c**ore' vs. '**e**ore', '**i**n' vs. '**l**n').

5.2 Analysed datasets

Four analysed datasets are public collections of historical documents obtained from four libraries. Two first datasets come from the first competition on post-OCR text correction [8], including Monograph, Periodical. Their details are mentioned in Section 3.1.2.

Two others are Overproof evaluation datasets [27]. The first one (denoted as OverNLA) consists of 159 medium-length news articles with at least 85% correct lines, which are extracted from one of the longest-running titles in the National Library of Australia's Trove newspaper archive - The Sydney Morning Herald, 1842-1954. Its corresponding GT was additionally corrected by Evershed *et al.* [27] after crowd sourcing corrections [34].

The second one (denoted as OverLC) contains 49 medium-length news articles randomly selected from 5 titles of the Library of Congress Chronicling America newspaper archive. The corresponding GT of OverNC was manually corrected by Evershed *et al.* [27]. Both of the Overproof datasets are noisier than the competition ones with totally 208 files.

Table 5.1: Details description of four datasets.

Sources	Types	Years	WER	Sizes	Files
Monograph	monograph	1862-1911	9%	4.2M	747
Periodical	periodical	1744-1894	16%	1.8M	66
OverNLA	news	1842-1954	25%	0.3M	159
OverLC	news	1871-1921	27%	0.1M	49

These OCRed documents are processed by ABBYY FineReader[1], which is the state-of-the-art commercial OCR system.

The four datasets thus include OCRed texts of past documents from popular libraries (National Library of France, British Library, National Library of Australia, Library of Congress Chronicling America). The selected documents are characterised by varying levels of degradation under independent conservation and originate from a relatively wide time range spanning from 1744 to 1954. In view of these, altogether the datasets are representative for historical OCRed texts with typical OCR errors. The details of sources, types, years, word error rates (WER)[2], sizes and the file counts of all the four datasets are listed in Table 5.1.

5.3 OCR errors vs. human misspellings

In the following sections, we present five main types of analyses conducted on all the datasets. Particularly, edit operation types and edit distance are considered. In addition, we concentrate not only on word lengths but also on OCRed token lengths. Moreover, positions of incorrect characters and real-word vs. non-word errors are analysed. Problems related to the wrong deletion/insertion of white spaces (word boundaries) are also examined.

[1] https://www.abbyy.com
[2] WER is derived from the Levenshtein distance, working at word level.

5.3.1 Edit operations

In this section, we discuss edit operation types, standard/non-standard substitution mappings (denoted as standard/non-standard mappings), edit distance and string similarity based on LCS.

Edit operation types

In order to transform token A to token B, four basic edit operation types can be performed: deletion, insertion, substitution, and transposition [14]. Prior works [51, 66, 97] indicated that transposition is common in misspellings but rarely occurs in OCR errors. We then only consider the first three types.

Figure 5.1 shows the percentages of single modification error types (deletion, insertion and substitution denoted as *del*, *ins* and *sub*, respectively) and ones of their possible combinations (*del+ins, del+sub, ins+sub, del+ins+sub*) in all the four datasets.

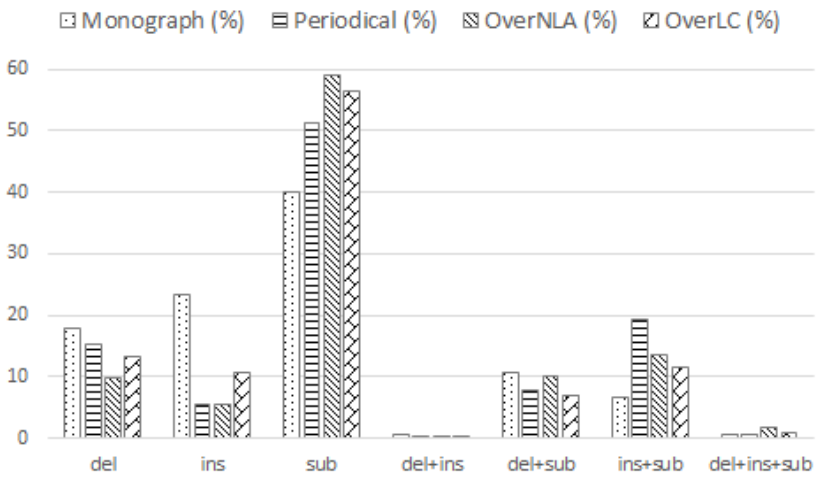


Figure 5.1: Error rates based on edit operation types.

Among single edit operation types, the average percentage of substitution (51.6%) is much higher than that of two others. Furthermore, the total percentage of three single edit operation types is about 77.02%, thus higher than that of their combinations. It leads to the conclusion that post-OCR techniques can correct most of errors by just concentrating on a single modification type.

As to the combinations of edit operation types, deletion and insertion rarely occur together. In fact, the combinations of deletion and insertion have very small occurrence

rate being 0.24% (*del+ins*) and 1% (*del+ins+sub*). Post-OCR approaches could then in our opinion pay less attention on the combinations in candidate generation.

Moreover, the average rate of OCR errors involving substitution, insertion, deletion is approximately 5:1:1, which is an useful information for some post-OCR approaches (e.g., [27, 60, 75]) to decide the number of substitution/insertion character candidates for each OCRed character position in candidate generation. If the rate is too small, no correct candidates can be suggested. Otherwise, many incorrect candidates are created negatively affecting the candidate ranking process.

Standard mapping

Secondly, we consider standard and non-standard mappings, which are results of wrongly zoning text regions. While misspellings often have standard mapping 1:1 (e.g., 'hear' vs. 'jear'), OCR errors contain not only standard mappings 1:1 but also non-standard mappings, such as n:1 and 1:n (e.g., 'link' vs. 'hnk', 'link' vs. 'liiik').

The standard mapping 1:1 of our datasets is illustrated in Tables 5.2 and 5.3. In these tables, we compute the percentage of appearance frequency of each GT character being recognised as an OCRed character for each dataset. Let us call this percentage as mapping percentage. In order to make the tables compact, we only show OCRed characters whose mapping percentages are equal or more than 0.1%. Other cases whose mapping percentages are less than 0.1% are denoted as @. Because one GT character can be recognised as one or n OCRed characters, so other cases (@) include OCRed characters in 1:1 mappings and 1:n mappings. For example, the percentages of frequency of character 'b' in Periodical being recognised as 'b', 'h' and other characters are 96.7%, 1.6% and 1.7%, respectively.

Tables 5.2 and 5.3 indicate that the characters with the highest and lowest recognition accuracy are 't', 'z' with 98.53% and 88%, respectively. Moreover, the statistics also reveal that characters sharing similar shapes are easily confused, such as 'b' vs. 'h'; 'c' vs. {'o', 'e'}; 'e' vs. {'o', 'c'}.

This standard mapping is used to create character confusion matrix which is one of the most important sources to generate and rank candidates. It is obvious that the more similar frequent error patterns between a training part and a testing part of the used datasets are, the higher the probability of the correct candidates are generated. However,

Table 5.2: Percentages of standard mapping 1:1 (part 1). One GT character is substituted by one OCRed character. Only values higher than 0.1% are shown, other characters (including sequences of more than one character) are denoted as @.

GT Char	Monograph	Periodical
a	{a: 99.5, @: 0.5}	{a: 97.5, u: 0.4, n: 0.2, e: 0.2, i: 0.2, @: 1.5}
b	{b: 98.7, h: 0.8, @: 0.5}	{b: 96.7, h: 1.6, @: 1.7}
c	{c: 97.0, o: 2.0, e: 0.6, @: 0.4}	{c: 96.2, e: 1.2, o: 1.0, @: 1.6}
d	{d: 99.7, @: 0.3}	{d: 98.4, l: 0.2, i: 0.2, @: 1.2}
e	{e: 98.7, o: 0.2, @: 1.1}	{e: 96.9, o: 0.6, c: 0.5, a: 0.2, s: 0.2, @: 1.6}
f	{f: 98.2, @: 1.8}	{f: 96.2, t: 1.2, l: 0.9, i: 0.4, @: 1.3}
g	{g: 99.6, @: 0.4}	{g: 98.3, @: 1.7}
h	{h: 99.1, b: 0.4, @: 0.5}	{h: 95.2, b: 1.7, i: 0.4, n: 0.2, @: 2.5}
i	{i: 99.1, @: 0.9}	{i: 97.6, l: 0.6, t: 0.2, @: 1.6}
j	{j: 99.7, @: 0.3}	{j: 97.4, i: 0.3, l: 0.3, c: 0.2, @: 1.8}
k	{k: 99.5, @: 0.5}	{k: 98.6, t: 0.2, @: 1.2}
l	{l: 95.6, i: 0.8, d: 0.2, @: 3.4}	{l: 96.9, i: 0.8, t: 0.2, @: 2.1}
m	{m: 99.1, @: 0.9}	{m: 97.4, n: 0.5, i: 0.2, @: 1.9}
n	{n: 99.1, u: 0.2, @: 0.7}	{n: 96.4, u: 1.2, a: 0.3, m: 0.2, o: 0.2, i: 0.2, @: 1.5}
o	{o: 99.4, @: 0.6}	{o: 97.9, e: 0.5, a: 0.2, @: 1.4}
p	{p: 99.8, @: 0.2}	{p: 98.7, n: 0.2, @: 1.1}
q	{q: 99.4, @: 0.6}	{q: 97.7, o: 0.2, i: 0.2, j: 0.2, @: 1.7}
r	{r: 99.4, @: 0.6}	{r: 98.5, i: 0.3, t: 0.2, @: 1.0}
s	{s: 98.8, a: 0.5, f: 0.3, @: 0.4}	{s: 94.2, a: 0.8, e: 0.7, t: 0.3, i: 0.3, @: 3.7}
t	{t: 99.7, @: 0.3}	{t: 98.7, i: 0.2, l: 0.2, @: 0.9}
u	{u: 99.2, n: 0.2, @: 0.6}	{u: 96.6, n: 1.1, a: 0.7, o: 0.3, i: 0.2, @: 1.1}
v	{v: 99.6, @: 0.4}	{v: 97.9, r: 0.7, y: 0.2, @: 1.2}
w	{w: 99.6, @: 0.4}	{w: 98.7, @: 1.3}
x	{x: 99.0, @: 1.0}	{x: 97.4, i: 0.8, s: 0.2, r: 0.2, t: 0.2, @: 1.2}
y	{y: 99.5, @: 0.5}	{y: 98.0, v: 1.1, @: 0.9}
z	{z: 99.2, s: 0.5, @: 0.3}	{z: 86.0, s: 2.5, x: 1.6, r: 1.2, i: 1.1, a: 0.9, g: 0.3, t: 0.3, v: 0.3, c: 0.2, b: 0.2, e: 0.2, k: 0.2, l: 0.2, o: 0.2, n: 0.2, u: 0.2, @: 4.2}

OCR errors can vary from OCR engines, layouts as well as degradation levels of documents, and etc. Therefore, some very frequent characters along with their highly possible misrecognition (e.g., 'e' vs. 'o', 'j' vs. 'i') may not occur in the large training part and only appear in the small testing part. In such cases, it is impossible to generate valid candidates for unseen error patterns of the testing part.

Non-standard mappings

Besides the standard mapping 1:1, OCR errors are also subject to more complex mappings [42, 53]. Different from past related work [1, 27, 42, 87, 88, 95], our study provides

Table 5.3: Percentages of standard mapping 1:1 (part 2). One GT character is substituted by one OCRed character. Only values higher than 0.1% are shown, other characters (including sequences of more than one character) are denoted as @.

GT Char	Overproof NLA	Overproof LC
a	{a: 92.7, n: 2.1, i: 1.1, u: 1.0, o: 0.3, m: 0.2, @: 2.6}	{a: 92.8, n: 2.7, u: 0.8, i: 0.5, m: 0.3, o: 0.2, @: 2.7}
b	{b: 96.2, h: 1.7, l: 0.5, t: 0.3, @: 1.3}	{b: 93.9, h: 1.8, l: 0.5, n: 0.4, i: 0.3, t: 0.3, o: 0.2, m: 0.2, @: 2.4}
c	{c: 93.9, e: 1.7, o: 1.5, r: 0.4, t: 0.2, i: 0.2, @: 2.1}	{c: 92.2, o: 3.1, e: 1.6, u: 0.3, s: 0.3, n: 0.2, a: 0.2, r: 0.2, t: 0.2, @: 1.7}
d	{d: 97.1, a: 0.4, l: 0.2, i: 0.2, @: 2.1}	{d: 96.8, l: 0.5, i: 0.3, u: 0.3, @: 2.1}
e	{e: 86.1, o: 9.2, c: 1.6, i: 0.3, a: 0.2, @: 2.6}	{e: 80.8, o: 14.8, c: 0.9, u: 0.4, i: 0.3, r: 0.2, n: 0.2, @: 2.4}
f	{f: 94.3, l: 1.5, t: 1.0, i: 0.9, @: 2.3}	{f: 94.1, l: 1.8, t: 1.4, i: 0.6, @: 2.1}
g	{g: 93.4, c: 0.4, p: 0.4, r: 0.4, e: 0.3, s: 0.3, i: 0.3, u: 0.3, t: 0.2, f: 0.2, @: 3.8}	{g: 95.2, j: 0.3, i: 0.3, c: 0.2, e: 0.2, @: 3.8}
h	{h: 95.1, b: 1.1, l: 0.8, i: 0.7, n: 0.2, @: 2.1}	{h: 95.7, l: 1.0, i: 0.6, b: 0.5, n: 0.3, @: 1.9}
i	{i: 90.7, l: 3.3, m: 0.4, t: 0.3, u: 0.2, n: 0.2, @: 4.9}	{i: 94.0, l: 1.6, @: 4.4}
j	{j: 85.0, i: 1.5, l: 0.4, t: 0.4, @: 12.7}	{j: 92.7, @: 7.3}
k	{k: 95.6, l: 1.0, i: 0.3, h: 0.2, t: 0.2, @: 2.7}	{k: 97.5, a: 0.2, i: 0.2, h: 0.2, @: 1.9}
l	{l: 96.2, i: 0.8, @: 3.0}	{l: 96.8, i: 0.7, @: 2.5}
m	{m: 94.3, n: 1.6, i: 0.8, r: 0.5, u: 0.2, @: 2.6}	{m: 93.9, n: 1.3, i: 1.1, u: 0.3, r: 0.2, t: 0.2, @: 3.0}
n	{n: 96.2, u: 1.0, i: 0.4, m: 0.3, a: 0.2, @: 1.9}	{n: 92.6, u: 4.0, i: 0.8, m: 0.2, a: 0.2, @: 2.2}
o	{o: 98.0, n: 0.2, i: 0.2, @: 1.6}	{o: 97.2, n: 0.3, u: 0.3, e: 0.3, @: 1.9}
p	{p: 97.9, n: 0.7, i: 0.2, r: 0.2, @: 1.0}	{p: 96.8, n: 0.5, j: 0.3, o: 0.2, i: 0.2, r: 0.2, @: 1.8}
q	{q: 97.3, a: 1.5, o: 0.9, @: 0.3}	{q: 90.7, i: 3.3, m: 2.9, @: 3.1}
r	{r: 93.4, i: 3.3, l: 0.4, n: 0.3, t: 0.2, @: 2.4}	{r: 98.1, i: 0.2, t: 0.2, @: 1.5}
s	{s: 91.7, a: 1.2, i: 0.5, e: 0.3, n: 0.2, b: 0.2, t: 0.2, @: 5.7}	{s: 90.8, t: 0.6, i: 0.5, e: 0.5, a: 0.4, n: 0.3, f: 0.3, u: 0.2, l: 0.2, o: 0.2, h: 0.2, @: 5.8}
t	{t: 97.7, l: 0.7, i: 0.2, @: 1.4}	{t: 98.0, l: 0.6, i: 0.2, @: 1.2}
u	{u: 96.1, n: 1.0, i: 0.6, a: 0.3, m: 0.2, @: 1.8}	{u: 96.1, i: 0.7, a: 0.5, o: 0.2, n: 0.2, j: 0.2, @: 2.1}
v	{v: 92.2, i: 0.8, r: 0.5, y: 0.3, n: 0.3, t: 0.2, @: 5.7}	{v: 97.7, i: 0.3, r: 0.3, m: 0.3, @: 1.4}
w	{w: 92.8, v: 1.1, n: 0.5, y: 0.3, m: 0.2, i: 0.2, @: 4.9}	{w: 98.1, v: 0.2, o: 0.2, @: 1.5}
x	{x: 94.6, v: 0.9, i: 0.7, t: 0.6, o: 0.4, n: 0.3, s: 0.2, @: 2.3}	{x: 97.1, g: 1.2, t: 0.6, @: 1.1}
y	{y: 87.9, j: 3.4, v: 3.1, i: 0.4, r: 0.3, s: 0.2, @: 4.7}	{y: 96.9, v: 1.3, j: 0.3, f: 0.2, @: 1.3}
z	{z: 68.7, r: 6.2, s: 1.9, b: 1.6, n: 1.6, m: 1.5, y: 1.5, i: 0.8, u: 0.7, l: 0.5, @: 15.0}	{z: 98.1, @: 1.9}

the detailed statistics on the four popular datasets instead of only giving examples of non-standard mappings.

The first point is 1:n mapping, in which one GT character is recognised as n OCRed characters (e.g., 'main' vs. 'rnain'). The mapping percentages of frequency of each GT character being recognised as n OCRed characters are calculated for each dataset in Tables 5.4 and 5.5. With the same compactness reason as in Tables 5.2 and 5.3, these tables only contain n OCRed characters whose mapping rates are equal or higher than 0.01%. Tables 5.4 and 5.5 clarify 1:n mapping of @ in Tables 5.2 and 5.3.

For instance, the percentage of frequency of character 'b' in Periodical being recognised as 'li', 'ti', 'th', 'l.' are 0.19%, 0.02%, 0.02%, 0.02%, respectively. The 1:n mapping statistics indicate that there are some frequent patterns along with their average percents, such as {'b': {'li':0.05, 'h':0.03}; 'd': {'il':0.07, 'cl':0.03}; 'h': {'li':0.34, 'ii':0.06}}.

