

**EFFICIENT REPRESENTATION AND MATCHING OF
TEXTS AND IMAGES IN SCANNED BOOK
COLLECTIONS**

A Dissertation Presented

by

ISMET ZEKI YALNIZ

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

February 2014

School of Computer Science

© Copyright by Ismet Zeki Yalniz 2013

All Rights Reserved

**EFFICIENT REPRESENTATION AND MATCHING OF
TEXTS AND IMAGES IN SCANNED BOOK
COLLECTIONS**

A Dissertation Presented

by

ISMET ZEKI YALNIZ

Approved as to style and content by:

R. Manmatha, Chair

James Allan, Member

W. Bruce Croft, Member

Patrick A. Kelly, Member

Lori A. Clarke, Department Chair
School of Computer Science

To my parents.

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor, R. Manmatha. He put an incredible effort to make this dissertation possible despite all the hardships in his life. Manmatha made me grasp the importance of conducting research with practical aspects and communicating the research results effectively. His unique research style has a profound impact on my research work and world view.

I would also like to thank my committee members: James Allan, W. Bruce Croft and Patrick A. Kelly. Their valuable comments and encouragement excelled this work in many ways. I also thank David A. Smith for his valuable discussions and suggestions on my research work.

Special thanks go to all the graduate students and alumni in the CIIR lab for their friendship and collaboration: Elif Aktolga, Niranjana Balasubramanian, Marc Cartright, Van Dang, Jeff Dalton, Shiri Dori-Hacohen, Henry Feild, Sam Huston, Myung-ha Jang, Jin Young Kim, Kriste Krstovski, Chia-Jung Lee, Tamsin Maxwell, Venkatesh Narasimha Murthy, Jae Hyun Park, Jangwon Seo, David Wemhoener, Xiaobing Xue, Xing Yi and everyone else. Additional thanks to Ethem Can for helping me with the experiments and figures for my partial duplicate detection work.

I am sincerely indebted to all CSCF and CIIR staff members for their support of my work, especially David Fisher, Jean Joyce, Leeanne M. Leclerc, Kate Morruzzi and Dan Parker.

Finally, I would like to thank the most important people in my life, my family. I thank my parents, Ali Rıza and Elif, for their unlimited encouragement and support throughout my studies. I thank my brother, Mehmet Akif, and sisters, Eda and

Yasemin, for always being there for me and giving me positive energy. I would not be able to make it without their support.

The work presented here was supported in part by the Center for Intelligent Information Retrieval, in part by NSF grant #IIS-0910884, and in part by a grant from the Andrew W. Mellon Foundation.

ABSTRACT

EFFICIENT REPRESENTATION AND MATCHING OF TEXTS AND IMAGES IN SCANNED BOOK COLLECTIONS

FEBRUARY 2014

ISMET ZEKI YALNIZ

B.Sc., BILKENT UNIVERSITY

M.Sc., BILKENT UNIVERSITY

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor R. Manmatha

Millions of books from public libraries and private collections have been scanned by various organizations in the last decade. The motivation is to preserve the written human heritage in electronic format for durable storage and efficient access. The information buried in these large book collections has always been of major interest for scholars from various disciplines. Several interesting research problems can be defined over large collections of scanned books given their corresponding optical character recognition (OCR) outputs. At the highest level, one can view the entire collection as a whole and discover interesting contextual relationships or linkages between the books. A more traditional approach is to consider each scanned book separately and perform information search and mining at the book level. Here we also show that one can view each book as a whole composed of chapters, sections, paragraphs, sentences, words or even characters positioned in a particular sequential order sharing the same

global context. The information inherent in the entire context of the book is referred to as *global information* and it is demonstrated by addressing a number of research questions defined for scanned book collections.

The global sequence information is one of the different types of global information available in textual documents. It is useful for discovering content overlap and similarity across books. Each book has a specific flow of ideas and events which distinguishes it from other books. If this global order is changed, then the flow of events and consequently the story changes completely. This argument is true across document translations as well. Although the local order of words in a sentence might not be preserved after translation, sentences, paragraphs, sections and chapters are likely to follow the same global order. Otherwise the two texts are not considered to be translations of each other.