The second point is n:1 mapping, in which n GT characters are recognised as one OCRed character (e.g., 'main' vs. 'mam'). The frequency rates of n GT characters being recognised as one OCRed character are computed on four datasets showed in Tables 5.6 and 5.7. These tables only show GT character ngrams whose mapping percentages are not less than 0.01% and which appear at least 10% of maximum frequency of their ngrams. Different from the above tables, in Tables 5.6 and 5.7, we group percentages according to OCRed characters because it is inefficient to show many GT character ngrams in the first column.

For example, in Monograph dataset, the percentage of appearance frequency of GT character bigram 'li' being recognised as 'b' is 0.03%. Based on the statistics of n:1 mappings, some common patterns with their average rates emerge, such as { 'b': {'si':0.05, 'li':0.04}; 'd': {'il':0.7, 'll':0.12}; 'h': {'li':0.16, 'ly':0.1}}.

Our observations on these mappings support a conclusion that some characters 'b', 'd', 'h', 'm', 'n' are easily recognised as 'li', {'il', 'cl'}, 'li', {'rn', 'in'}, {'ri', 'ii'}, respectively. In opposite way, 'li', {'il', 'cl'}, 'li', {'rn', 'in'}, {'ri', 'ii'} can be recognised as 'b', 'd', 'h', 'm', 'n', respectively. These kinds of mappings also play important roles in generating and ranking candidates.

It should be noted that the statistics of these non-standard mappings are extracted

Table 5.4: Percentages of non-standard mapping 1:n (part 1) Only values higher than 0.01% are shown. For each GT character, percentages shown for each dataset are parts of corresponding percents of @ in Tables 5.2 and 5.3.

GT Char	Monograph	Periodical
b		{li: 0.19, ti: 0.02, th: 0.02, l.: 0.02}
c		{See: 0.03, foe: 0.02}
h	{li: 0.07}	{li: 0.78, ii: 0.23, il: 0.07, ri: 0.05, ir: 0.04}
j		{.t: 0.08, i.: 0.08}
k		{lc: 0.06, fc: 0.03}
m	{rn: 0.36, ni: 0.04, in: 0.03}	{in: 0.17, ra: 0.12, rn: 0.09, ni: 0.08, tn: 0.06}
n		{r.: 0.07, ri: 0.03, ii: 0.03}
p		{ji: 0.03}
q	{cp: 0.03}	{tj: 0.1, .l: 0.05, ri: 0.05, -'t: 0.05}
u		{ti: 0.04, ii: 0.02, tt: 0.02, it: 0.02}
w		{vv: 0.03, vr: 0.02, sr: 0.02}
x	{'∼: 0.02}	{ts: 0.03}
z		{sa: 0.16, .i: 0.16, r.: 0.16, id: 0.16, ti: 0.16}

from aligned OCR and their corresponding GT. Although we make a full use of OCRed text along with its corresponding GT, there are still some unavoidable noises in our statistics due to the lack of character recognition confidences from OCR engines.

Edit distances

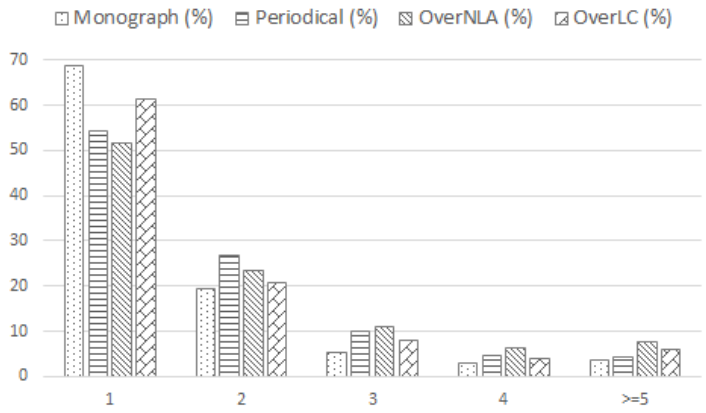


Figure 5.2: Error rates based on edit distances.

In case of edit distances, the survey on spelling errors [54] pointed out two main types: *single-error* tokens (one edit distance) and *multi-error* tokens (higher edit distances). It is obvious that the smaller edit distance an error has, the easier the correction task is.

Table 5.5: Percentages of non-standard mapping 1:n (part 2). Only values higher than 0.01% are shown. For each GT character, percentages shown for each dataset are parts of corresponding percents of @ in Tables 5.2 and 5.3.

GT Char	Overproof NLA	Overproof NC
a	{ii: 0.05, in: 0.03, -i: 0.02, .i: 0.02}	{ii: 0.21, it: 0.05, in: 0.05, .i: 0.05, iu: 0.03}
b		{'h: 0.11, ili: 0.04}
c	{t-: 0.05, e-: 0.04, le: 0.03, i': 0.02, .e: 0.02}	{Hle: 0.07, 'C: 0.02, iriw: 0.02}
d	{il: 0.15, tl: 0.05, cl: 0.03, ri: 0.03, t4: 0.02}	{il: 0.15, cl: 0.07, rt: 0.06, tl: 0.05, nl: 0.04}
e	{io: 0.04, lc: 0.02, ic: 0.02}	{io: 0.14, iu: 0.03, no: 0.02, oo: 0.02, n;: 0.02}
f	{'l: 0.03, l': 0.02}	{l': 0.1, l": 0.05, he: 0.02}
g	{iR: 0.09, a-: 0.08, tr: 0.08, fr: 0.07, er: 0.06}	{i": 0.33, e:: 0.21, uu: 0.14, (;: 0.14, ..:.-: 0.13}
h	{li: 0.3, il: 0.06, ll: 0.05, ji: 0.02, i(.li: 0.02}	{li: 0.21, di: 0.04, Ii: 0.04, 'li: 0.04, ti: 0.03}
i	{vl: 0.03, ll: 0.02}	{ll: 0.04, l': 0.02, '.: 0.02}
k	{lr: 0.12, l;: 0.12, lt: 0.08, fc: 0.06, ',: 0.04}	
l	{ii: 0.02, uit: 0.02, ->: 0.02}	{'.: 0.05}
m	{in: 0.37, rn: 0.29, ni: 0.13, ra: 0.09, tn: 0.08}	{in: 0.65, ni: 0.48, ro: 0.16, rn: 0.15, tn: 0.11}
n	{ii: 0.11, ti: 0.03}	{ii: 0.12, ti: 0.11, ri: 0.08, t.: 0.06, iti: 0.03}
o		{in: 0.03, .i: 0.02, i.: 0.02}
p	{ii: 0.05, iv: 0.03, .i: 0.02}	{fi: 0.1, iiiti: 0.07, ii: 0.03}
q	{.v: 0.03}	
r	{ii: 0.02, i-: 0.02, li: 0.02, i': 0.02}	{ii: 0.04, t': 0.02}
s	{la: 0.03, t,: 0.02, iB: 0.02}	{.-: 0.04, c-: 0.04, nl': 0.04, i': 0.04, .": 0.03}
t		{ln: 0.03, Uo: 0.02}
u	{ii: 0.19, ti: 0.08, li: 0.04, tii: 0.03, i.: 0.02}	{ti: 0.11, ii: 0.1, tl: 0.06, ri: 0.05, i': 0.04}
v	{Ham: 0.09, %': 0.05, s': 0.04, «.: 0.02}	{*: 0.24}
w	{vv: 0.44, tv: 0.15, ir: 0.07, *v: 0.05, v»: 0.05}	{st: 0.11, fiH: 0.07}
x	{.i: 0.39}	
y	{nj: 0.07, i,: 0.05, ij: 0.05,)*: 0.05, 'j: 0.04}	{tv: 0.04, iiv: 0.04, ino: 0.04, IV: 0.04}

Table 5.6: Percentages of non-standard mapping n:1 (part 1). n GT characters are substituted by one OCRed character. Only OCRed characters are results of n GT characters mis-recognition are listed, and only values higher than or equal 0.05% are shown. Even though this table shows n:1 mapping, the presentation is in a reverse way (1:n) in order to save space.

OCR Char	Monograph	Periodical
a		{ste: 0.07, ur: 0.05, pe: 0.05}
b		{li: 0.08}
c	{pe: 0.05}	
d	{il: 2.62, ll: 0.45}	{il: 0.1, el: 0.06}
e	{ho: 0.04}	
h	{li: 0.28}	{li: 0.08, la: 0.06}
i	{wa: 0.02}	
m	{in: 0.07}	{us: 0.15, ns: 0.11, un: 0.1, in: 0.1, res: 0.08, nt: 0.08, ur: 0.05, ss: 0.05}
n	{ri: 0.22}	{ri: 0.24, rs: 0.14, us: 0.05, wh: 0.05}
s		{ear: 0.06}
u		{ss: 0.12, as: 0.1, ta: 0.08, nde: 0.06, ie: 0.06, }
w		{ss: 0.19, ec: 0.05, se: 0.05}

Percentages of errors based on edit distances of our datasets in Figure 5.2 show that most of OCR errors are *single-error* tokens with approximately 58.92% occurrences. That rate is smaller than the rate of *single-error* typos in misspelled words (74.5% on average) [54]. In terms of *multi-error* tokens, most of them are of edit distance 2 (on average 22.57%). These statistics reveal that OCR post-processing approaches can mainly concentrate on edit distances 1 and 2 (with total 81.49% on average) at beginning steps. Relying on these statistics, the edit distance threshold can be set at 2 for removing many irrelevant candidates.

String similarity based on Longest Common Sequence (LCS)

LCS is another way to measure the similarity between two strings. Islam *et al.* [39] proposed two variations of LCS, including Normalised Longest Common Subsequence (NLCS) and Normalised Maximal Consecutive Longest Common Subsequence (NMCLCS).

NLCS considers lengths of two related strings w_c and w_e, as follows:

$$NLCS(w_c, w_e) = \frac{length(LCS(w_c, w_e))^2}{length(w_c) * length(w_e)} \tag{5.1}$$

Table 5.7: Percentages of non-standard mapping n:1 (part 2). n GT characters are substituted by one OCRed character. Only OCRed characters are results of n GT characters mis-recognition are listed, and only values higher than or equal 0.05% are shown. Even though this table shows n:1 mapping, the presentation is in a reverse way (1:n) in order to save space.

OCR Char	Overproof NLA	Overproof NC
a	{s.: 1.69, s,: 0.36, ce: 0.09, ut: 0.07}	{si: 0.55, he: 0.07}
b	{hi: 0.07, li: 0.06, is: 0.05}	{si: 0.21}
c	{e,: 0.47, le: 0.13, ee: 0.12, ne: 0.12}	{ess: 0.36, es: 0.25, ee: 0.18, se: 0.1}
d	{il: 0.08}	{si: 0.24, on: 0.08}
e	{s.: 0.12, ic: 0.07}	{can: 1.31, ic: 0.57, ac: 0.43, his: 0.27}
f	{ta: 0.13}	
h	{ly: 0.38, li: 0.26}	{ld: 0.12}
i	{r.: 3.46, s.: 0.39, nce: 0.38, al: 0.05, as: 0.05}	{ta: 0.26, on: 0.05}
j	{or: 0.05}	
k	{ly: 0.22}	
l	{ni: 0.26, ri: 0.19, si: 0.18, di: 0.14, r.: 0.07}	{ir: 1.02, ai: 0.6, ot: 0.21, in: 0.12, re: 0.06}
m	{n,: 0.43, ur: 0.3, ni: 0.29, in: 0.29, ia: 0.25, ns: 0.16, ai: 0.15, ree: 0.12, as: 0.07, rs: 0.05, }	{ld: 0.55, ns: 0.42, ll: 0.21, nt: 0.11, es: 0.09, ee: 0.08, on: 0.05}
n	{ri: 1.61, ry: 0.54, ia: 0.54, am: 0.23, ma: 0.13, ra: 0.13, s.: 0.12, s,: 0.12, st: 0.12, ll: 0.11, ti: 0.1, ay: 0.08, ar: 0.05, at: 0.05}	{rs: 0.99, ss: 0.47, om: 0.41, as: 0.28, ar: 0.05}
o	{e,: 0.75, ic: 0.35, ie: 0.27, nc: 0.23, ne: 0.13, me: 0.12, es: 0.09, ive: 0.07, he: 0.07}	{ee: 0.32, se: 0.3, ll: 0.15, es: 0.14, en: 0.08}
p	{ve: 0.12, s,: 0.12, ing: 0.1}	
q	{s.: 0.27}	{ar: 0.1}
r	{ac: 0.23, ss: 0.12, ee: 0.09}	{me: 0.21, en: 0.18}
s	{e,: 0.14, ng: 0.07}	{tor: 1.99, an: 0.08}
t	{ine: 0.6, e,: 0.29, one: 0.22, s.: 0.16, le: 0.13}	
u	{is: 0.37, ri: 0.28, so: 0.27, ia: 0.25, ll: 0.25, rs: 0.19, as: 0.19, hi: 0.16, ti: 0.13, il: 0.12, ha: 0.11, le: 0.1, in: 0.07, ra: 0.06, st: 0.06, li: 0.06, ee: 0.05, }	{ns: 0.7, na: 0.65, fo: 0.24, st: 0.2, an: 0.15, ea: 0.14, te: 0.06, is: 0.06}
w		{ear: 1.08, se: 0.29}

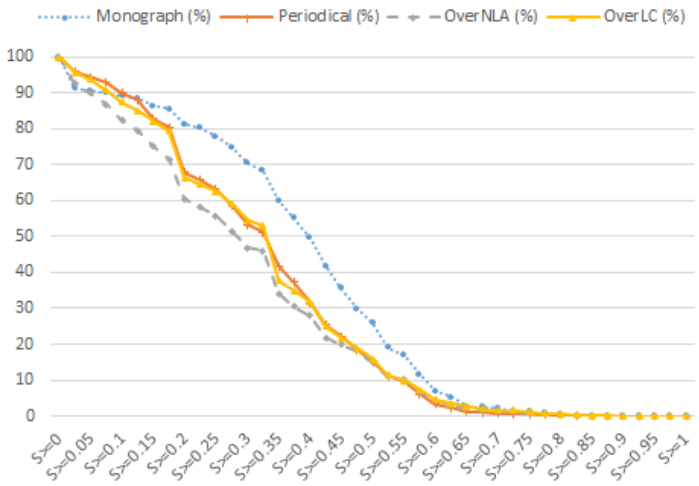


Figure 5.3: Error rates based on the LCS similarity S.

There are three variations of MCLCS (Maximal Consecutive Longest Common Subsequence) with some additional conditions. $MCLCS_1$ and $MCLCS_n$ use MCLCSs beginning at the first, and at the n-th character, respectively; $MCLCS_z$ only considers MCLCSs ending at the last character. NMCLCS is a normalised version of MCLCS, and it is computed as the following equation:

$$NMCLCS_i(w_c, w_e) = \frac{len(MCLCS_i(w_c, w_e))^2}{len(w_c) * len(w_e)} \tag{5.2}$$

where $MCLCS_i$ can be $MCLCS_1$, $MCLCS_n$ or $MCLCS_z$.
The similarity of the two strings S is calculated as below:

$$S(w_c, w_e) = \alpha * NLCS(w_c, w_e) + \sum_{i \in \{1,n,z\}} \alpha_i * NMCLCS_i(w_c, w_e) \tag{5.3}$$

where α, α_i are weights of NLCS and $NMCLCS_i$.

We reuse the same weights suggested by Islam *et al.* [39] in our statistics. Figure 5.3 shows rates of errors on the four datasets with different threshold values of similarity S.

Our observation reveals that about 83.5% of all errors have the similarity S equal or greater than 0.125. Similar to edit distance, the threshold of LCS similarity can be used in removing many incorrect candidates for each error.

5.3.2 Length effects

As to length effects, we examine not only word lengths but also OCRed token lengths. Furthermore, we suggest a novel classification by grouping errors according to word lengths and edit distances.

Word length

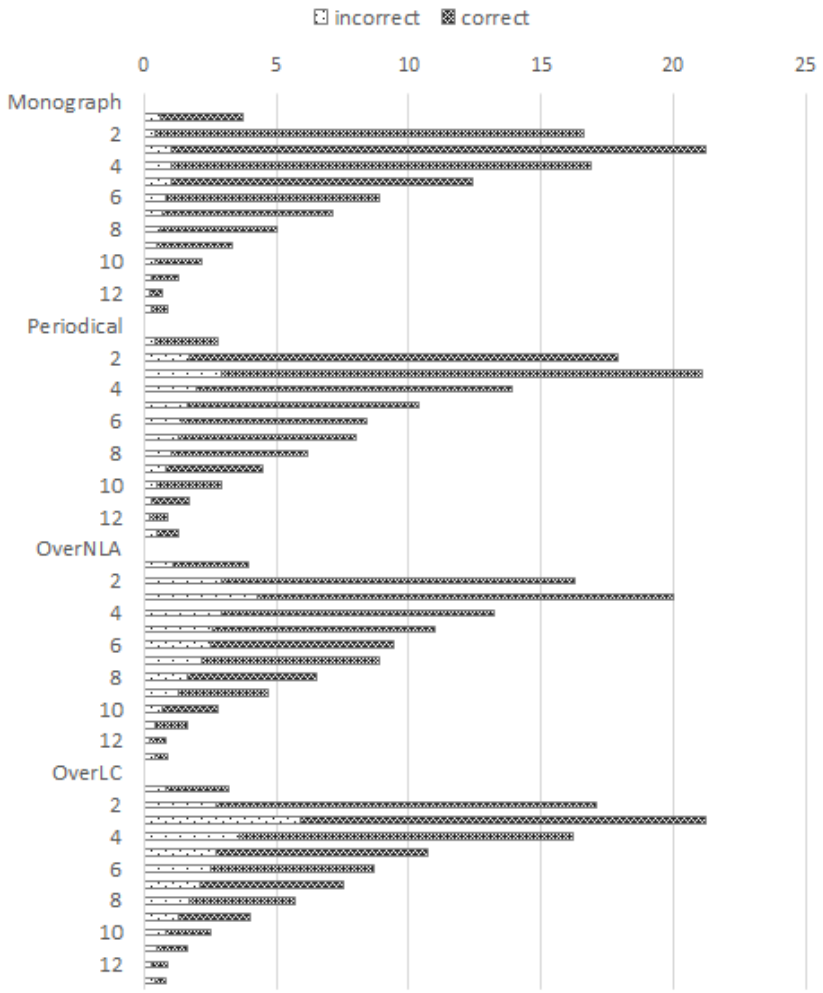


Figure 5.4: Rates of correct and incorrect word recognition based on word lengths.