A global sequence alignment approach is therefore proposed to discover the contextual similarity between the books. The problem is that conventional sequence alignment algorithms are slow and not robust for book length documents especially with OCR errors, additional or missing content. Here we propose a general framework which can be used to efficiently align and compare the textual content of the books at various coarseness levels and even across languages. In a nut-shell, the framework uses the sequence of words which appear only once in the entire book (referred to as “the sequence of unique words”) to represent the text. This representation is compact and it is highly descriptive of the content along with the global word sequence information. It is shown to be more accurate compared to the state of the art for efficiently i) detecting which books are partial duplicates in large scanned book collections (DUPNIQ), and, ii) finding which books are translations of each other without explicitly translating the entire texts using statistical machine translation approaches (TRANSNIQ).

Using the global order of unique words and their corresponding positions in the text, one can also generate the complete text alignment efficiently using a recursive approach. The Recursive Text Alignment Scheme (RETAS) is several orders of magnitude faster than the conventional sequence alignment approaches for long texts and it is later used for iii) the automatic evaluation of OCR accuracy of books given the OCR outputs and the corresponding electronic versions, iv) mapping the corresponding portions of the two books which are known to be partial duplicates, and finally it is generalized for v) aligning long noisy texts *across* languages (Recursive Translation Alignment - RTA).

Another example of the global information is that books are mostly printed in a single global font type. Here we demonstrate that the global font feature along with the letter sequence information can be used for facilitating and/or improving text search in noisy page images. There are two contributions in this area: (vi) an efficient word spotting framework for searching text in noisy document images, and, (vii) a state of the art dependence model approach to resolve arbitrary text queries using visual features. The effectiveness of these approaches is demonstrated for books printed in different scripts for which there is no OCR engine available or the recognition accuracy is low.

TABLE OF CONTENTS

	Page
ACKNOWLEDGMENTS	v
ABSTRACT	vii
LIST OF TABLES	xiv
LIST OF FIGURES	xviii
 CHAPTER	
1. INTRODUCTION	1
2. LITERATURE OVERVIEW	12
2.1 Alignment of long noisy texts	13
2.2 Duplicate detection	17
2.3 Translation detection	19
2.4 Aligning texts across languages	20
2.5 Searching text in noisy document images	23
3. ALIGNMENT OF LONG NOISY TEXTS	26
3.1 The Recursive Text Alignment Scheme	27
3.1.1 The properties of unique words	27
3.1.2 Alignment of unique words	30
3.1.3 The recursive stage	31
3.1.4 Word and character level alignment	33
3.2 Verification of the proposed alignment scheme	34
3.3 Computational complexity	38
3.4 Efficiency	38
3.5 Evaluation of OCR accuracy for real scanned books	39

4. PARTIAL DUPLICATE DETECTION FOR LARGE SCANNED BOOK COLLECTIONS	43
4.1 The proposed framework	49
4.1.1 The document representation	49
4.1.2 Scoring schemes for document pairs	51
4.1.2.1 Correlation Score (CS)	51
4.1.2.2 Information Theoretic Score (ITS)	51
4.2 Duplicate detection experiments	53
4.2.1 Datasets	53
4.2.2 Baselines	54
4.2.3 Evaluation of duplicate detection results	57
4.2.4 Word sampling experiments	59
4.2.5 Synthetic experiments	61
4.2.6 Efficiency	65
4.3 Mapping duplicated portions of texts	67
5. FINDING TRANSLATIONS IN SCANNED BOOK COLLECTIONS	70
5.1 The proposed framework	73
5.2 Experiments	75
5.2.1 Datasets	76
5.2.2 Evaluation metrics	77
5.2.3 Baselines	79
5.2.4 EUROPARL experiments	80
5.2.5 Experiments with real scanned books	86
5.2.6 Synthetic Experiments	90
5.2.7 Efficiency	94
6. ALIGNING LONG NOISY TEXTS ACROSS LANGUAGES	96
6.1 The proposed framework	101
6.1.1 Recursive Translation Alignment (RTA)	101
6.1.2 Passage level alignment	104
6.2 Experiments	105
6.2.1 Datasets	105
6.2.2 Baselines	107