In terms of word length, Kukich [54] found that more than 63% of the spelling errors are *short-word* errors. Percentages of correct/incorrect word recognition according to word lengths on our datasets are shown in Figure 5.4. According to our statistics, about 42.1% of OCR errors are *short-word* errors, which is a lower value than that of misspellings with

63% on average. In addition, from the highest percentage at length 3, the percentage of incorrect word recognition decreases gradually according to the increase of GT token length. Furthermore, around 85.27% of all OCR errors occur in words of lengths from 2 to 9.

OCR token length

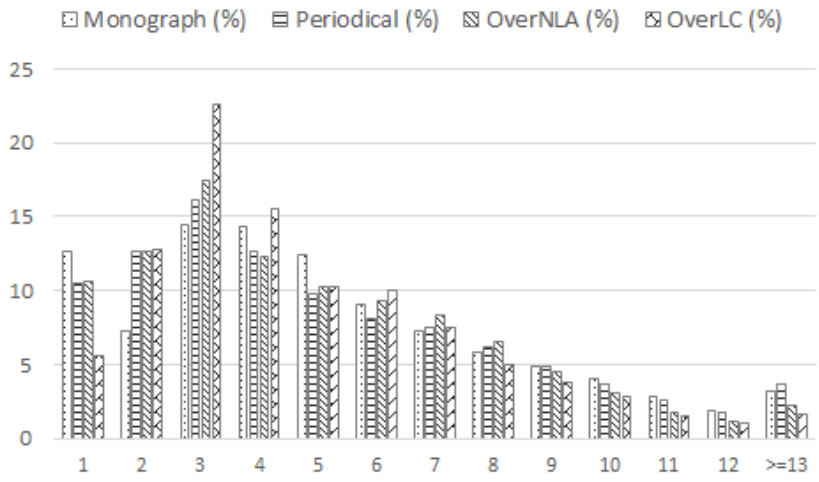


Figure 5.5: Error rates based on OCRed token lengths.

In practice, post-OCR approaches have to deal with OCRed tokens instead of GT words, and lengths of OCRed tokens can differ from those of GT words, therefore we consider lengths of OCRed tokens. For example, in OCRed tokens 'scho ol' and their GT word 'school', two incorrect OCRed tokens are 'scho' of length 4, and 'ol' of length 2; these OCRed tokens come from GT word of length 6.

Similar to word length, the analysis of incorrect OCRed token lengths (see Figure 5.5) suggests that incorrect OCRed tokens of length 3 are the most common one. In addition, about 80.55% of all invalid OCRed tokens are of lengths between 2 and 9.

Two-dimensional classification based on word lengths and edit distances

There are some arguments that it is more difficult to deal with *short-word* errors than with errors appearing in longer-length words. This is because *short-word* errors are more likely to yield another lexicon entry when applying character edit operations [55].

However, the problem does not only result from length but also from edit distance between an error and its GT word. For example, there are two errors (e.g., 'ict', 'lct') and

their GT word (e.g., 'let'). The first error 'ict' requires 2 edit operations to be transformed into its GT word, which is more challenging than the second error 'lct' needing only 1 modification to be converted to its GT word. To give a clear view of such problem, we suggest a novel classification by grouping errors according to word lengths and edit distances. With run-on errors (e.g., 'blue sky' vs. 'blucsky'), we assume the sum of lengths of all words related to the errors as their word length.

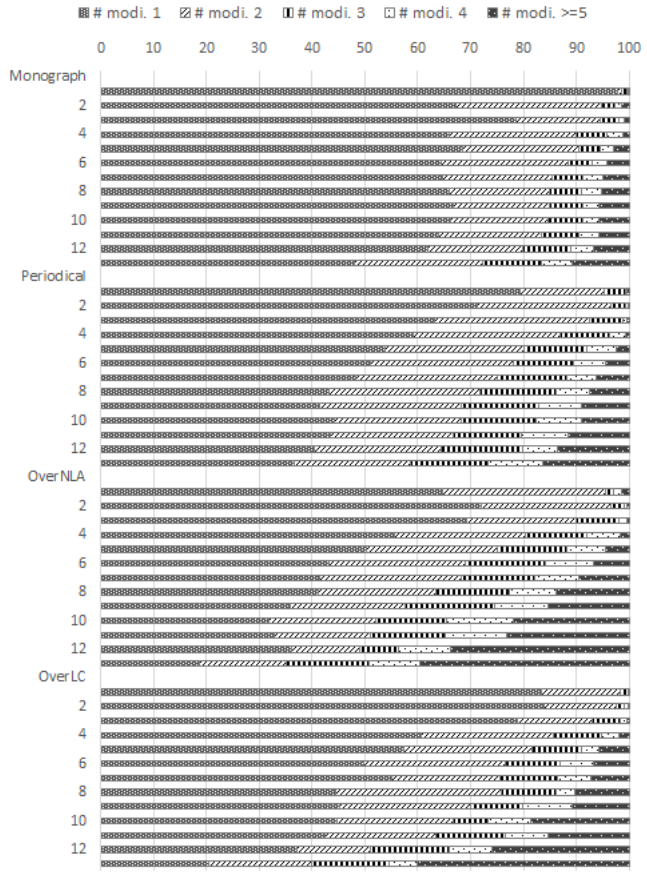


Figure 5.6: Error rates based on word lengths and edit distances.

The two-dimensional classification of four datasets is shown in Figure 5.6. Based on this classification, certain post-processing approaches could decide edit distance thresholds for each word length. As mentioned in Sec 5.3.1, around 81.49% of errors have edit distance of 1, 2. In other words, maximum number of possible errors that post-processing approaches can correct is about 81.49% if edit distance threshold is set as 2 for all word lengths.

In our opinion, by adjusting edit distance threshold according to word length, post-

OCR techniques can deal with higher rate of errors. Based on our observations, we suggest to set edit distance thresholds 2, 3, 4 for word lengths less than 4, 10, 13, respectively. On average, those settings increase the rate of errors that post-OCR techniques can process from 81.49% to 89.15%.

5.3.3 Erroneous character positions

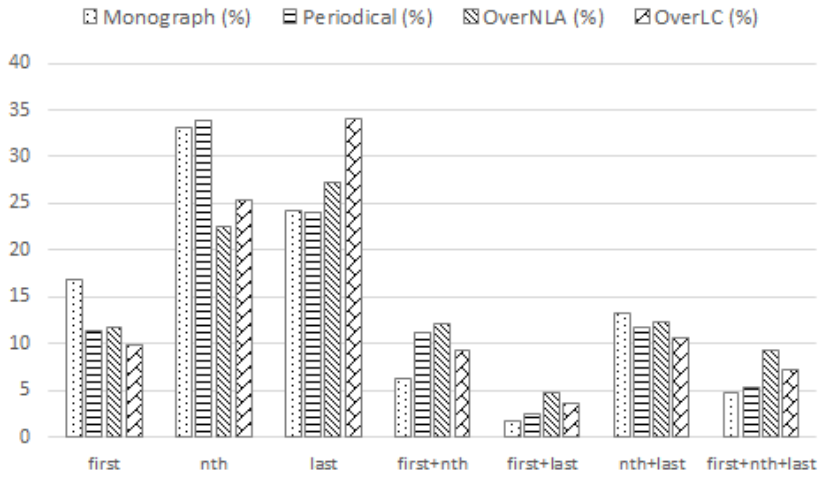


Figure 5.7: Error rates of erroneous character positions.

The survey on misspellings [54] showed that there are a few errors at the 1st character. However, there is no research related to erroneous character positions in OCRed text. Hence, we examine OCR errors at different character positions, including the first/last/middle position (denoted as *first, last, nth,* respectively), and their possible combinations (denoted as *first+last, first+nth, last+nth, first+last+nth* respectively). In case of *run-on* errors, because this error type incorrectly removes white space at the end of the first word, we decide that this error type always has one *last-position* error.

Details of erroneous character positions of our datasets are shown in Figure 5.7. While 12.46% of OCR errors are *first-position* errors, spelling errors have slightly smaller percent of such errors with average 11% of all errors.

It is noticeable that on average 27.37% of all errors are *last-position* errors, which are even comparable with that of *middle-position* errors (28.69%). Moreover, our observations on four datasets indicate that erroneous characters rarely appear at the first/last position

in the same error. In fact, statistics show that less than 10% of errors belong to (*first + last*) or (*first + last + nth*) combinations. Therefore, OCR post-processing can firstly focus on single positions or some combinations (*first+nth, last+nth*).

5.3.4 Real-word vs. non-word errors

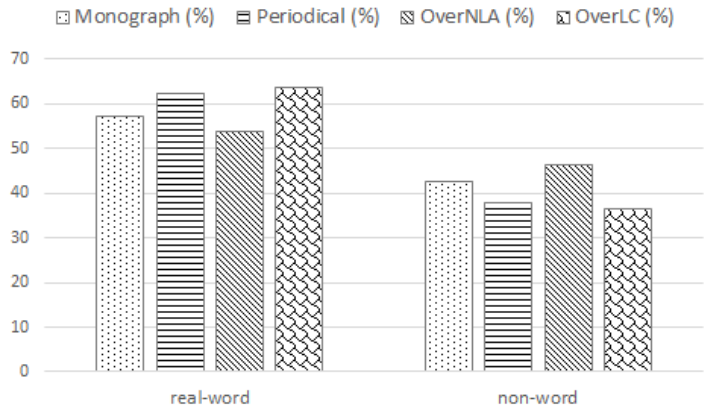


Figure 5.8: Rates of real-word vs. non-word errors.

In the next analysis we study the rate of *real-word* and *non-word* errors in OCRed text. *Real-word* errors are valid in dictionary but incorrect in context (e.g., 'hear' vs. 'bear'). The number of *real-word* errors vary naturally with the size of the lexicon [82]. Too small lexicon can ignore valid tokens as well as increase the number of false negatives. In contrast, a too large dictionary can match invalid tokens to low-frequent lexical entries or special domain terms, potentially raising the number of false positives. In other words, the larger the lexicon is, the more *real-word* errors can occur.

On the other hand, *non-word* errors are invalid in dictionary (e.g., 'hear' vs. 'hcar'). It is obvious that *non-word* errors are easier to be detected and corrected than *real-word* errors. In addition, there are words which appear in GT but are not lexicon entries, known as out-of-vocabulary (OOV) words. Using the word frequency of COHA corpus, the rate of OOV words in our datasets is found to be about 1%.

The statistics of *real-word* errors and *non-word* errors in Figure 5.8 show that approximately 59.21% of OCR errors are *real-word* errors. The proportion of *real-word* errors in our four datasets is about 1.47 times higher than that of *non-word* ones. On the contrary,

misspellings have an opposite trend with 67.5% *non-word* errors.

Our observations on the four datasets also indicate that approximately 13.77% of non-word errors involve digits, and 25.08% of real-word errors relate to punctuation. High percentage of punctuation errors is one notable feature of OCRed text. In fact, the low physical quality of old documents causes misrecognition of punctuation. Therefore, OCRed texts tend to contain more incorrect/redundant commas and dots than human-generated texts.

5.3.5 Word boundary

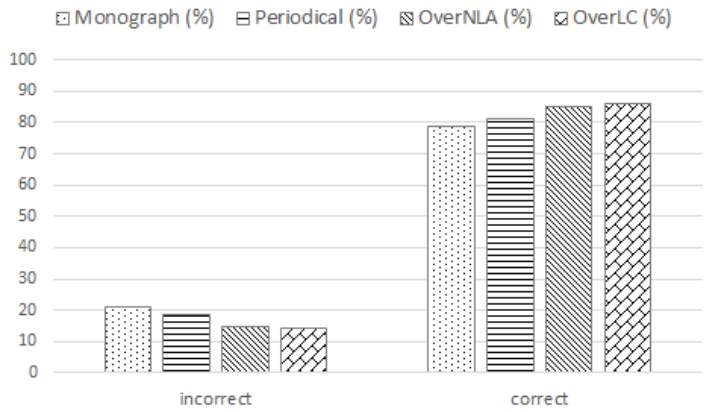


Figure 5.9: Rates of correct vs. incorrect word boundary errors.

This subsection observes word boundary aspect in detail. Let us call errors related to wrongly identified word boundaries as incorrect word boundary errors, and ones unrelated to word boundary problems as correct word boundary errors.

To give clearer views of OCR errors, we suggest to make a hierarchical classification based on incorrect/correct word boundary error types, and *real-word*/*non-word* error types. We firstly separate OCR errors into incorrect/correct word boundary error types. Secondly, in terms of incorrect word boundary error type, depending on inserting/deleting white spaces we classify into two main sub-types, including *incorrect split*/*run-on* error types. In terms of correct word boundary error type, we divide into *real-word*/*non-word* error types. Finally, for *incorrect split*/*run-on* error types we continue grouping into *real-word*/*non-word* error types.

Percentages of incorrect/correct word boundary types of our four datasets are shown in Figure 5.9. It is clear that all of the four datasets give a similar trend. Around 82.85% of errors are correct word boundary errors, which is much higher than that of incorrect word boundary ones.

Incorrect word boundary errors

In terms of incorrect word boundary errors, we study two popular sub-types: *incorrect split*/*run-on* error types. Incorrectly putting two or more words together creates a *run-on* error which is often not in the lexicon. In other words, most of *run-on* errors are *non-word* errors, and they are easy to be detected. Otherwise, correcting such errors is more complicated because it easily leads to a combinatorial explosion of the number of possible word combinations.

Wrongly splitting one word into some strings results in *incorrect split* errors. Both detecting and correcting such errors are challenging because some of split strings are not in the lexicon (*non-word* errors) and others are lexicon entries (*real-word* errors).

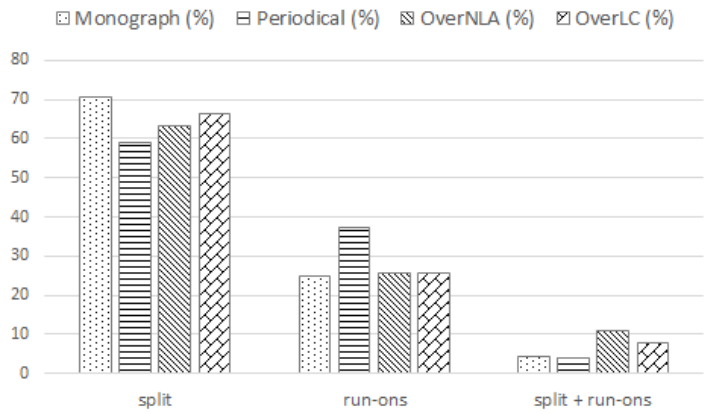


Figure 5.10: Error rates of incorrect word boundary subtypes.

Percentages of incorrect word boundary sub-types of four datasets are shown in Figure 5.10 with *incorrect split* errors denoted as *split*, *run-on* errors denoted as *run-on* and their combination (*split + run-on*). It is notable that the percent of *incorrect split* errors is on average 2.36 times higher than that of *run-on* errors. In contrast, most of incorrect word boundary errors in misspellings are *run-on* errors with 6.5 times higher occurrence than *incorrect split* ones. In addition, *incorrect split* and *run-on* errors rarely appear to-

gether in errors. The percentage of their combination (*split + run-on*) is only 6.8% on average, therefore, post-processing approaches can ignore it at first steps.

Correct word boundary

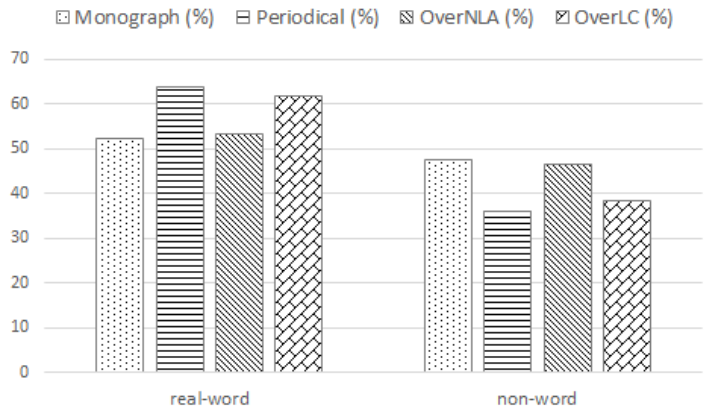


Figure 5.11: Error rates of correct word boundary subtypes.

In terms of correct word boundary, we directly classify errors into *real-word*/*non-word* error types. Percentages of *real-word* and *non-word* errors in correct word boundary type are shown in Figure 5.11. Real-word/non-word errors mentioned in this section are sub-sets of the *real-word* and *non-word* errors pointed out in Figure 5.8, which reveals the similar trend with their super-sets. Other *non-word* and *real-word* errors are of the incorrect word boundary type, with 28.72% *real-word* errors and 24.82% *non-word* ones, on average.

5.4 Summary of main findings

We summarise in this section the key observations from our study. Firstly, we examine OCR errors and compare them with spelling errors in several aspects. Misspellings and OCR errors have similar trends in two cases. In particular, most of them are *single-error* errors (74.5% misspellings, 58.92% OCR errors), and few of them are *first-position* errors (11% misspellings, 12.46% OCR errors).

However, misspellings and OCR errors differ in three other aspects, including *real-word* vs. *non-word* errors, *incorrect split* vs. *run-on* errors, and *short-word* errors. We find that most of misspellings (67.5%) are *non-word* errors while most of OCR errors (59.21%) are

real-word ones. Regarding the incorrect word boundary error type, the percentage of *run-on* errors is 6.5 times higher than that of *incorrect split* ones in case of spelling errors. In contrast, the proportion of *incorrect split* errors is on average 2.36 times greater than that of *run-on* errors in case of OCR errors. Moreover, while 63% of misspellings appear in short words, only 42.1% of OCR errors are *short-word* errors.

Secondly, besides similar aspects as in Kukich's survey, we present novel statistics (non-standard mappings, string similarities based on LCS, OCRed token lengths, and erroneous character positions).

For non-standard mappings, our analysis reveals that some characters 'b', 'd', 'h', 'm', 'n' are easily recognised as 'li', {'il', 'cl'}, 'li', {'rn', 'in'}, {'ri', 'ii'}, respectively. In opposite way, some strings 'li', {'il', 'cl'}, 'li', {'rn', 'in'}, {'ri', 'ii'} can be recognised as 'b', 'd', 'h', 'm', 'n', respectively.

In case of string similarities based on LCS, around 83.5% of OCR errors achieve no less than 0.125 similarity S with their GT words.

As to OCRed token lengths, they show similar trend with word lengths. Particularly, incorrect OCRed tokens of length 3 are the most common, and most of erroneous OCRed tokens are of lengths from 2 to 9.

For erroneous character positions, around 27.37% errors are *last-position* errors, and they thus are comparable to *middle-position* errors (28.69%). In addition, we observe that errors rarely have erroneous characters at both the first and last position (in total 9.75% of *first+last* and *first+last+nth*).

Finally, based on the analysis on four datasets, we make some suggestions for designing post-processing approaches. Because *last-position* errors rarely appear together with *first-position* errors, post-OCR techniques can ignore their combinations (*first+last*, *first+last+nth*).

Our observations show that deletion, insertion and substitution occasionally appear together in the same word (around 22.98%); algorithms of candidate generation can then pay more attention on single modification types instead of their combinations. Moreover, the rate of the number of substitution/deletion/insertion character candidates for each character position of OCRed token can be set as 5:1:1 in generating candidates.

Edit distance is considered as an important criteria in selecting relevant candidates. Interestingly, 81.49% of OCR errors are of edit distance 1 or 2, so with edit distance threshold 2, post-processing approaches could easily remove many irrelevant candidates. Moreover, edit distance thresholds can be adjusted according to word lengths. With flexible settings of edit distance threshold, post-processing techniques would be able to handle about 89.15% of errors.

5.5 Conclusions

This chapter examines different aspects of OCR errors towards a better understanding of OCR errors and related challenges. Based on our observations on four datasets, we also reveal some guidelines for designing post-processing approaches. In addition, we propose a novel two-dimensional classifications, including grouping errors according to word lengths and edit distances, as well as grouping of real-word/non-word errors following word boundary types.

Our work can be viewed as an important, initial step to further analyses or towards more efficient and robust post-OCR techniques. The lessons learnt in this chapter are applied to our post-OCR processing approaches, described in details in the remainder of the manuscript.

CHAPTER 6

Post-OCR error detection

Several post-processing approaches detect and correct remaining errors to improve the quality of OCRed texts. This chapter focuses on the error detection approaches. Based on our observation in Section 2.2, there is no clear comparison between the performance of NMT models and that of classification models on post-OCR error detection. Therefore, both of them are exploited with some novel features to locate error positions. The first one explores different features from both character and word levels to classify errors (denoted as *stat-detection-proposal*), while the second one applies neural network based language model to find errors (denoted as *nn-detection-proposal*).

6.1 Statistical approach

This approach employs various features at both character and word levels in order to identify whether an OCRed token is correct or incorrect via binary classification. An OCRed token needs to prove to be a valid word via feature values computed from its plausible candidate set. This section gives details of this method and assesses its performance on the competition datasets.

6.1.1 System description

Since feature values of each OCRed token are computed relying on its candidate set, our approach has one more step - candidate generation before typical steps of statistical approaches. In the following section, we discuss each step in detail.

Candidate generation

In this section, we focus on generating possible candidates for each token position in OCRed documents. In order to produce candidates, we utilise information related to an OCRed token at character and word levels. We consider character level as important as word level, therefore the same number of top candidates ($k = 5$) are used for each level.

At character level, we implement the approach similar to the one described in Section 7.1.1 to suggest candidates for each OCRed token. Candidates are generated from a character candidate graph and are ranked by the probabilistic character error model. For example, an OCRed token 'comes' can have its candidate set {'comes', 'cones', 'comas'}.

At word level, we make a use of local context to produce candidates. Word trigrams related to an OCRed token are considered. Let us assume the phrase $w_{-2}\ w_{-1}\ w\ w_{+1}\ w_{+2}$. The three trigrams involving the OCRed token w are $w_{-2}\ w_{-1}\ w$ (denoted as *left-trigram*), $w_{-1}\ w\ w_{+1}$ (denoted as *middle-trigram*), and $w\ w_{+1}\ w_{+2}$ (denoted as *right-trigram*).

In case of the *left-trigram*, we keep w_{-2}, w_{-1} and select candidate w_0 for replacing the OCRed token w. Then, we choose top k candidates based on the trigram frequency w_{-2}, w_{-1}, w_0. Similarly with the *middle-trigram* and *right-trigram*, top k possible candidates are chosen for the OCRed token position.

For example, given the OCRed token 'the' in the phrase 'his friend comes from the north we.t', its three related trigrams include 'comes from the', 'from the north', 'the north we.t'. Regarding the *left-trigram* 'comes from the', we keep 'comes from' and select candidates for the OCRed token 'the'. Three trigrams with the highest frequency are 'comes from the', 'comes from a', 'comes from an'. Thus, top k alternatives {'the', 'a', 'an'} of the *left-trigram* are chosen for the token position.

Similarly, we get word candidates for the OCRed token position from the *middle-trigram* and *right-trigram*, which are, {'the', 'up', 'a'} and the empty set, respectively.

Feature extraction

Several features are extracted at character and word levels. They can be divided into four groups: character ngram frequency, word ngram frequency, part-of-speech, and the frequency of the OCRed token in its candidate generation sets. It should be noted that due to shared characteristics between our datasets and the Corpus of Historical American English (COHA) [15], we use frequencies of ngrams and parts-of-speech (POS) of this largest corpus of historical English text. The CLAWS tagset is applied for all POS tags of this corpus.

Character ngram frequency

The index of peculiarity (or the peculiar index) of a token is recommended for detecting OCR errors [54], therefore we consider the peculiar index as a classifier feature. In our work, we reuse the formula of computing the peculiar index on frequency statistics of the historical corpus COHA.

The key idea of the peculiar index is that if strings contain non-existent or very infrequent ngrams (like 'jtg' or 'bkm'), they are detected as potential erroneous tokens. The peculiar index (*pe-index-orig*) of a token is the root-mean-square of the indices of its trigrams, and is formulated as follows:

$$\textit{pe-index-orig}(w) = \sqrt{\frac{\sum_{x \in trilist} index(x)^2}{n}} \tag{6.1}$$

where $trilist$ is the list of trigrams of the OCRed token w, n is the size of $trilist$, and $index(x)$ is the index of trigram x which is computed as follows:

$$index(x) = \frac{\sum_{y \in bilist} log(freq(y) - 1)}{2} - log(freq(x) - 1) \tag{6.2}$$

where $bilist$ is the list of bigrams of trigram x, $freq(x)$ is the frequency of x.

For example, we reuse the OCR phrase 'his friend comes from the south we.t' with the OCRed token 'we.t' and its two trigrams {'we.', 'e.t'}, the peculiar index of this token is:

$$\textit{pe-index-orig}(\text{'we.t'}) = \sqrt{\frac{index(\text{'we.'})^2 + index(\text{'e.t'})^2}{2}} \tag{6.3}$$

$$index(\text{`we.'}) = \frac{log(freq(\text{`we'}) - 1) + log(freq(\text{`e.'}) - 1)}{2} - log(freq(\text{`we.'}) - 1) \tag{6.4}$$

Tokens with a higher index of peculiarity tend to be incorrect [71]. However, it is unfair to compare the index of the peculiarity of long tokens and that of short tokens. Our analysis on the training part of our datasets suggests that long tokens tend to have a higher index of peculiarity. Therefore, we adapt the peculiar index to token length.

For each dataset, we group the indices according to the length of their corresponding tokens; for each group, the peculiar index is normalised by the highest peculiar index of that group. The adapted index of peculiarity (denoted as *pe-index*) and the corresponding token length (denoted as *tok-len*) are used as classifier features.

By catching frequent character ngrams, the character-level features have potential to correctly recognise out-of-vocabulary (OOV) words, which are often considered as errors because they are not present in the dictionary.

Word ngram frequency

Some features related to word ngram frequency are studied, including word frequency, bigram frequencies, skip-gram frequencies, and split-word.

Word frequency: The frequency of the OCRed token w is normalised by the maximum frequency of its candidate set C, and is utilised as one feature value (denoted as *word-freq*).

$$word\text{-}freq(w) = \frac{freq(w)}{\max_{c_i \in C}(freq(c_i))} \tag{6.5}$$

For example, for the OCRed token w='comes', its candidates with corresponding frequencies {'comes': 300, 'cones': 100, 'comas': 200}, the feature score of w is as follows:

$$word\text{-}freq(\text{`comes'}) = \frac{300}{\max(300, 100, 200)} = 1 \tag{6.6}$$

Bigram frequencies of the OCRed token and its neighbours: The bigram fre-

quency of the OCRed token and its previous token is normalised by the maximum bigram frequency of the OCRed token's candidates and its previous token's candidates, and then is used as one feature (denoted as *pre-bi-freq*).

$$\textit{pre-bi-freq}(w) = \frac{\max_{i \in C_{-1}} freq(i, w)}{\max_{i \in C_{-1}, j \in C} freq(i, j)} \tag{6.7}$$

where C, C_{-1} are the candidate set of the OCRed token, and that of its previous token; $freq(i, w)$ is the frequency of word bigram (i, w).

For example, there are the OCRed token 'the' in the phrase 'from the north', the candidate set C_{-1} {'from', 'front'}, and the candidate set C {'the', 'he', 'she'}. The occurrence frequencies between 'from', 'front' and each candidate of set C are {100, 2, 1}, {105, 2, 3}. The feature score of the OCRed token 'the' is computed as follows:

$$\textit{pre-bi-freq}(\text{'the'}) = \frac{\max(100, 105)}{\max(100, 105, 2, 2, 1, 3)} = 1 \tag{6.8}$$

Similarly, the bigram frequency of the OCRed token and its next word is normalised by the maximum bigram frequency of the OCRed token's candidates and its next token's candidates, and is used as a feature (denoted as *next-bi-freq*).

$$\textit{next-bi-freq}(w) = \frac{\max_{j \in C_{+1}} freq(w, j)}{\max_{i \in C, j \in C_{+1}} freq(i, j)} \tag{6.9}$$

where C, C_{+1} are the candidate set of the OCRed token, and that of its next token.

In addition, the product of the previous bigram frequency and the next bigram frequency (denoted as *prod-bi-freq*) is also considered as one feature. Totally, three features concerning word bigrams are used in classifying errors, including *pre-bi-freq*, *next-bi-freq*, and *prod-bi-freq*.

Skip-gram frequencies: In order to reduce the data sparsity problem, skip-grams [84] are examined beside contiguous ngrams. Since COHA provides only 4-gram word frequencies, we focus on 2-skip-bigrams of an OCRed token and its neighbors.

Frequencies of 2-skip-bigrams of the OCRed token and three left/right words are calculated. The sum of these concurrent frequencies is normalised by the number of 2-skip-

bigrams (*i.e.* six in our work) and frequency of the OCRed token, is then used as the skip-gram feature value. In particular, for an OCRed token w in the phrase w_{-3} w_{-2} w_{-1} w w_{+1} w_{+2} w_{+3}, the score of skip-gram feature (denoted as *skip-grams*) is calculated as follows:

$$\textit{skip-grams}(w) = \frac{\sum_{w_i \in L_1} freq(w_i, w) + \sum_{w_j \in L_2} freq(w, w_j)}{6 * freq(w)} \tag{6.10}$$

where the list L_1 is $\{w_{-3}, w_{-2}, w_{-1}\}$, the list L_2 is $\{w_{+1}, w_{+2}, w_{+3}\}$.

For example, for the OCRed token $w =$ 'from' in the phrase 'his friend comes from the north we.t', the skip-gram feature is computed as follows:

$$\begin{aligned} \textit{skip-grams}(\text{'from'}) &= \frac{sumfreq}{6 * freq(\text{'from'})} \\ sumfreq &= freq(\text{'his'}, \text{'from'}) + freq(\text{'friend'}, \text{'from'}) \\ &\quad + freq(\text{'comes'}, \text{'from'}) + freq(\text{'from'}, \text{'the'}) \\ &\quad + freq(\text{'from'}, \text{'north'}) + freq(\text{'from'}, \text{'we.t'}) \end{aligned} \tag{6.11}$$

Split-word feature: While *run-on* errors are easily found by lexical techniques, *incorrect split* errors are more challenging to identify. In order to deal with incorrect split errors (e.g., 'made' vs. 'ma lie'), we generate a split-word candidate list C_s from the OCRed token w and its next token. For each candidate, we compute a score based on word frequency and bigram frequency, then use the highest score as a classifier feature (denoted as *split-word*), as shown in Equation 6.12.

$$\textit{split-word}(w) = \max_{c_i \in C_s} (\alpha * x(c_i, w) + (1 - \alpha) * y(c_i, w)) \tag{6.12}$$

where w_{-1} is the previous word, $x(c_i, w)$ is 1 if $freq(c_i) > freq(w)$ else 0, $y(c_i, w)$ is 1 if $freq(w_{-1}, c_i) > freq(w_{-1}, w)$ else 0, α is the contribution rate between word frequency and bigram frequency, which is selected by keeping all other parameters same and trying different values between 0 and 1 with a step of 0.1. Our experiments shows that $\alpha = 0.5$ is the best level.

Consider, for example, the OCR phrase 'they main tain good relations', two adjacent OCRed tokens 'main', 'tain' and the previous token 'they'. The split-word candidate set C_s of 'main tain' is {'maintain', 'maintan', 'niaintain'}. The OCRed token 'main' has the highest frequency in comparison to the candidates and only 'maintain' has higher bigram frequency than 'main', hence the feature score is as below:

$$\textit{split-word}(\text{'main'}) = \max(0.5, 0, 0) = 0.5 \tag{6.13}$$

Part-Of-Speech (POS) features

POS is considered as the general form of ngram feature. Our work utilise POS tag in length 3 as a classifier feature.

Let us assume that there are the OCRed token w in the phrase $w_{-2}\ w_{-1}\ w\ w_{+1}\ w_{+2}$, and its candidate set C $\{c_1, c_2, c_3\}$. The tri-POSs related to the OCRed token w are pos_{-2} pos_{-1} pos, pos_{-1} pos pos_{+1}, and pos pos_{+1} pos_{+2} denoted as the *left-POS*/*mid-POS*/*right-POS*, respectively.

We first get a list of POS tags of each token $w_{-2}, w_{-1}, w, w_{+1}, w_{+2}$ from COHA corpus, denoted as L_{-2}, L_{-1}, L, L_{+1}, L_{+2}, respectively. OCRed token's candidates c_1, c_2, c_3 also have the POS lists $L_{c_1}, L_{c_2}, L_{c_3}$, respectively.

As to the *left-POS*, all possible tri-POSs of the OCRed token and its candidates are created by combining POS tags of L_{-2}, L_{-1} and each POS list of $\{L, L_{c_1}, L_{c_2}, L_{c_3}\}$.

The maximum frequency of the *left-POS* of OCRed token is computed, then normalised by the maximum frequency of tri-POSs created from L_{-2}, L_{-1} and each POS list of $\{L_{c_1}, L_{c_2}, L_{c_3}\}$. The normalised *left-POS* frequency of OCRed token is used as one feature.

$$\textit{left-POS}(w) = \frac{\max_{i \in L_{-2}, j \in L_{-1}, k \in L} freq(i, j, k)}{\max_{L_{c_i} \in L_c} \max_{i \in L_{-2}, j \in L_{-1}, k \in L_{c_i}} freq(i, j, k)} \tag{6.14}$$

with $L_c = \{L_{c_1}, L_{c_2}, L_{c_3}\}$.

We reuse the same OCR phrase to illustrate this feature, 'his friend comes from the north we.t'. The OCRed token w is 'comes', and its candidate set is {'comes', 'cones', 'comas'}. The POS lists of two previous words 'his', 'friend' are {*appge, ppge*}, {*nn1,*

np1}, respectively. The POS lists of 'comes', 'cones', 'comas' are {*vvz*}, {*nn2*}, {*nn2*}, respectively. The possible *left-POS* of the OCRed token 'comes' include {*appge nn1 vvz*, *ppge nn1 vvz*, *appge np1 vvz*, *ppge np1 vvz*} and their maximum frequency is 10,625. Similarly, the maximum frequencies of the *left-POS* of the candidates 'comes', 'cones', 'comas' are 10,625; 16,425; 16,425; respectively. Consequently, the feature score is:

$$\textit{left-POS}(\text{'comes'}) = \frac{10,625}{16,425} = 0.65 \tag{6.15}$$

Similarly, the normalised *mid-POS*, *right-POS* frequencies of OCRed token are computed. In total, there are four features scores extracted from POS, including *left-POS*, *mid-POS*, *right-POS*, and their product (denoted as *prod-POS*).

The OCRed token frequency in its candidate generation sets

As mentioned in Section 6.1.1, there are four sources to generate error candidates: character error model, local word context (*left-trigram*, *middle-trigram*, and *right-trigram*). The number of appearances of the OCRed token in its candidate sets is normalised by the number of candidate sets, then it is used as a feature (denoted as *tok-freq*).

While other features are built from individual words or word context, the *tok-freq* feature is designed from both of them. Therefore, we think that this combined feature can deal with *non-word* as well as *real-word* errors.