6.2.3	Experiments with the EUROPARL dataset	109
6.2.4	Experiments with Gutenberg books	110
6.2.5	Synthetic experiments	113
6.2.6	Experiments with real scanned books	115
6.3	Mapping translated portions of texts	118
7.	SEARCHING DOCUMENTS USING IMAGE FEATURES	120
7.1	Visual features	122
7.2	An efficient word spotting approach for noisy document images	124
7.2.1	The proposed framework	124
7.2.1.1	Coverage analysis	125
7.2.1.2	Configuration analysis	126
7.2.2	Experiments	128
7.2.2.1	Datasets	128
7.2.2.2	Evaluation	129
7.3	A Discrete MRF Model for Searching Arbitrary Text in Document Images	132
7.3.1	The general MRF framework	132
7.3.2	The proposed approach	134
7.3.2.1	Modeling visterm-letter bigram dependencies	135
7.3.2.2	Modeling the order of letter bigrams	137
7.3.2.3	Probability estimation	138
7.3.2.4	Indexing letter bigrams	144
7.3.3	Query Resolution	144
7.3.3.1	Morphological expansion	145
7.3.3.2	Stemming	147
7.3.4	Experiments	147
7.3.4.1	Datasets	148
7.3.4.2	Training	152
7.3.4.3	Latin script experiments	156
7.3.4.4	Telugu script experiments	162
7.3.4.5	Ottoman script experiments	167
8.	CONCLUSIONS AND FUTURE WORK	175

BIBLIOGRAPHY 180

LIST OF TABLES

Table	Page
2.1 Longest Common Subsequences illustrated for two words “NEIHBOURHOOD” and “NEIGHBORHOOD”. Matching characters are indicated with lines in the middle row. Insertions and deletions are shown with the “@” character. LCS length is eleven characters in this example.	14
3.1 Estimated character and word OCR accuracies for books in English, French, German and Spanish from the Internet Archives. Punctuations are ignored.	42
4.1 Example records for <i>Othello</i> from the Internet Archive’s catalog. The first and the third records include the original text of Othello with significant amount of additional text in the form of commentary and footnotes. The second book consists of a number of works written by Shakespeare also including the full text of Othello. The last record is an entirely different work written by a different author. The first three books are partial duplicates of each other.	47
4.2 DUPNIQ-cs and DUPNIQ-its scores for three pairs of books. X and Y refer to the sequences of unique words for Book X and Book Y respectively. GT is the ground truth.	52
4.3 Precision (P), recall (R) and F-measure (F) scores for the Training, 1K, 3K and Partial Duplicates sets.	58
4.4 Precision (P), recall (R) and F-measure (F) scores for the 100K dataset.	58
4.5 Precision (P), recall (R) and F-measure (F) scores of the proposed approach (DUPNIQ-its and DUPNIQ-cs) for the Training, 1K, 3K and Partial Duplicates sets. M and T correspond to the maximum term frequency and duplicate score thresholds respectively.	60
5.1 Detailed statistics for three pairs of books compared using the proposed translation detection framework.	74

5.2	Dictionary statistics after ignoring phrasal translations.	75
5.3	Translation retrieval and ranking all-pairs experimental results for the EUROPARL dataset using the proposed approach with dictionary transformation.	81
5.4	Classification experiments for the EUROPARL dataset using the proposed approach with dictionary transformation.	81
5.5	Precision (P), recall (R) and F-measure (F) scores of TRANSNIQ-its and TRANSNIQ-cs for the EUROPARL dataset without the dictionary transformation . M and T correspond to the maximum term frequency and translational similarity score thresholds, respectively.	84
5.6	Ranking all-pairs (Average Precision - AP) and translation retrieval (Mean Average Precision - MAP) results for TRANSNIQ-its and TRANSNIQ-cs on the EUROPARL dataset without the dictionary transformation . M corresponds to the maximum term frequency threshold.	85
5.7	Translation retrieval and ranking all-pairs experimental results for the scanned book datasets.	87
5.8	Binary classification results on English-German datasets using the sequence of unique words representation with dictionary transformation. “T”, “P”, “R” are threshold, precision, recall and F-measure scores respectively.	87
5.9	Precision (P), recall (R) and F-measure (F) scores of the proposed approach (TRANSNIQ-its and TRANSNIQ-cs) without the dictionary transformation for the Training and 2K sets. M and T correspond to the maximum term frequency and translational similarity score thresholds respectively.	89
5.10	Ranking all-pairs (Average Precision - AP) and translation retrieval (Mean Average Precision - MAP) results for TRANSNIQ-its and TRANSNIQ-cs on the scanned book datasets without the dictionary transformation . M corresponds to the maximum term frequency threshold.	89