For example, the OCRed token w='the' in the phrase 'his friend comes from the north we.t' has the noisy channel candidate set {'the', 'she', 'he'}. The local context candidate sets include the *left-trigram* set {'the', 'a', 'an'}, the *middle-trigram* set {'the', 'up', 'a'}, and the empty *right-trigram* set. As a result, the feature score is that:

$$\textit{tok-freq}(\text{'the'}) = \frac{3}{4} = 0.75 \tag{6.16}$$

Error classification

If the OCRed token is an error, then its feature vector is labeled as 1, otherwise 0. Gradient Tree Boosting is one of the best performing classifiers [101], therefore we use it to train and classify OCR errors. In our experiments, we use the scikit-learn library [80] with the

maximum of 5 nodes in the tree, 800 boosting stages, and other default parameters.

6.1.2 Experimental results

As mentioned in Section 3.2.2, our results are evaluated by the same metrics (Precision, Recall, F-measure) and the official evaluation tool of the competition.

Results on the two competitions

We compare our proposed approach with the top six approaches of the first competition (CLAM, Char-SMT/NMT, EFP, MMDT, WFST-PostOCR, 2-pass RNN), and those of the second one (CCC, CLAM, CSIITJ, RAE1, RAE2, UVA), which are presented in Section 2.2. The performances shown in Tables 6.1 and 6.2 indicate that our method outperforms the highest performing approaches in the first competition, with 6% and 2% greater F-measure than the state-of-the-art (WFST-PostOCR) on Monograph and Periodical, respectively. In case of the second competition, our method underperforms the winner (CCC) which is the fine-tuned neural network model.

Furthermore, the performance on different error types (*non-word* vs. *real-word*) is clarified. In our opinion, due to the sparsity problem, our context features are not really effective in detecting *real-word* errors. In fact, despite applying possible features at word level (from ngram, skip-gram to part-of-speech) our approach enables to identify 43% of them on Monograph, 49% on Periodical, 33% on Comp2019.

In case of *non-word* errors, our approach correctly detects the majority of *non-word* errors (96% of them on Monograph, 85% on Periodical, and 65% on Comp2019). Most of the unidentified *non-word* errors are erroneous tokens related to numbers.

It should be reminded that *non-word* errors are easier to detect than *real-word* errors. Therefore, the rate of *non-word* and *real-word* errors of datasets heavily affects on detection performance. We believe that our method performs differently on three datasets partly due to this reason. In fact, the rate of Monograph is about 1.5 while that of Periodical and Comp2019 is around 0.5. Furthermore, *real-word* errors in Comp2019 are more difficult to handle. They are not only some single tokens but also a whole sentence which is a result of wrongly line recognition.

Table 6.1: Experimental results of OCR error detection task in English datasets of the first competition ICDAR2017, '-' denotes no reported result.

	Monograph			Periodical		
Approaches	P(%)	R(%)	F(%)	P(%)	R(%)	F(%)
Char-SMT/NMT [3]	98	51	67	89	50	64
Single Char-SMT/NMT [3]	-	-	62	-	-	59
CLAM [8]	94	52	67	-	-	-
EFP [8]	63	77	69	53	55	54
MMDT [91]	83	55	66	70	32	44
WFST-PostOCR [8]	67	82	**73**	68	68	**68**
2-pass RNN [8]	58	77	66	64	68	66
stat-detection-proposal	82	76	**79**	81	61	**70**

In addition, we also take into account out-of-vocabulary (OOV) words which are correct word but do not exist in a normal dictionary. On average, 38% of OOV words in our three datasets are correctly recognised.

Feature analysis

In total, our approach relies on 13 features to classify OCR errors. Three features (*pe-index*, *tok-len*, *word-freq*) built from individual words mainly focus on detecting *non-word* errors. The features created from word context (bigram frequencies, skip-grams, POS tags) concentrate on identifying *real-word* errors. Regarding the combined feature *tok-freq*, it has potential to deal with both of *non-word* and *real-word* errors. The remaining feature *split-word* is designed to handle incorrect-split errors.

Each feature contributes differently to predict the target. The more frequently a feature is used in the split points of a decision tree, the more important that feature is. In ensemble classifier like Gradient Tree Boosting, the relative feature importance is computed by

Table 6.2: Experimental results of OCR error detection task in English datasets of the second competition ICDAR2019, '-' denotes no reported result.

	Comp2019		
Approaches	P(%)	R(%)	F(%)
CCC [89]	-	-	**67**
CLAM [89]	-	-	45
CSIITJ [89]	-	-	45
RAE1 [89]	-	-	53
RAE2 [89]	-	-	57
UVA [89]	-	-	47
stat-detection-proposal	70	52	60

summing up the feature importances of the individual trees, then dividing by the total number of trees [80].

As our approach detects more *non-word* errors than *real-word* ones, we believe that the features based on individual words (*pe-index*, *tok-len*, *word-freq*) and the combined feature *tok-freq* are more important than the features relying on word context. The relative importance of the features of our best model shown in Fig. 6.1 supports our assumption.

In particular, *tok-freq* is the most important one on Monograph (36.8%) and Periodical (52.3%) but not on Comp2019 (2.9%). The feature *pe-index* is the third important one on Monograph (15%) and Periodical (9.4%), but is one of two most important features on Comp2019 (21.1%).

The importance of our novel features are evaluated separately in Table 6.3. In overall, *pe-index* and *tok-freq* help to increase recall on three datasets, and precision on Periodical. This trend can be partly explained by characteristics of these features and different rates of non-word/real-word errors of the three datasets. In fact, *pe-index* and *tok-freq* mainly detect non-word errors, and the rates of non-word/real-word in Monograph are higher than in Periodical and Comp2019 (1.5, 0.5, and 0.5, respectively).

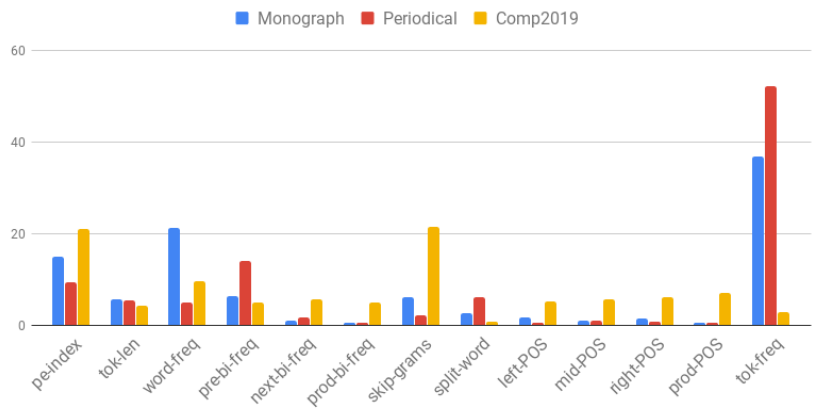


Figure 6.1: The relative importance of the features.

Table 6.3: Performance of our models without some novel features, $model_1$ (11 basic features), $model_2$ ($model_1$ + *pe-index*), $model_3$ ($model_1$ + *tok-freq*), Stat-proposal is a *stat-detection-proposal*.

	Monograph			Periodical			Comp2019		
Models	P(%)	R(%)	F(%)	P(%)	R(%)	F(%)	P(%)	R(%)	F(%)
$Model_1$	87	31	46	76	59	66	71	49	58
$Model_2$	91	48	63	80	61	69	71	50	59
$Model_3$	82	74	78	80	58	67	70	52	**60**
Stat-proposal	82	76	**79**	81	61	**70**	70	52	**60**

6.2 Neural network based approach

BERT [20] is a well-known contextual language representation model. The pre-trained BERT model can be fine-tuned to handle a variety of down-stream tasks. In this section we investigate the application of BERT model on post-OCR processing.

6.2.1 System description

BERT [20] is a multi-layer bidirectional transformer encoder. It is pre-trained on unlabeled data over two different tasks, including Masked Language Model (MLM), and Next Sentence Prediction (NSP). In the MLM task, the authors [20] obtain a bidirectional pre-trained model by randomly masking some percentages of the input tokens, and then predicting those masked tokens. The second task (NSP) enables the model to learn the relationship between two sentences.

BERT models can be modified to deal with NLP problems. Downstream tasks are firstly set with the pre-trained parameters, which are adjusted by their labelled data. There are multiple task-specific BERT models [20], some of them work at sentence level, others perform at token level. Error detection problem can be viewed as token classification which classifies OCRed tokens as either valid or invalid. We focus on fine-tuning BERT models at token level.

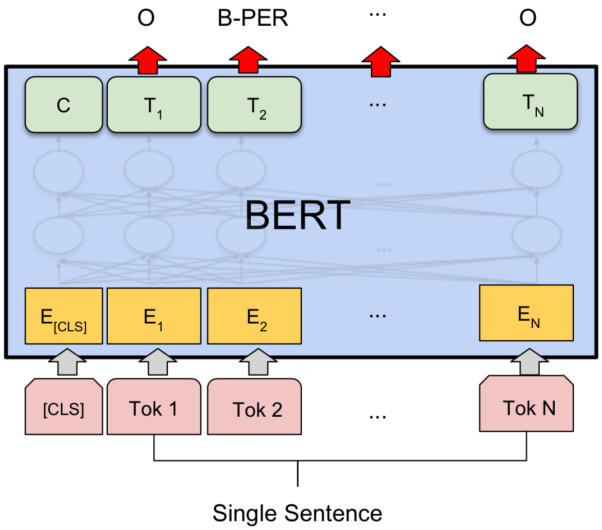


Figure 6.2: Single sentence tagging tasks [20].

We adapt the model of named entity recognition (NER) (as shown in Figure 6.2) to an error detection model. Particularly, instead of tagging tokens with NER taggers, we tag tokens with label 1 (invalid token) or 0 (valid token). Our approach is similar to the one of the winner of the 2019 competition, but we simplify the model with only one fully-connected layer on the top of the hidden-states output. In addition, it is proved

that pre-trained word embedding models increase the performance of NLP tasks. Thus, instead of randomly initialising embeddings like the competition winner CCC does, we employ popular word embeddings (Fasttext, Glove) in our model.

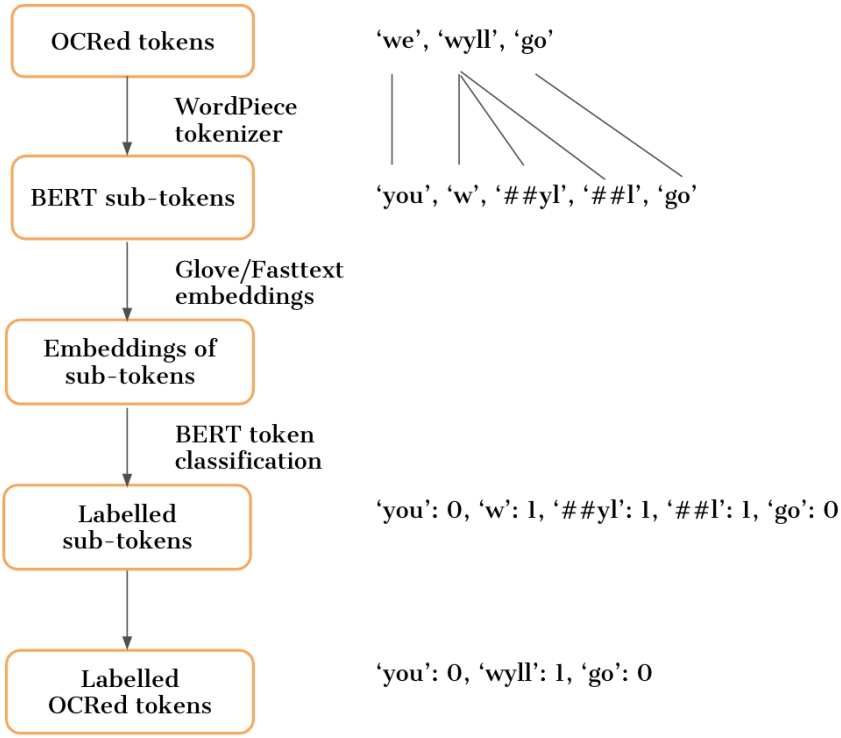


Figure 6.3: Error detection based on BERT model.

Figure 6.3 illustrates steps of our approach. OCR input is first split into OCRed tokens based on white-space. Next, we apply WordPiece [102] tokenisation on each token to get corresponding sub-tokens. A mapping between the original OCRed token and its sub-tokens is also maintained. Then, Glove or Fasttext is used to embed sub-tokens in lieu of assigning random numbers as initial embeddings.

After that, these embeddings are combined with segment and position embeddings as inputs of BERT token classification model, which is a BERT model with an additional fully-connected layer. This design is simpler than the state of the art which uses both convolutional and fully-connected layers.

The outcome of this stage is labelled sub-tokens, with label 1 for invalid tokens and 0 for valid tokens. Finally, the original tokens are considered as invalid ones if at least one of their sub-tokens is labelled as invalid.

Take an OCR sequence ‘we wyll go’ with an error ‘wyll’ as an example to illustrate our approach. The input of the first step is a list of OCRed tokens tokenised by white-spaces, {‘we’, ‘wyll’, ‘go’}. Applying WordPiece on each OCRed token, we have the corresponding sub-tokens and their mappings to their original tokens, {‘we’: ‘we’, ‘wyll’: {‘w’, ‘##yl’, ‘##l’}, ‘go’: ‘go’}. Next, the pre-trained word embeddings Glove or Fasttext embed the sub-tokens to be used as inputs for BERT token classification. The classifier labels each sub-token as either a valid or invalid word. The original token (‘wyll’) is identified as the error since its sub-tokens are classified as invalid ones (‘w’, ‘##yl’, ‘##l’).

In our experiments, we apply uncased BERT-base model with batch size as 32, learning rate of the optimizer Adam as 3e-5, maximum sequence length as 75. The model is trained with a higher number of epochs than recommended (20) while the other hyperparameters remain unchanged.

6.2.2 Experimental results

Our approach is evaluated on English datasets of the competition on post-OCR text correction ICDAR2017 [8], ICDAR2019 [89] by its official evaluation metrics. Details of datasets and metrics are indicated in Section 3.

Tables 6.4 and 6.5 illustrate performances of our approach in detail. In overall, we outperform other approaches on Periodical (with 4% higher F-measure) and Comp2019 (with 1% higher F-measure) but not on Monograph.

These results are possibly explained by the rate of real-word and non-word errors in each dataset and the strengthen of our neural network approach. In fact, there are more real-word errors on two datasets (Periodical and Comp2019) than on Monograph. In addition, BERT is a contextual language model, so it is reasonable that the BERT-based model is possible to detect more real-word errors.

Percentage of correctly detected real-word errors support our assumption. Our approach is able to identify 64% of context-sensitive errors on Monograph, 63% on Periodical, 48% on Comp2019, which is much better than our *stat-detection-proposal*. This approach also attains higher results of correctly detected non-word errors with 95% on Periodical, 93% on Comp2019, but not on Monograph (82%). Similarly, the percentage of correctly

Table 6.4: Experimental results of OCR error detection task in English datasets of the first competition ICDAR2017, '-' denotes no reported result.

	Monograph			Periodical		
Approaches	P(%)	R(%)	F(%)	P(%)	R(%)	F(%)
Char-SMT/NMT [3]	98	51	67	89	50	64
Single Char-SMT/NMT [3]	-	-	62	-	-	59
CLAM [8]	94	52	67	-	-	-
EFP [8]	63	77	69	53	55	54
MMDT [91]	83	55	66	70	32	44
WFST-PostOCR [8]	67	82	**73**	68	68	**68**
2-pass RNN [8]	58	77	66	64	68	66
stat-detection-proposal	82	76	**79**	81	61	70
nn-detection-proposal	83	63	72	83	66	**74**

recognised OOV words is comparable to our *stat-detection-proposal* with about 61% on average on three datasets.

6.3 Conclusions

In this chapter, we present two different approaches on error detection task: 1) the first one is based on classification model, 2) the second one relies on bidirectional contextual language model. Each approach has both advantages and disadvantages.

The *stat-detection-proposal* examines a novel OCR error detection approach which verifies whether an OCRed token is a correct one using feature values computed from its plausible candidate set. Our *stat-detection-proposal* allows to detect several non-word errors. In addition, two novel features (*pe-index*, *tok-freq*) are found to play important roles in the detection of incorrect OCRed tokens.

A problem of this approach is that it is time-consuming to generate the plausible

Table 6.5: Experimental results of OCR error detection task in English datasets of the second competition ICDAR2019, '-' denotes no reported result.

	Comp2019		
Approaches	P(%)	R(%)	F(%)
CCC [89]	-	-	**67**
CLAM [89]	-	-	45
CSIITJ [89]	-	-	45
RAE1 [89]	-	-	53
RAE2 [89]	-	-	57
UVA [89]	-	-	47
stat-detection-proposal	70	52	60
nn-detection-proposal	74	62	**68**

candidate set for each OCRed token. Nevertheless, the candidate set can be reused for the error correction step in statistical approaches. Another issue of this method is that its performance on real-word errors is limited although contextual features are considered.

Regarding the latter, it does not require to generate candidates for each token and it enables to detect several real-word errors. However, it does not function well without the pre-trained BERT model which was trained on large datasets using 4 Cloud Tensor Processing Units (TPUs), with totally 16 TPUs in 4 days.

Error detection approaches provide the positions of errors, which are a required input of the next step - correction. The following chapter details our post-OCR error correction approaches.

CHAPTER 7

Post-OCR error correction

After we have presented approaches for error detection in Chapter 6, the remaining step is naturally to attempt to correct the identified errors. This chapter describes two error correction approaches: a statistical approach and an approach based on neural networks.

7.1 Statistical approach

Our statistical method makes use of confusion probability obtained from the noisy channel model of single or multiple edits, and context probability given by language model. These two features are essential ones as suggested by related work [48, 66], where they are used to predict the likelihood of each candidate to be the corrected version of an error using a regression model.

7.1.1 System description

Our regression approach is divided into three steps: candidate generation and weighting relying on an error model, candidate scoring using a language model and candidate ranking based on a regression model. Details of each step are discussed in the following subsections.

Candidate Generation and Weighting based on an error model (Step 1)

In the first step, we generate candidates based on the character candidate graph which can deal with run-on and split-word errors. Such candidates are weighted by using a modified confusion probability.