5.11	Detailed statistics for the three pairs of books examined in Figure 5.3. $ X $ and $ Y $ corresponds to the number of unique words in books X and Y respectively. $ X \cap Y $ corresponds to the number of common words between $ X $ and $ Y $ without any translation. $ X_T \cap Y $ refers to the number of common words after translating the words in $ X $ to the language of book $ Y $. $ LCS $ is the length of the longest common subsequence between the word sequence representations. TRANSNIQ-its and TRANSNIQ-cs scores are also shown where the corresponding thresholds are 0.49 and 0.023, respectively.	92
6.1	Precision, recall and F-measure scores for the sentence alignment task on the EUROPARL dataset.	110
6.2	Average P@1 scores of RTA and the CLIR baseline for the passage retrieval task on the Gutenberg dataset. PL refers to the average word count of the passages in the English version. TW means term weighting, nTW means no term weights are used, SW means stopwords are used while nSW means without stop words.	111
6.3	The effect of dictionary size for the book Othello (English-German). RTA compared with the CLIR baseline using P@1 score.	111
6.4	Average precision (P), recall (R) and F-measure (F) scores of RTA and the SLA baseline for the sentence alignment task on the Gutenberg dataset.	112
6.5	P@1 scores for varying amounts of document noise and the content overlap.	114
6.6	Average P@1 scores for the sentence retrieval task on the scanned book collection.	115
7.1	MAP scores comparing the document image search and OCR text search for the English Book	129
7.2	MAP scores of the proposed image search framework for the Telugu books	129
7.3	Comparison of the methods for the query result evaluation. It is seen that morphological expansion underestimates the MAP score.	146
7.4	Frequency distribution of the words in the Latin, Telugu and Ottoman books after ignoring punctuation.	149

7.5	The training and test sets used for evaluation purposes (Latin, Telugu and Ottoman scripts).	155
7.6	MAP scores results for resolving arbitrary query in the test book titled “Wuthering Heights”.	157
7.7	Experimental results for the Telugu dataset.	162
7.8	Experimental results for the Ottoman dataset for three different patch size selection approaches.	170

LIST OF FIGURES

Figure	Page
1.1 Table of contents of the book “The Critique of Pure Reason” by Emmanuel Kant printed in a) English and b) German. The sections follow the same global order across translations.	3
1.2 Two instances of the word “Holmes” are shown on a scanned page image from the book “Sherlock Holmes” by A. Conan Doyle. The visual features follow exactly the same sequential order in both instances of the word.	4
1.3 a) a page image from the book “Tremendous Toronto” in the Internet Archive’s database b) the corresponding OCR text output. Underlined portion of the text is not recognized correctly by the OCR engine.	5
3.1 Word frequency as a function of word rank as defined by Zipf’s law.....	28
3.2 The scatter plot for the ratio (density) of unique words in the text as a function of total word count in the OCR output.	29
3.3 The Recursive Text Alignment Scheme (RETAS) depicted for two short texts. “...” stands for the skipped content for illustration purposes. Double headed arrows indicate matching words. “@” is a “null” indicator used for designating character insertion and deletions.	32
3.4 a) The accuracy and b) the speed of the character alignment versus the document noise for different values of maximum candidate term frequency threshold.	35
3.5 The total number of anchor words as a function of the maximum candidate term frequency threshold for different amounts of synthetic character level document noise.	36
3.6 Estimated and ground truth OCR accuracies for characters, words and stopwords versus document noise.	41