Candidate Generation:

A string can be generated from the other string by edit operations of three edition types (deletion, insertion, or substitution). Therefore, we create the character candidate graph based on a 'seed' word (an OCR error) with three corresponding node types (deletion, insertion, or substitution). Then we use this graph to produce candidates by one or more edit operations. More specifically, if two characters are generated from one 'seed' character, this is a deletion node; otherwise, if one character is created from two adjacent 'seed' characters, it is an insertion node. In case that one character is substituted by one 'seed' character, we have a substitution node.

Training dataset reveals that insertion and deletion caused by two adjacent characters are more common than those caused by three or more adjacent characters, therefore in this chapter, we limit to two adjacent characters.

After graph construction, Breadth First Search (BFS) with some heuristic tuned from training dataset (maximum length of candidates, minimum confusion probability) is used to deal with the complexity.

The example graph is shown in Figure 7.1. If 'ar.d' is an OCR error, all 'seed' characters 'a', 'r', '.', and 'd' are denoted as yellow nodes. High frequency substitution characters of 'a', 'r', '.', and 'd' are 'e', 'n', ',' and 'l', respectively, which are denoted as green nodes. Two adjacent characters 'r', '.' can be combined to generate the character insertion node 'n' denoted as a red node; one 'seed' character 'd' can be divided into two characters 'il' denoted as a blue node. One possible candidate of the error 'ar.d' in Figure 7.1 is 'and' which is generated from the substitution node 'a', the insertion node 'n', and the substitution node 'd'.

By using the character candidate graph, our approach can deal with two difficult error types, which are split-word errors (for instance, 'appointed' is recognised as 'ap pointed') and run-on errors (for example, 'doubtfull of' is recognised as 'doubtfud.of'). However,

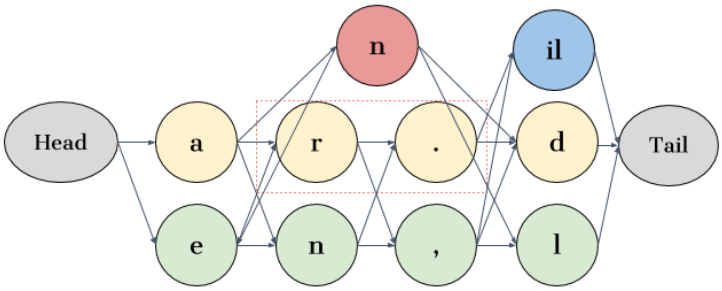


Figure 7.1: Example of character candidate graph.

there are some limitations in quality control of too many candidates generated by one run-on error. Therefore, this chapter only allows a punctuation and a digit be substituted by the space.

Candidate Weighting: Candidate are weighted relying on confusion matrices obtained from the training part of each dataset.

The conditional probability $p(x|w)$ of the given source word w recognised by the OCR software as the string x (also named as confusion probability of source word and OCR string) can be estimated by the confusion probabilities of the characters in x assuming that character recognition in OCR is an independent process [97].

Let $x_{1,i}$ be the first i characters of OCR string x and let $w_{1,j}$ be the first j characters of source string w. We define $p(x_{1,i}|w_{1,j})$ to be the conditional probability that the substring $w_{1,j}$ is recognised as $x_{1,i}$ by the OCR process. $p(x_{1,i}|w_{1,j})$ is calculated as below:

$$p(x_{1,i}|w_{1,j}) = \max \begin{cases} p(x_{1,i}|w_{1,j-1}) \times p(del(w_j)) \\ p(x_{1,i-1}|w_{1,j}) \times p(ins(x_i)) \\ p(x_{1,i-1}|w_{1,j-1}) \times p(sub(x_i|w_j)) \end{cases} \tag{7.1}$$

In typical formula, the probabilities of insertion, deletion are conditioned on the previous character [11], are computed as below:

$$p(del(w_j)) = \frac{del[w_{j-1}w_j]}{count[w_{j-1}w_j]}, \text{ if deletion} \tag{7.2}$$

where $del[w_{j-1}w_j]$ is a number of times that the source characters $w_{j-1}w_j$ are recognised

as w_{j-1}; $count[w_{j-1}w_j]$ is the appearance frequency of $w_{j-1}w_j$.

$$p(ins(x_i)) = \frac{ins[w_{j-1}x_i]}{count[w_{j-1}]}, \text{ if insertion} \tag{7.3}$$

where $ins[w_{j-1}x_i]$ is a number of times that w_{j-1} is recognised as $w_{j-1}x_i$; $count[w_{j-1}]$ is the appearance frequency of w_{j-1}.

The probability of substitution are calculated as follows:

$$p(sub(x_i|w_j)) = \frac{sub[x_i, w_j]}{count[w_j]}, \text{ if substitution} \tag{7.4}$$

where $sub[x_i, w_j]$ is a number of times that w_j is recognised as x_i.

Because erroneous OCRed characters frequently appear together, two or more error characters can be recognised as one different correct character, or one character can be recognised as different correct characters [42]. It means that the insertion, deletion can depend on the different previous character instead of the same previous one like Equations 7.2 and 7.3. For example, 'li' can be wrongly recognised as 'h', 'm' can be wrongly recognised as 'in', etc.

As a result, we propose to compute the probabilities of deletion and insertion by using the probability of substitution of many characters by one character or one character by many characters. The modified probabilities of deletion and insertion can be formulated as below:

$$p(del(w_j)) = p(sub(x_{i-1}|w_{j-1}w_j)) = \frac{sub[x_{i-1}, w_{j-1}w_j]}{count[w_{j-1}w_j]}, \text{ if deletion} \tag{7.5}$$

$$p(ins(x_i)) = p(sub(x_{i-1}x_i|w_{j-1})) = \frac{sub[x_{i-1}x_i, w_{j-1}]}{count[w_{j-1}]}, \text{ if insertion} \tag{7.6}$$

For instance, the correction is 'and', and the error is 'ar.d'. In this case, the correct letter 'n' is recognised as the error characters 'r.'. It is similar to insertion error type except that it does not have the same previous character, so we cannot apply the typical formula of insertion (Equation 7.3) directly. Typical approach (denoted as *typical-prob*)

uses the substitution formula twice to calculate that confusion probability:

$$p(\text{`ar.d'}|\text{`and'}) = p(sub(\text{`a'}|\text{`a'})) \times p(sub(\text{`r'}|\text{`n'})) \times p(sub(\text{`.'}|\text{`'})) \times p(sub(\text{`d'}|\text{`d'})) \quad (7.7)$$

Our approach (denoted as *modified-prob*) applies the substitution formula once:

$$p(\text{`ar.d'}|\text{`and'}) = p(sub(\text{`a'}|\text{`a'})) \times p(sub(\text{`r.'}|\text{`n'})) \times p(sub(\text{`d'}|\text{`d'})) \quad (7.8)$$

Our proposal also affects on creating the character confusion matrix. In particular, if one character (c) is recognised as two characters which are different from character (c), it is insertion. If two characters (c_1, c_2) are recognised as one character which is different from characters (c_1, c_2), it is deletion. Otherwise, it is substitution.

Candidate Scoring using Language Model (Step 2)

After generating and weighting candidates at character level, in the second step, we utilise context information to score candidates of each OCR error.

Similar to some approaches of the context-based type, we consider the typical statistical language model (SLM) to deal with this problem. Moreover, we also explore the state-of-the-art recurrent neural network based language model (RNN-LM) [67] to compare two types of language models in context of erroneous OCRed text. SLM and RNN-LM are trained on the same training dataset used in 'One Billion Word Language Model Benchmark' of Chelba *et al.* [7]. Each candidate is weighted according to the probability of trigram in SLM or that of predicting next word in RNN-LM.

In terms of SLM, the weight of each candidate is a sum of probabilities of three trigrams related to that candidate. For example, we have a phrase 'yield to thbse who are', and two candidates {'those', 'there'}, of the error 'thbse'. With the probability of ngram x as $p(x)$, the SLM weight of each candidate ($weight_{SLM}$) is calculated as follows:

$$weight_{SLM}(\text{`those'}) = p(\text{`yield to those'}) + p(\text{`to those who'}) + p(\text{`those who are'}) \quad (7.9)$$

$$weight_{SLM}(\text{‘there’}) = p(\text{‘yield to there’}) + p(\text{‘to there who’}) + p(\text{‘there who are’}) \quad (7.10)$$

For constructing RNN-LM, we apply one of the most common type of RNN - Long Short Term Memory (LSTM) [36]. The weight of each candidate is a sum of probabilities of predicting the next word related to that candidate. More specifically, LSTM needs a seed which is a context to predict a next word. The candidate can appear in the seed or be the next word.

To compare with trigram language model, we keep the total length of the seed and the next word to be three. For instance, we have the same phrase with SLM ‘yield to thbse who are’ with the same error candidate {‘those’, ‘there’} of the error ‘thbse’, the weight of each candidate (denoted as $weight_{LSTM}$) is computed as below:

$$\begin{aligned} weight_{LSTM}(\text{‘those’}) = {} & p(seed = \text{‘yield to’}, next\text{-}word = \text{‘those’}) \\ & + p(seed = \text{‘to those’}, next\text{-}word = \text{‘who’}) \\ & + p(seed = \text{‘those who’}, next\text{-}word = \text{‘are’}) \end{aligned} \quad (7.11)$$

where $p(seed = x, next\text{-}word = y)$ is the probability of predicting next word y given the previous word x.

$$\begin{aligned} weight_{LSTM}(\text{‘there’}) = {} & p(seed = \text{‘yield to’}, next\text{-}word = \text{‘there’}) \\ & + p(seed = \text{‘to there’}, next\text{-}word = \text{‘who’}) \\ & + p(seed = \text{‘there who’}, next\text{-}word = \text{‘are’}) \end{aligned} \quad (7.12)$$

Candidate Ranking based on a Regression Model (Step 3)

After generating candidates and weighting them at character level and word level in two previous steps, this step reuses these features and some complementary features to predict the confidence of each candidate becoming a correction by a regression model. Then such confidence is used for candidate ranking. This step consists of two parts: feature extraction and candidate ranking.

Feature extraction

Four important features used in our approach are selected and modified from a set of features of two related works [48, 66]. The first feature is 'Probability of 3-length sequences related to errors' is the modified version of context feature, suggested by both of the related works; in terms of [48], this feature is 'backward/forward bigram frequency'; in terms of [66], this is 'exact/relax-context popularity'. The second feature is 'Probability of ngram candidate', which is the general version of 'unigram frequency' of two prior works. Two last important features are features missing from each related work. As presented in Section 2.2.3, 'similarity feature' is an important feature used in error correction, which is ignored by [48]. Similarly, 'confusion probability' is successfully used in several post-processing approaches, but is ignored by [66]. As to other features, we cannot use them because of different reasons. In fact, due to lack of information from the dataset, 'word confidence' is ignored. We also remove the feature which is a part of other features, for example, 'lexicon existence', which is included in 'unigram frequency' feature. In addition, we refuse features that easily lead to bias such as 'term frequency in OCRed text'.

Let w_c be a candidate, C be a set of all candidates, and w_e be an OCR error. The details of each feature score are described in this section.

Probability of 3-length sequences related to errors: This feature (denoted as *relatedLM*) is the normalised weight of step 2 mentioned in Section 7.1.1, and is computed as below:

$$relatedLM(w_c, w_e) = \frac{weight_{LM}(w_c)}{\sum_{w'_c \in C} weight_{LM}(w'_c)} \tag{7.13}$$

where $weight_{LM}$ is either $weight_{SLM}$ or $weight_{LSTM}$.

Probability of ngram candidate: Candidate can be a single word or a sequence of multiple words, it means that candidate is word ngram. Instead of using the frequency of candidate and accepting 0 value if candidate is not in the training data, we use the probability of candidate in word ngram model which already applies smoothing techniques for solving sparsity problem. This feature (denoted as *candi-prob*) is the normalised proba-

bility of candidate in word ngram model, and is formulated as follows:

$$candi\text{-}prob(w_c, w_e) = \frac{p(w_c)}{\sum_{w'_c \in C} p(w'_c)} \tag{7.14}$$

where $p(w_c)$ is the probability of candidate w_c in word ngram model.

Longest common subsequence (LCS) is an alternative in qualifying the similarity between two strings. Two variations of LCS, including Normalised Longest Common Subsequence (NLCS) and Normalised Maximal Consecutive Longest Common Subsequence (NMCLCS) are proposed by Islam *et al.* [38].

The similarity of the two strings w_c and w_e (denoted as *similar*) is calculated as below:

$$similar(w_c, w_e) = \alpha * NLCS(w_c, w_e) + \sum_{i \in \{1,n,z\}} \alpha_i * NMCLCS_i(w_c, w_e) \tag{7.15}$$

where $NLCS$ and $NMCLCS_i$ are computed as in Equations 5.1, and 5.2; $\alpha = \alpha_i = 0.5$ weights of $NLCS$ and $NMCLCS_i$, which is recommended by Islam *et al.* [39].

Confusion probability: This feature (denoted as *conf-prob*) is the normalised weight of step 1 in Section 7.1.1, and is computed as below:

$$conf\text{-}prob(w_c, w_e) = \frac{p(w_c|w_e)}{\sum_{w'_c \in C} p(w'_c|w_e)} \tag{7.16}$$

Candidate ranking

A regression model is used for scoring candidates. For training and testing a regressor, if a candidate is a correction, its feature vector is labeled as 1. Otherwise, the feature vector is labeled as 0. Candidate with the highest confidence is suggested as the correction.

However, correcting run-on errors often produces irrelevant candidates which cause a big difference between corrected word and ground truth one. For example, post-processing tries to find the most suitable candidate from a dictionary for a run-on error ‘whereloan’, and suggests the top candidate such as ‘helios’ which is totally different from the GT ‘where I can’. Therefore, the final filter based on the edit distance between error and its top candidate decides whether use the top candidate or keep the error.

7.1.2 Experimental results

In this section, we perform the comparative evaluation of our approach. As mentioned in Section 3.1.2, English datasets of the ICDAR Post-OCR text correction competitions [8, 89] are used as benchmarks for the evaluation of our approaches.

Similarly, we use the percentage of improvement as the main evaluation metric. It is calculated based on the comparison of the original Levenshtein distance (between the OCR output and the GT) and the *avgDistance* distance (between the corrected output and the GT). The details of these metrics are described in Section 3.2.2.

By combining two ways of calculating step 1's weight and two ways of calculating step 2's weight, we consider four approaches in total: *typical-prob.SLM*, *modified-prob.SLM*, *typical-prob.LSTM*, and *modified-prob.LSTM*. The overall performance of our approaches is shown in Table 7.1.

Table 7.1: Overall performance, the relative improvement (%), of our statistical approaches over English datasets of the first competition ICDAR2017, 'x' denotes no improvement.

Approach	Monograph	Periodical
Baselines		
Char-SMT/NMT [3]	**43**	**37**
Single Char-SMT/NMT [3]	18	21
CLAM [8]	29	22
EFP [8]	13	x
MMDT [91]	20	x
WFST-PostOCR [8]	28	x
stat-correction-proposal		
typical-prob.SLM	22	3
modified-prob.SLM	**30**	**10**
typical-prob.LSTM	22	3
modified-prob.LSTM	**30**	**10**

Firstly, let us consider the dataset Monograph. In terms of performance of step 1, as mentioned in Section 7.1.1, our approach of calculating the confusion probability con-

sidered the common way in which erroneous characters appear, therefore our suggestion (*modified-prob*) strongly outperforms the typical approach (*typical-prob*) even with different techniques of step 2 (SLM or LSTM).

As to the performance of step 2, in erroneous OCR context, SLM has a comparative result to LSTM. However, if we consider the improvement percentage with one decimal value, SLM performs just slightly better (1.8% of relative increase of improvement in case of *typical-prob* and 2.4% in case of *modified-prob* on Monograph).

Regarding the performance of step 3, experimental results show that gradient boosting regression model [28] on top of decision trees with the least square loss function outperforms other regression models.

Among 10 participants of the correction task in the Post-OCR text correction ICDAR2017, only six teams can improve the distance with GT, and other teams almost deteriorate documents. These final results confirm the difficulty of post-processing step to deal with the noisy OCR output and the uncleaned GT.

Our best regression model (*modified-prob.SLM*) obtains higher performance than 9 teams. It should be emphasised that our approach is better than the multi-modular approach of statistical machine translation approach and spelling checker (MMDT) and the neural machine translation approach applied on post-OCR problem (CLAM), with around 10% and 1% absolutely increase, respectively.

Although our result is still lower than the one of the best-performing participant (Char-SMT/NMT), it is unfair to compare our multi-modular approach with the ensemble one. While our solution uses only one best regression model to score candidates, Char-SMT/NMT trains several models of both statistical and neural machine translation, and combines the top candidates generated from such models.

If we recommend top 5 candidates for each error, the best improvement percentage of our best approach *modified-prob.SLM* is 41%. It should be clear that the best improvement percentage is calculated based on the original distance and the best corrected distance (the distance of the most relevant candidate among the top n candidates with the GT word). In other words, in semi-automatic mode, our multi-modular approach can suggest correct candidates with the comparable performance with the ensemble approach of the winner.

Secondly, the dataset Periodical is taken into account. In our analyses of OCR errors (Chapter 5), we indicate that this dataset has higher percentage of real-word errors and more various mappings, which cause difficulties in generating and ranking candidates. Our post-processing approach obtains some improvements on Periodical, which only better than other statistical approaches (e.g., WFST-PostOCR, MMDT), but still much lower than the neural network based ones. These improvements confirm the mentioned-above conclusions, our *modified-prob* method outperforms the typical approach *typical-prob* and SLM is comparable to LSTM in erroneous context of OCRed text.

Lastly, concerning the dataset of the competition 2019, our correction approach does not get any improvements. It should be reminded that the second competition in 2019 is different from the one in 2017. Instead of being provided the list of error position, we have to use our list of errors. In our case, we apply the list obtained from our statistical detection approach (Section 6.1). In other words, the task is more difficult because correction results are dependent on not only correction techniques but also detection ones.