4.1	An example pair of books which are partial duplicates of each other. One book subsumes the other.	44
4.2	Two sample pages from two different versions of Shakespeare’s Othello downloaded from the Internet Archive’s website. The duplicate text is shown with red rectangles.	45
4.3	An example pair of books which are partial duplicates of each other. The version on the right contains an extra essay in the middle section.	46
4.4	Illustration of partial duplicate detection for the two versions of Robert Burns’ poem. Unique words are underlined and listed according to their original order in the text. The two sequences of unique words are finally compared using LCS alignment.	50
4.5	DUPNIQ-its and DUPNIQ-cs versus percentage of overlap and character level document noise. Each line in the plot represents results for the given amount of noise. Dashed horizontal lines corresponds to the duplicate detection score thresholds. Note that the word error rate is around 70% for 20% character level noise.	62
4.6	The ratio of preserved text elements across the synthetic texts as a function of document noise.	64
4.7	The total number of book pairs compared per second by DUPNIQ as a function of maximum term frequency threshold.	66
4.8	Examples for partial duplicates of books. Green bars show matching portions of text with 50% or above word overlap. (a) The book Points of Humour (top) contains a selection of verses from the Complete Works of Robert Burns (bottom). (b) Tales from Shakespeare (top) contains selections from Shakespeare’s plays including the Tempest (bottom).	68
4.9	Mapping duplicated portions of two books which contain a number of stories in common. The story “The German Chicago” is not colored green in the visualization output since it does not follow the same order in both books.	69
5.1	Illustration of the proposed translation detection framework.	71

5.2	a) Covers of the English translation and the German original of John Wiclif’s biography. b) the approximate overlap between the two translations (upper bar English, lower bar German).	72
5.3	The effect of OCR errors on the translation scores are investigated for three different scenarios. TRANSNIQ-its (left) and TRANSNIQ-cs (right) scores are shown as a function of word level synthetic document noise. Both measures are able to classify the book pairs correctly for the given thresholds even for high rates of character level document noise.	91
6.1	The red box on the left is a paragraph from the original version of the book “The world as will and idea” by A. Schopenhauer. The corresponding passage (red box) in the English translation on the right includes additional text (green box) which is not present in the original book.	98
6.2	The red box on the left denotes a query passage from the English translation of Kant’s Critique of Pure Reason. On the right the corresponding passage (red box) is automatically derived in the German original of the book. The passages can be derived because of the matching words (underlined in red) between the two books derived by alignment. Note that the German book contains a lot of extraneous material - comments - which are not found in the English version.	99
6.3	The recursive translation alignment scheme (RTA) depicted for two short texts in English and French. “...” stands for skipped content for illustration purposes. Double headed arrows indicate matching words.	103
6.4	Sample corresponding passages for the English and German versions of the History of Dogma. Some OCR errors. RTA finds this passage while both other techniques fail.	117
6.5	The figure shows the approximate overlap between a German original (upper bar) and the English translation (lower bar) of Goethe’s Faust using the proposed visualization scheme for translations.	118

7.1	Visual features are shown for an example word image. In a) small dots correspond to the local interest points. Local patches extracted from interest points are quantized into visual terms (v_1, v_2, \dots, v_9) which are represented with large big circles at the bottom. b) and c) shows an example image patch from the word image and the corresponding SIFT features respectively. There are typically around 100 visual terms per word image.	123
7.2	Corner points and corresponding visterm IDs for a letter bigram image. Visterms having the same ID are shown in circles. Notice that some visterms are spatially very close and therefore image features extracted from these regions are almost identical.	126
7.3	Matching visterms between two instances of a Telugu word from Telugu-1718 are shown. There are a large number of matching visterms following the same order even though the top image is underlined.	127
7.4	Example Telugu word images which are correctly retrieved using our methodology.	130
7.5	A ranked list for the long query word shown at the top. There are 109 relevant word images in the book. AP score for this query is 1.0.	131
7.6	A ranked list for the short query word shown at the top. Incorrect matches are shown along with their rank and the matching characters of the image are underlined. AP score for this query is 0.85.	131
7.7	The configuration of our MRF model for searching text in document images.	135
7.8	The learning models illustrated for learning the probability distributions of visterms for the letter bigram class q_j from three training word images C_1, C_2 and C_3 . The visterm distribution of visterms for each training sample are shown in a), b) and c). The horizontal and vertical axes represent the visterm IDs v_i and the corresponding probability respectively. Estimated visterm distributions for the letter bigram class q_j are shown using d) the Union and e) the Intersection learning models.	141
7.9	Example text lines from the Ottoman dataset. The Ottoman script is quite similar to the Arabic script with some additional letters and missing diacritics.	152