Furthermore, the dataset contains a high rate of real-word errors. More candidates should be generated at word level to handle real-word errors. In addition, among our three evaluation datasets, Comp2019 is the smallest and the one with the highest character error rate, and our regression model requires more data to predict relevant candidates.

7.2 Neural network based approach

As mentioned in Section 2.1, character-level machine translation is the current state of the art of error correction task. Regarding MT techniques, SMT consists of many small sub-components that are tuned separately. In contrast, NMT aims at building a single neural network which maximises the translation performance. Its performance is comparable to the existing state-of-the-art phrase-based model [102]. Consequently, we decided to use NMT at character level to translate OCRed text into its corrected version in the same language.

7.2.1 System description

We utilise an open-source toolkit for neural machine translation (OpenNMT) [49] to build our models. This toolkit is easy to use with handy guidelines and performs comparably with the high-performing NMT framework *Nematus* [92].

For hyperparameter setting, we use most of the default values of OpenNMT, except for embedding, hidden layer size, sequence length. Input and output texts are of the same language, therefore we configure to share embeddings between the source and target side with embedding size of 160 (tested against 100). Hidden layer size is increased from 500 to 1000. We set the maximum sequence length to 70 (instead of the default one, 50) to cover longer sequences of training data.

It is the fact that most of OCRed tokens are correct. If the MT system is trained on a dataset with a large proportion of valid tokens, then it might not rectify errors. In order to reduce the negative impact of imbalanced data and deal with real-word errors, we consider erroneous OCRed tokens and some nearby tokens (which can be correct or incorrect) as input; the corresponding GT texts are provided as output of NMT models.

Particularly, given one error and its four neighbors, we generate four word 5-grams which are represented at character level and used as input sequences. By doing this, we augment data for training NMT models. In the data representation, space and '#' are viewed as character delimiters and word boundary markers, respectively. If an error is a run-on one, '$' is used as word delimiter within its target text.

It should be noted that we do not consider an input sequence with all four words on the left side of the error and no word on its right side. The reason is that we expect to tackle incorrect split errors, such as 'main tain' vs. GT word 'maintain'.

For example, given an error 'andjust' in OCR phrase 'twenty in number andjust then published in a', and its corresponding GT 'twenty in number and just then published in a', four input sequences of the error and their output ones are shown in Table 7.2.

Furthermore, Sennrich *et al.* concluded that linguistic features (e.g., POS tags, morphological features, etc.) yield high performance of NMT systems. However, these features are specifically designed for words rather than characters. Instead of using such linguistic features, Amrhein *et al.* [3] applied two other features in their character-level NMT mod-

Table 7.2: Example of input/output sequences.

OCRed text (source side)
t w e n t y # i n # n u m b e r # a n d j u s t # t h e n
i n # n u m b e r # a n d j u s t # t h e n # p u b l i s h e d
n u m b e r # a n d j u s t # t h e n # p u b l i s h e d # i n
a n d j u s t # t h e n # p u b l i s h e d # i n # a
GT text (target side)
t w e n t y # i n # n u m b e r # a n d $ j u s t # t h e n
i n # n u m b e r # a n d $ j u s t # t h e n # p u b l i s h e d
n u m b e r # a n d $ j u s t # t h e n # p u b l i s h e d # i n
a n d $ j u s t # t h e n # p u b l i s h e d # i n # a

els, including the text types and the written time span. Nevertheless, both of the features are missing from Comp2019 dataset.

We think that OCRed texts of Comp2019 dataset might share some common characteristics, thus, our work considers the source of this dataset as its type. In total, there are three text types in the competition datasets (monograph and periodical from Comp2017, and Comp2019), which are exploited as additional input feature (or factor) for MT model. In OpenNMT, factors can be described in the input format with the symbol '|' as the beginning of a factor.

By applying factored NMT, we have more training data. Moreover, instead of training different models for each dataset, we only need to train a single model to test on our three datasets. An example of an input sequence of Monograph dataset with factored representation is shown in Table 7.3. Factored NMT model is the first version of our approach (denoted as Version 1).

MT techniques apply pre-trained word embeddings to improve translation performance. Several word embeddings are available and free to access while it is not easy to find a character embedding. McCann *et al.* [65] reported that a pre-trained encoder of a MT model increases the performance of other NLP tasks. Their contextualised word vectors are known as Context Vectors (CoVe). Broadening this idea, we extract embeddings from character-level NMT model trained with an aligned data.

Table 7.3: Example of an input sequence of Monograph dataset along with the feature document source M (Monograph).

OCRed text (source side)
n|M u|M m|M b|M e|M r|M #|M a|M n|M d|M j|M u|M s|M t|M #|M t|M h|M e|M n|M
GT text (target side)
n u m b e r # a n d $ j u s t # t h e n

Particularly, we align OCRed text with its corresponding GT text, then we generate input sequences from each aligned error with its contextual tokens. New character embeddings are extracted from models trained with the aligned data and shared embeddings between source and target side. It is expected that the embeddings (called as *aligned embeddings*) are able to put characters closer together in the vector space provided that they have similar contexts and/or shapes. The second version of our approach (called as Version 2) is similar to the first one but uses *aligned embeddings*.

As mentioned in Section 5.3, more than 80% of OCR errors have an edit distance less than 3. We can apply this feature to remove some irrelevant candidates. Specifically, after getting candidates for each error from MT models, we only select candidates which have edit distance with the error lower than 3. Furthermore, the analyses also indicate that percentage of deletion and insertion errors are much lower than that of substitution errors. While it is expensive to compute edit distance between two sequences, the length difference between candidate length and OCRed token length is simple and fast to calculate. We find that by setting the *length difference* threshold to 4, we obtain a performance comparable to using edit distance. The last version of our approach (denoted as Version 3) is the same as Version 2 with the addition of the *length difference* filter.

7.2.2 Experimental results

Our approach is evaluated on English datasets of the first two competitions with their evaluation metrics. Details of evaluation datasets as well as metrics are described in Sections 3.1.2 and 3.2.2. In the first competition, the organisers provided the predefined list of error positions. In contrast, the correction task of the second one is more challenging

as we need not only to identify error positions but also to rectify such detected errors. In this case, we use the list of errors detected by our neural network based detection approach (Section 6.2).

Performances of our approach on the competition datasets in ICDAR2017, and ICDAR2019 are shown in Tables 7.4 and 7.5, respectively. In general, the third version of our approach outperforms some of our counterparts as well as our statistical approaches.

Table 7.4: Overall performance, the relative improvement (%), of our neural network based approaches over English datasets of the Post-OCR text correction ICDAR2017, 'x' denotes no improvement.

Approach	Monograph	Periodical
Baselines		
Char-SMT/NMT [3]	**43**	**37**
Single Char-SMT/NMT [3]	18	21
CLAM [8]	29	22
EFP [8]	13	x
MMDT [91]	20	x
WFST-PostOCR [8]	28	x
stat-correction-proposal		
typical-prob.SLM	22	3
modified-prob.SLM	**30**	**10**
typical-probLSTM	22	3
modified-prob.LSTM	**30**	**10**
nn-correction-proposal		
Version 1	31	19
Version 2	32	20
Version 3	**36**	**27**

As to the first competition in ICDAR2017, our single model performs better than most of participants, except for the state-of-the-art approach (Char-SMT/NMT) which combines five different models of statistical MT and neural MT. The authors of the method [3] said that their combining system is complicated and hard to apply to new datasets. They

suggested the most promising single system which works across all data sets. However, its performance is significantly lower than the ensemble model with 18% and 21% improvement [3].

In opposite to the complex model of the winner, our model is easy to implement with the help of the open source OpenNMT. Moreover, it should be emphasised that our improvement is much higher than the neural translation based approach (CLAM) or multi-modular statistical based approach (MMDT) [91]. Consequently, we think that our model can be considered as one of the possible solutions to reduce OCR errors.

Table 7.5: Overall performance, the relative improvement (%), of our neural network based approaches over English datasets of the Post-OCR text correction ICDAR2019.

Approach	Comp2019
Baselines	
CCC [89]	**11**
CLAM [89]	0.4
CSIITJ [89]	2
RAE1 [89]	9
RAE2 [89]	6
UVA [89]	0
nn-correction-proposal	
Version 1	1
Version 2	2
Version 3	**4**

Regarding the second competition in ICDAR2019, without the provided error list, we have to use our list of errors obtained from our neural network based error detector (Section 6.2) in order to generate corresponding input and output sequences. Our best model still underperforms than some other methods, including RAE2, RAE1 and CCC.

In our opinion, the reason is that our models are built on the limited resources of the 2019 competition which is small and contains several real-word errors involving wrong line recognition. The RAE1, RAE2 competitors and the CCC team benefit from using external

materials like the Google Books Ngram Corpus. Nonetheless, there is no clear conclusion between the performance of our best model and that of RAE1&2 (called WFST-PostOCR) as the former performed better in the first competition.

7.3 Conclusions

In this chapter, we explored two ways of correcting OCR errors. The first one employs regression model to score candidates based on features extracted from the modified method of computing confusion probability and different types of language model. Our experiments show that SLM is comparable with LSTM in terms of 3-length sequence in erroneous OCR contexts. Our best model is comparative with those of participants of the competition in ICDAR2017, even with the neural network based model. However, there is a substantial decrease when dealing with datasets with several real-word errors.

Our second approach applies NMT techniques on contextual input data and some additional features (e.g., our character embedding, candidate filter) is promising to reduce OCR errors. Experimental results show that our best model outperforms several approaches of the teams of the competition in ICDAR2017. Nonetheless, if real-word errors caused by wrong line recognition like those in the competition dataset Comp2019 are taken into consideration, the performance of our approach is limited. Future work will focus on employing additional external resources to improve our results.

CHAPTER 8

Conclusions and future work

Previous chapters described different approaches to support accessing and exploring digitised output. Logical book structures are re-established for easy navigation. The quality of OCRed text is improved by detecting and correcting erroneous tokens. This chapter provides some conclusions as well as discusses future work.

8.1 Conclusions

Some limitations of OCR techniques lead to difficulties in exploiting digitised collections. One of them is that they typically produce textual digitised books only with physical structure information like paragraphs, pages, etc. It is not easy to navigate inside books or search information page by page, and this structure does not reflect well the semantic organisation of documents. Another problem is that the accuracy of modern OCR engines is reduced while converting old documents. Erroneous OCR output considerably impacts the way digital documents are indexed, accessed and explored. This thesis addresses these above problems by providing users with tables of contents for digitised documents and improving the quality of OCRed text.

Firstly, we introduce an aggregation method to enhance ToC extraction using system submissions from the first three competitions on book structure extraction. Our experimental results show that the union of two best approaches outperforms the existing approaches using both the title-based and link-based evaluation measures on the dataset of more than 2000 books. By efficiently combining the results of existing systems in an unsupervised way, we consistently outperform the state-of-the-art in book structure extraction, with statistically significant performance improvements. Our ensemble approach can work effectively providing that the outputs of relevant structure extraction are available.

Secondly, post-processing approaches are applied to improve the quality of OCRed texts. Although OCR errors share some common features with spelling errors, OCR errors have their own special characteristics. Thus, we examine different aspects of OCR errors towards a better understanding of OCR errors and related challenges. Based on our observations on four datasets we suggest some guidelines for designing post-processing approaches and apply them into our proposals.

Two high-efficient approaches are proposed on error detection task: i) error- and language-based model; ii) BERT based model. The first detection approach called *stat-detection-proposal* identifies erroneous tokens using feature values computed from its plausible candidate set. Experimental results show that several non-word errors can be detected and two novel features (*pe-index*, *tok-freq*) are useful in classifying errors. However, the method has some limitations in detecting real-word errors. In addition, it requires to generate the plausible candidate set for each OCRed token, nonetheless, it should be noted that the set can be reused in the correction part.

The second detection approach named *nn-detection-proposal* classifies errors based on fine-tuning BERT models with pre-trained word embeddings. Thanks to the contextual language model BERT, this proposal is able to handle real-word errors and it is unnecessary to generate candidates for each token. Nevertheless, it is dependent on the pre-trained BERT model which needs to be trained for several hours on large datasets.

Similar to the detection part, we present two ways of correcting OCR errors. The first correction approach is to predict the relevant candidates by using regression model. It is worth noting that features extracted from the modified method of computing confusion

probability and different types of language model are employed as the criterion to score the candidates. Our best model is comparative with participants of the competition in ICDAR2017. Nonetheless, there is a substantial drop in detecting errors on datasets with several real-word errors.

Our second correction proposal applies neural machine translation techniques. By considering more context, and utilising advance features of NMT, this method can rectify not only non-word but also real-word errors. Numerical results show that our best model outperforms some approaches, including other NMT based ones. However, real-word errors which are caused by wrong line recognition like those in the competition dataset Comp2019 are still problematic.

8.2 Future work

This thesis reports some supportive approaches to browse inside books and to increase the usability of digitised books. The advantages and disadvantages of each method are clarified in each chapter. Our future work will focus on eliminating such disadvantages from our next approaches.

Regarding book structure extraction problem, as more than 80% of books has physical ToC pages, it is obvious that more attention should be payed for this kind of books. Besides methods of the ToC-recognition-based type, approaches of the book-content-based kind can be applied on books with ToC as well. It is apparent that this is a double verification on extracting ToC on books with ToC, which enables to improve performance of book structure extraction.

As to error detection task, the *stat-detection-proposal* shows some benefits in detecting errors but it is time-consuming to generate the list of candidates. Further work can focus on speeding up the current algorithm of candidate generation. The *nn-detection-proposal* outperforms other approaches, but it depends on BERT models. Hence, it is interesting to design more complex models on the top of BERT models or a new neural network model which takes into account specific features of OCR errors.

Concerning error correction task, real-word errors are always a big problem especially

with the *stat-correction-proposal*. The heart of the statistical approach is, of course, the candidate generation. Besides candidates at character level, it is reasonable to consider candidates at word level as well. More contextual features should be taken into account to deal with real-word errors.

Applying machine translation techniques to post-OCR processing is the state-of-the-art approaches. We will integrate the post-filter into the NMT model. Our *nn-correction-proposal* can be upgraded with some modifications based on characteristics of OCR errors. Experimental results show that our neural network based approach is comparable to other approaches, but, it has some limitations in dealing with real-word errors like wrong line recognition errors. Thus, one feasible way is to provide more contextual information to our models, such as fine-tuned BERT models or Google Books Ngrams. Besides machine translation techniques, applying sequence-to-sequence approaches or sequence generation techniques like Sequence Generative Adversarial Nets with Policy Gradient (SeqGAN) [104] is also a promising solution.

In our next step, ensemble approaches will be considered to take advantages from various detection and correction methods. Our post-OCR methods can be expanded to benefit from not only natural language processing resources but also information of OCR software. Furthermore, multi-modal learning model might be exploited to incorporate text generation and post-processing.

The post-processing approaches could be further adapted to solve other problems of string-string transformation, for example, spelling normalisation that converts historical spellings into corresponding modern word forms, post-processing on textual outputs of Automatic Speech Recognition which enables the recognition and translation of spoken language into text by computers, or on transcriptions of handwritten documents.

Bibliography

[1] Haithem Afli, Loïc Barrault, and Holger Schwenk. "OCR Error Correction Using Statistical Machine Translation." In: *Int. J. Comput. Linguistics Appl.* (2016), pp. 175–191.

[2] Haithem Afli, Zhengwei Qiu, Andy Way, and Páraic Sheridan. "Using SMT for OCR error correction of historical texts". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. 2016, pp. 962–966.

[3] Chantal Amrhein and Simon Clematide. "Supervised OCR Error Detection and Correction Using Statistical and Neural Machine Translation Methods". In: *Journal for Language Technology and Computational Linguistics (JLCL)* 33 (Jan. 2018).

[4] Chloé Artaud, Nicolas Sidere, Antoine Doucet, Jean-Marc Ogier, and Vincent Poulain D'Andecy Yooz. "Find it! Fraud Detection Contest Report". In: *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE. 2018, pp. 13–18.

[5] Youssef Bassil and Mohammad Alwani. "Ocr post-processing error correction algorithm using google online spelling suggestion". In: *Journal of Emerging Trends in Computing and Infor- mation Sciences* (2012).

[6] Federico Boschetti, Matteo Romanello, Alison Babeu, David Bamman, and Gregory Crane. "Improving OCR accuracy for classical critical editions". In: *International*

Conference on Theory and Practice of Digital Libraries. Springer. 2009, pp. 156–167.

[7] Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. "One billion word benchmark for measuring progress in statistical language modeling". In: (2013).

[8] Guillaume Chiron, Antoine Doucet, Mickaël Coustaty, and Jean-Philippe Moreux. "ICDAR2017 competition on post-OCR text correction". In: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. Vol. 1. IEEE. 2017, pp. 1423–1428.

[9] Guillaume Chiron, Antoine Doucet, Mickaël Coustaty, Muriel Visani, and Jean-Philippe Moreux. "Impact of OCR errors on the use of digital libraries: towards a better access to information". In: *2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*. IEEE. 2017, pp. 1–4.

[10] Otto Chrons and Sami Sundell. "Digitalkoot: Making old archives accessible using crowdsourcing". In: *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*. 2011.

[11] Kenneth W. Church and William A. Gale. "Probability scoring for spelling correction". In: *Statistics and Computing* 1.2 (1991), pp. 93–103.

[12] Simon Clematide, Lenz Furrer, and Martin Volk. "Crowdsourcing an OCR gold standard for a German and French heritage corpus". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. 2016, pp. 975–982.

[13] WB Croft, SM Harding, K Taghva, and J Borsack. "An evaluation of information retrieval accuracy with simulated OCR output". In: *Symposium on Document Analysis and Information Retrieval*. 1994, pp. 115–126.

[14] Fred J Damerau. "A technique for computer detection and correction of spelling errors". In: *Communications of the ACM* 7.3 (1964), pp. 171–176.

[15] Mark Davies. “The Corpus of Contemporary American English as the first reliable monitor corpus of English”. In: *Literary and linguistic computing* (2010), pp. 447–464.

[16] Hervé Déjean. “Using page breaks for book structuring”. In: *International Workshop of the Initiative for the Evaluation of XML Retrieval*. Springer. 2011, pp. 57–67.