7.10	The number of distinct letter bigrams as a function of the length of the text.....	154
7.11	Distribution of query words as a function of their length is given in a). MAP score distribution as a function of the query word length is given for b) the proposed approach (Intersection model), c) OCR text search baseline and d) the two approaches combined.	159
7.12	a), b) and c) shows example word images which are correctly retrieved by the proposed dependence framework but missed by the OCR text search baseline because of OCR errors. d), e) and f) shows word images which are correctly recognized by the OCR engine and retrieved by the text search baseline. These word image images were missed by the proposed approach. The combined approach (image + OCR text search) correctly retrieved all these word images.....	160
7.13	Intersection model's MAP score distribution on the TELUGU-1718 test set as a function of the query word length.	163
7.14	Telugu word images successfully retrieved by the proposed model.	165
7.15	An unsuccessful short query example.	167
7.16	MAP scores as a function of the patch scale factor for different configurations of the proposed model computed for the Ottoman training set.....	169
7.17	MAP score distribution of the best configuration on the OTTO-2 test set as a function of the query word length.	172
7.18	Two example queries on the OTTO-2 test set.	173

CHAPTER 1

INTRODUCTION

Public libraries and private collections host millions of books all around the globe. These book collections constitute the written part of human cultural heritage. Most of the books are currently in physical form. Several organizations, such as the Internet Archive [1], are digitizing physical copies for preservation purposes. As of today, several million books have been scanned and they are available in digital image format. Now the question is how to extract textual information automatically from individual page images and use them to infer more information about the books in the collection.

Several abstraction levels can be defined for a collection of scanned books. At the highest level, one can view the entire collection as a whole and discover interesting contextual relationships or linkages between the books. These links might take different forms. Books might be related by being on the same topic, written by the same author or having overlapping content. A more traditional approach is to consider each scanned book separately and perform information search and mining at the book level. Examples of this include searching for text and phrases, finding named entities, investigating the social network of the people in the book etc. Here we also show that one can view each book as a whole composed of chapters, sections, paragraphs, sentences, words or even characters positioned in a particular sequential order sharing the same global context. The information inherent in the entire context of the book is referred to as *global information* and it is demonstrated by addressing a number of research questions defined for scanned book collections.

Different types of global information might be available in the scanned page images of books. The *global sequence information* is inherent in the textual content of the books and here we demonstrate that it is essential for discovering content overlap and similarity across books. The observation is that each book has a specific flow of ideas and events which distinguishes it from other books. If this global order is changed, then the flow of events and consequently the story changes completely. For example, in the story of Adam and Eve, the snake tempts Eve to eat the apple, then Eve eats it, and finally Adam and Eve are expelled from the garden. If this order is changed, then the flow of events and consequently the story changes completely. This argument is true across document translations as well. Although the local order of words in a sentence might not be preserved after translation, sentences, paragraphs, sections and chapters are likely to follow the same global order. Otherwise the two texts are not considered to be translations of each other. Figure 1.1 shows the table of contents for an example translation pair of books where it is seen that the sections are clearly preserved after translation. The global sequence information is always inherent in textual documents and it motivates improved representation and matching of texts in scanned book collections.

Another example of global information is the *global font* feature. Books are mostly printed in a single font type and here we demonstrate that this global information can be used for improving text search in page images. As an example, Figure 1.2 shows two instances of the word Holmes on the same scanned page image. One of the instances (at the top) is misrecognized by the OCR engine because of the document image noise localized on the character “s”. However, if one compares these two word images as a whole, it turns out that these images are visually quite similar because of font similarity. Word image search mechanisms relying on the global font feature can therefore match these two words reliably even though OCR engines can not recognize one of them correctly. It should be noted that OCR engines recognize

CONTENTS	
INTRODUCTION	
	PAGE
I. Of the Difference between Pure and Empirical Knowledge..	1
II. The Human Intellect, even in an unphilosophical state, is in possession of certain cognitions <i>à priori</i>	2
III. Philosophy stands in need of a Science which shall determine the possibility, principles, and extent of Human Knowledge <i>à priori</i>	4
IV. Of the Difference between Analytical and Synthetical Judgments	7
V. In all Theoretical Sciences of Reason, Synthetical Judgments <i>à priori</i> are contained as Principles.....	9
VI. The Universal Problem of Pure Reason.....	12
VII. Idea and Division of a Particular Science, under the Name of a Critique of Pure Reason.....	15

(a)

Einleitung.		1 - 30	35 - 64
I. Von dem Unterschiede der reinen und empirischen Erkenntniss	1		35
II. Wir sind im Besitze gewisser Erkenntnisse <i>a priori</i> und selbst der gemeine Verstand ist niemals ohne solche	3		40
III. Die Philosophie bedarf einer Wissenschaft, welche die Möglichkeit, die Principien und den Umfang aller Erkenntnisse <i>a priori</i> bestimme	6		42
IV. Von dem Unterschiede analytischer und synthetischer Urteile	10		45
V. In allen theoretischen Wissenschaften der Vernunft sind synthetische Urteile <i>a priori</i> als Principien enthalten	14		49
VI. Allgemeine Aufgabe der reinen Vernunft	19		53
VII. Idee und Einteilung einer besonderen Wissenschaft unter dem Namen einer Kritik der reinen Vernunft	24		56

(b)

Figure 1.1. Table of contents of the book “The Critique of Pure Reason” by Emmanuel Kant printed in a) English and b) German. The sections follow the same global order across translations.

the text at the word or character level and they do not exploit the global font information. Another observation is that individual characters and the visual features extracted from the corresponding word image follow a particular order in each word. Figure 1.2 shows the visual features extracted from the two word images and they are represented with circles. A line is drawn across the two word images for the matching visual features. Clearly the matching visual features follow the same order in both word images because of the inherent letter sequence information. The sequence of visual terms and the global font features are demonstrated to improve text search in noisy document images.

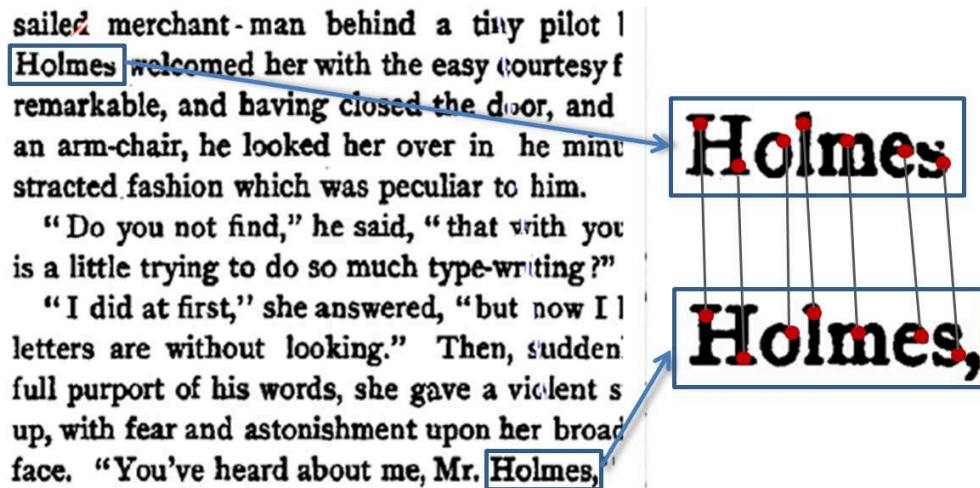
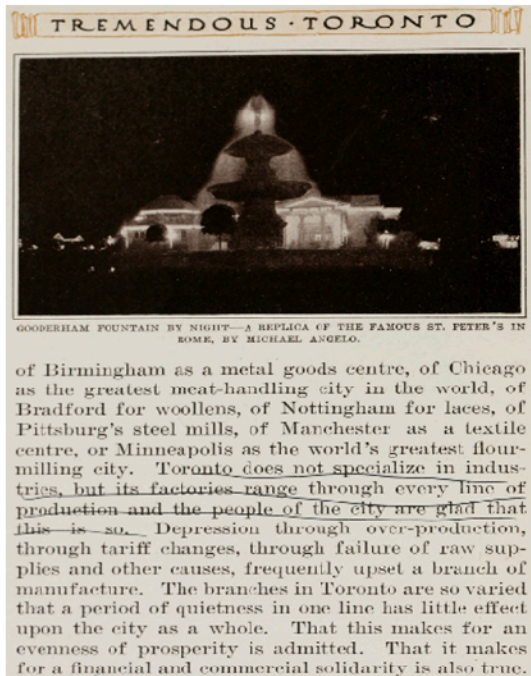