[17] Hervé Déjean and Jean-Luc Meunier. “On tables of contents and how to recognize them”. In: *International Journal of Document Analysis and Recognition (IJDAR)* 12.1 (2009).

[18] Hervé Déjean and Jean-Luc Meunier. “XRCE participation to the 2009 book structure task”. In: *International Workshop of the Initiative for the Evaluation of XML Retrieval*. Springer. 2009.

[19] Hervé Déjean and Jean-Luc Meunier. “Reflections on the inex structure extraction competition”. In: *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*. ACM. 2010.

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).

[21] Antoine Doucet. “Logical Structure Extraction from Digitized Books: Benchmarking State-of-the-Art Systems”. In: Apr. 2018, pp. 3–28.

[22] Antoine Doucet, Gabriella Kazai, and Jean-Luc Meunier. “ICDAR 2011 book structure extraction competition”. In: *Proceedings of the Eleventh International Conference on Document Analysis and Recognition*. IEEE. 2011.

[23] Antoine Doucet, Gabriella Kazai, Bodin Dresevic, Aleksandar Uzelac, Bogdan Radakovic, and Nikola Todic. “ICDAR 2009 Book Structure Extraction Competition”. In: *Proceedings of the Tenth International Conference on Document Analysis and Recognition*. Barcelona, Spain, 2009.

[24] Antoine Doucet, Gabriella Kazai, Bodin Dresevic, Aleksandar Uzelac, Bogdan Radakovic, and Nikola Todic. "Setting up a competition framework for the evaluation of structure extraction from ocr-ed books". In: *International Journal on Document Analysis and Recognition* 14.1 (2011).

[25] Antoine Doucet, Gabriella Kazai, Sebastian Colutto, and Günter Mühlberger. "ICDAR 2013 Competition on Book Structure Extraction". In: *Proceedings of the Twelfth International Conference on Document Analysis and Recognition*. IEEE. 2013.

[26] Bodin Dresevic, Aleksandar Uzelac, Bogdan Radakovic, and Nikola Todic. "Book layout analysis: TOC structure extraction engine". In: *International Workshop of the Initiative for the Evaluation of XML Retrieval*. Springer. 2008.

[27] John Evershed and Kent Fitch. "Correcting noisy OCR: Context beats confusion". In: *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage*. ACM. 2014, pp. 45–51.

[28] Jerome H Friedman. "Greedy function approximation: a gradient boosting machine". In: *Annals of statistics* (2001), pp. 1189–1232.

[29] Lenz Furrer and Martin Volk. "Reducing OCR Errors in Gothic-Script Documents". In: *Language Technologies for Digital Humanities and Cultural Heritage* (2011), p. 97.

[30] Lukas Gander, Cornelia Lezuo, and Raphael Unterweger. "Rule based document understanding of historical books using a hybrid fuzzy classification system". In: *Proceedings of the 2011 Workshop on Historical Document Imaging and Processing*. ACM. 2011.

[31] Emmanuel Giguet, Alexandre Baudrillart, and Nadine Lucas. "Resurgence for the book structure extraction competition". In: *INEX 2009 Workshop Pre-Proceedings*. Citeseer. 2009, pp. 136–142.

[32] Emmanuel Giguet and Nadine Lucas. "The book structure extraction competition with the resurgence software at caen university". In: *International Workshop of the Initiative for the Evaluation of XML Retrieval*. Springer. 2009.

[33] Annette Gotscharek, Ulrich Reffle, Christoph Ringlstetter, and Klaus U Schulz. "On lexical resources for digitization of historical documents". In: *Proceedings of the 9th ACM symposium on Document engineering*. ACM. 2009, pp. 193–200.

[34] Paul Hagon. "Trove crowdsourcing behaviour". In: *Australian Library & Information Association Information Online 2013 Proceedings* (2013).

[35] Ahmed Hamdi, Axel Jean-Caurant, Nicolas Sidere, Mickaël Coustaty, and Antoine Doucet. "An Analysis of the Performance of Named Entity Recognition over OCRed Documents". In: *2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*. IEEE. 2019, pp. 333–334.

[36] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[37] Rose Holley. *Many hands make light work: Public collaborative OCR text correction in Australian historic newspapers*. National Library of Australia, 2009.

[38] Aminul Islam and Diana Inkpen. "Semantic text similarity using corpus-based word similarity and string similarity". In: *ACM Transactions on Knowledge Discovery from Data* (2008).

[39] Aminul Islam and Diana Inkpen. "Real-word spelling correction using Google Web IT 3-grams". In: *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*. 2009, pp. 1241–1249.

[40] RichardSocher JeffreyPennington and ChristopherD Manning. "Glove: Global vectors for word representation". In: Citeseer.

[41] Hongyan Jing, Daniel Lopresti, and Chilin Shih. "Summarization of noisy documents: a pilot study". In: *Proceedings of the HLT-NAACL 03 on Text summarization workshop-Volume 5*. Association for Computational Linguistics. 2003, pp. 25–32.

[42] M. A. Jones, G. A. Story, and B. W. Ballard. "Interating multiple knowledge sources in a bayesian ocr post-processor". In: *International Journal on Document Analysis and Recognition*. 1991, 925–933.

[43] Gabriella Kazai, Antoine Doucet, and Monica Landoni. "Overview of the INEX 2008 Book Track". In: *Advances in Focused Retrieval*. Ed. by Shlomo Geva, Jaap Kamps, and Andrew Trotman. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 106–123.

[44] Gabriella Kazai, Antoine Doucet, Marijn Koolen, and Monica Landoni. "Overview of the INEX 2009 book track". In: *International Workshop of the Initiative for the Evaluation of XML Retrieval*. Springer, pp. 145–159.

[45] Gitansh Khirbat. "OCR Post-Processing Text Correction using Simulated Annealing (OPTeCA)". In: *Proceedings of the Australasian Language Technology Association Workshop 2017*. 2017, pp. 119–123.

[46] Khurram Khurshid. "Analysis and retrieval of historical documents images". PhD thesis. Jan. 2009.

[47] Scott Kirkpatrick, C Daniel Gelatt, and Mario P Vecchi. "Optimization by simulated annealing". In: *science* 220.4598 (1983), pp. 671–680.

[48] Ido Kissos and Nachum Dershowitz. "OCR error correction using character correction and feature-based word classification". In: *Document Analysis Systems (DAS), 2016 12th IAPR Workshop on*. IEEE. 2016, pp. 198–203.

[49] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. "OpenNMT: Open-Source Toolkit for Neural Machine Translation". In: *Proceedings of ACL 2017, System Demonstrations*. 2017, pp. 67–72.

[50] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. "Moses: Open source toolkit for statistical machine translation". In: 2007.

[51] Okan Kolak, William Byrne, and Philip Resnik. "A generative probabilistic OCR model for NLP applications". In: *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. 2003, pp. 55–62.

[52] Okan Kolak and Philip Resnik. "OCR error correction using a noisy channel model". In: *Proceedings of the second international conference on Human Language Technology Research*. Morgan Kaufmann Publishers Inc. 2002, pp. 257–262.

[53] Karen Kukich. "Spelling correction for the telecommunications network for the deaf". In: *Communications of the ACM* 35.5 (1992), pp. 80–90.

[54] Karen Kukich. "Techniques for automatically correcting words in text". In: *Acm Computing Surveys (CSUR)* 24.4 (1992), pp. 377–439.

[55] Thomas K Landauer and Lynn A Streeter. "Structural differences between common and rare words: Failure of equivalence assumptions for theories of word recognition". In: *Journal of Memory and Language* 12.2 (1973), p. 119.

[56] Guillaume Lazzara, Roland Levillain, Thierry Géraud, Yann Jacquelet, Julien Marquegnies, and Arthur Crépin-Leblond. "The SCRIBO module of the Olena platform: a free software framework for document image analysis". In: *2011 International Conference on Document Analysis and Recognition*. IEEE. 2011, pp. 252–258.

[57] Chun Chen Lin, Yosihiro Niwa, and Seinosuke Narita. "Logical structure analysis of book document images using contents information". In: *Proceedings of the Fourth International Conference on Document Analysis and Recognition*. Vol. 2. IEEE. 1997, pp. 1048–1054.

[58] Xiaofan Lin. "Reliable OCR solution for digital content re-mastering". In: *Document Recognition and Retrieval IX*. Vol. 4670. International Society for Optics and Photonics. 2001, pp. 223–231.

[59] Caihua Liu, Jiajun Chen, Xiaofeng Zhang, Jie Liu, and Yalou Huang. "TOC structure extraction from ocr-ed books". In: *International Workshop of the Initiative for the Evaluation of XML Retrieval*. Springer. 2011.

[60] Rafael Llobet, Jose-Ramon Cerdan-Navarro, Juan-Carlos Perez-Cortes, and Joaquim Arlandis. "OCR post-processing using weighted finite-state transducers". In: *2010 International Conference on Pattern Recognition*. IEEE. 2010, pp. 2021–2024.

[61] Daniel Lopresti and Jiangying Zhou. "Using consensus sequence voting to correct OCR errors". In: *Computer Vision and Image Understanding* 67.1 (1997), pp. 39–47.

[62] William B Lund, Douglas J Kennard, and Eric K Ringger. "Combining multiple thresholding binarization values to improve OCR output". In: *Document Recognition and Retrieval XX*. Vol. 8658. International Society for Optics and Photonics. 2013, 86580R.

[63] William B Lund and Eric K Ringger. "Improving optical character recognition through efficient multiple system alignment". In: *Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*. ACM. 2009, pp. 231–240.

[64] William B Lund, Daniel D Walker, and Eric K Ringger. "Progressive alignment and discriminative error correction for multiple OCR engines". In: *2011 International Conference on Document Analysis and Recognition*. IEEE. 2011, pp. 764–768.

[65] Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. "Learned in translation: Contextualized word vectors". In: *Advances in Neural Information Processing Systems*. 2017, pp. 6294–6305.

[66] Jie Mei, Aminul Islam, Yajing Wu, Abidalrahman Moh'd, and Evangelos E Milios. "Statistical Learning for OCR Text Correction". In: *arXiv preprint arXiv:1611.06950* (2016).

[67] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. "Recurrent neural network based language model". In: *Eleventh Annual Conference of the International Speech Communication Association*. 2010.

[68] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. "Advances in Pre-Training Distributed Word Representations". In: *Proceed-*

ings of the International Conference on Language Resources and Evaluation (LREC 2018). 2018.

[69] David Miller, Sean Boisen, Richard Schwartz, Rebecca Stone, and Ralph Weischedel. "Named entity extraction from noisy input: speech and OCR". In: *Proceedings of the sixth conference on Applied natural language processing*. Association for Computational Linguistics. 2000, pp. 316–324.

[70] Elke Mittendorf and Peter Schäuble. "Information retrieval can cope with many errors". In: *Information Retrieval* 3.3 (2000), pp. 189–216.

[71] Robert Morris and Lorinda L Cherry. "Computer detection of typographical errors". In: *IEEE Transactions on Professional Communication* (1975), pp. 54–56.

[72] Stephen Mutuvi, Antoine Doucet, Moses Odeo, and Adam Jatowt. "Evaluating the impact of OCR errors on topic modeling". In: *International Conference on Asian Digital Libraries*. Springer. 2018, pp. 3–14.

[73] Thi Tuyet Hai Nguyen, Antoine Doucet, and Mickaël Coustaty. "Enhancing Table of Contents Extraction by System Aggregation". In: *14th IAPR International Conference on Document Analysis and Recognition, ICDAR 2017, Kyoto, Japan*. 2017, pp. 242–247.

[74] Thi Tuyet Hai Nguyen, Mickaël Coustaty, Antoine Doucet, Adam Jatowt, and Nhu-Van Nguyen. "Adaptive Edit-Distance and Regression Approach for Post-OCR Text Correction". In: *Maturity and Innovation in Digital Libraries - 20th International Conference on Asia-Pacific Digital Libraries, ICADL 2018, Hamilton, New Zealand*. 2018, pp. 278–289.

[75] Thi Tuyet Hai Nguyen, Mickaël Coustaty, Antoine Doucet, Adam Jatowt, and Nhu-Van Nguyen. "Adaptive Edit-Distance and Regression Approach for Post-OCR Text Correction". In: *Maturity and Innovation in Digital Libraries - 20th International Conference on Asia-Pacific Digital Libraries*. 2018, pp. 278–289.

[76] Thi Tuyet Hai Nguyen, Adam Jatowt, Mickaël Coustaty, Nhu-Van Nguyen, and Antoine Doucet. "Deep Statistical Analysis of OCR Errors for Effective Post-OCR Processing". In: *19th ACM/IEEE Joint Conference on Digital Libraries, JCDL 2019, Champaign, IL, USA*. 2019, pp. 29–38.

[77] Thi Tuyet Hai Nguyen, Mickaël Coustaty, Antoine Doucet, Adam Jatowt, and Nhu-Van Nguyen. "Post-OCR Error Detection by Generating Plausible Candidates". In: *15th IAPR International Conference on Document Analysis and Recognition, ICDAR 2019, Sydney, Australia*. 2019.

[78] Kai Niklas. "Unsupervised post-correction of ocr errors". In: (2010).

[79] Richard Pearce-Moses. *A glossary of archival and records terminology*. 2013.

[80] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[81] Juan Carlos Perez-Cortes, Juan-Carlos Amengual, Joaquim Arlandis, and Rafael Llobet. "Stochastic error-correcting parsing for OCR post-processing". In: *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*. Vol. 4. IEEE. 2000, pp. 405–408.

[82] James L Peterson. "A note on undetected typing errors". In: *Communications of the ACM* 29.7 (1986), pp. 633–637.

[83] Michael Piotrowski. "Natural language processing for historical texts". In: *Synthesis lectures on human language technologies* 5.2 (2012), pp. 1–157.

[84] Antonio Reyes, Paolo Rosso, and Tony Veale. "A multidimensional approach for detecting irony in twitter". In: *Language resources and evaluation* (2013), pp. 239–268.

[85] Martin Reynaert. "Text induced spelling correction". In: *Proceedings of the 20th international conference on Computational Linguistics*. Association for Computational Linguistics. 2004, p. 834.

[86] Martin Reynaert. "Corpus-Induced Corpus Clean-up." In: *LREC 2006: Fifth International Conference on Language Resources and Evalua- tion*. Citeseer. 2006.

[87] Martin Reynaert. "Non-interactive OCR post-correction for giga-scale digitization projects". In: *International Conference on Intelligent Text Processing and Computational Linguistics*. Springer. 2008, pp. 617–630.

[88] Martin WC Reynaert. "Character confusion versus focus word-based correction of spelling and OCR variants in corpora". In: *International Journal on Document Analysis and Recognition (IJDAR)* 14.2 (2011), pp. 173–187.

[89] Christophe Rigaud, Antoine Doucet, Mickaël Coustaty, and Jean-Philippe Moreux. "ICDAR 2019 Competition on Post-OCR Text Correction". In: *15th IAPR International Conference on Document Analysis and Recognition, ICDAR 2019, Sydney, Australia*. 2019.

[90] Christoph Ringlstetter, Klaus U Schulz, and Stoyan Mihov. "Adaptive text correction with Web-crawled domain-dependent dictionaries". In: *ACM Transactions on Speech and Language Processing (TSLP)* 4.4 (2007), p. 9.

[91] Sarah Schulz and Jonas Kuhn. "Multi-modular domain-tailored OCR post-correction". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 2017, pp. 2716–2726.

[92] Rico Sennrich, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, et al. "Nematus: a Toolkit for Neural Machine Translation". In: *Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics*. 2017, pp. 65–68.

[93] Uwe Springmann, Christian Reul, Stefanie Dipper, and Johannes Baiter. "Ground Truth for training OCR engines on historical documents in German Fraktur and Early Modern Latin". In: *JLCL* (2018), p. 97.

[94] Christian M Strohmaier, Christoph Ringlstetter, Klaus U Schulz, and Stoyan Mihov. "Lexical postcorrection of OCR-results: The web as a dynamic secondary dictionary?" In: *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.* Citeseer. 2003, pp. 1133–1137.

[95] Kazem Taghva and Eric Stofsky. "OCRSpell: an interactive spelling correction system for OCR errors in text". In: *International Journal on Document Analysis and Recognition* 3.3 (2001), pp. 125–137.

[96] Kazem Taghva, Julie Borsack, Allen Condit, and Srinivas Erva. "The effects of noisy data on text retrieval". In: *Journal of the American Society for Information Science* 45.1 (1994), pp. 50–58.

[97] Xiang Tong and David A Evans. "A statistical approach to automatic OCR error correction in context". In: *Fourth Workshop on Very Large Corpora.* 1996.

[98] Shinji Tsuruoka, Chihiro Hirano, Tomohiro Yoshikawa, and Tsuyoshi Shinogi. "Image-based structure analysis for a table of contents and conversion to XML documents". In: *Workshop on document layout interpretation and its application (DLIA 2001).* Citeseer. 2001.

[99] Roelof Van Zwol and Tim Van Loosbroek. "Effective use of semantic structure in XML retrieval". In: *European Conference on Information Retrieval.* Springer. 2007, pp. 621–628.

[100] Luis Von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. "recaptcha: Human-based character recognition via web security measures". In: *Science* 321.5895 (2008), pp. 1465–1468.

[101] Jacques Wainer. "Comparison of 14 different families of classification algorithms on 115 binary datasets". In: *arXiv preprint arXiv:1606.00930* (2016).

[102] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. "Google's neural machine translation system: Bridging the gap between human and machine translation". In: *arXiv preprint arXiv:1609.08144* (2016).

[103] Zhaohui Wu, Prasenjit Mitra, and C Lee Giles. "Table of contents recognition and extraction for heterogeneous book documents". In: *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*. IEEE. 2013.

[104] L Yu, W Zhang, J Wang, and Y Yu. "Seqgan: sequence generative adversarial nets with policy gradient". In: *AAAI-17: Thirty-First AAAI Conference on Artificial Intelligence*. Vol. 31. Association for the Advancement of Artificial Intelligence (AAAI). 2017, pp. 2852–2858.