Figure 1.2. Two instances of the word “Holmes” are shown on a scanned page image from the book “Sherlock Holmes” by A. Conan Doyle. The visual features follow exactly the same sequential order in both instances of the word.

In the first part of this dissertation, the global text alignment approach is proposed to discover the contextual similarity between a given pair of books. Traditional text based approaches rely only on the existence of common words, their frequencies and/or their local ordering (n-grams of words) to determine the similarity between two input texts [115, 22, 44, 103, 40, 14, 18, 17, 64, 99, 25, 20]. On the other hand, the global text alignment approach accounts not only for the existence of words but also their global sequence information in the entire text. Given the OCR text outputs of books, the global sequence alignment is achieved by looking for the longest subsequence of words which are common in both texts. If the two texts are similar, then a large number of words are expected to be in the Longest Common Subsequence (LCS). The problem with this approach is that conventional LCS algorithms are slow and not robust for book length documents especially with OCR errors, additional or missing content. The standard dynamic programming implementation of LCS takes 23 min on a single core to align two books of size 100K words each. In the literature, conventional text alignment approaches are therefore assumed to be computationally prohibitive for long texts such as books [34]. Another challenge is



(a)

of Birmingham as a metal goods centre, of Chicago as the greatest meat-handling city in the world, of Bradford for woollens, of Nottingham for laces, of Pittsburg's steel mills, of Manchester as a textile centre, or Minneapolis as the world's greatest flour-milling city. Toron to does no LÄÄÄt>Ätci arize in industries, but its factories ra'oro ti-ivnivrVi every-lme of ^ji'lm4m> n"A +lä.,ç pÄÄ^plc of _thecity are glacttllia t llux äe" i*-äe" sll__ Depression through over-production, through tariff changes, through failure of raw supplies and other causes, frequently upset a branch of manufacture. The branches in Toronto are so varied that a period of quietness in one line has little effect upon the city as a whole. That this makes for an evenness <of prosperity is admitted. That it makes for a financial and commercial solidarity is also true.

(b)

Figure 1.3. a) a page image from the book “Tremendous Toronto” in the Internet Archive’s database b) the corresponding OCR text output. Underlined portion of the text is not recognized correctly by the OCR engine.

that the accuracy of character recognition also depends on various factors such as the script, font type, scanning quality etc. Figure 1.3 shows an example OCR output for a page image with recognition errors. The OCR text output for scanned books can be characterized as long noisy strings containing hundreds of thousands of words without any structure. These features make scanned books distinct from other types of documents and database records widely studied in the literature. It is shown that well known text representation and matching schemes (fingerprinting [14, 18, 97], shingling [43, 94], I-Match [22] and other word frequency based approaches [98]) are sensitive to the document noise and therefore they are not as effective in this particular problem domain. There is a high need for efficient and robust alignment and/or matching techniques for long noisy texts such as OCR text output of scanned book collections.

As a solution, we propose a general framework which can be used to efficiently align and match the textual content of the books at various levels and even across languages. More specifically, the framework uses the sequence of words which appear only once in the entire book (referred to as “the sequence of unique words”) to represent the text. Only a small percentage of the words are typically unique in the text. Along with the global sequence information, they are highly descriptive of the content. The general framework uses the sequence of unique words to align input texts and produce an initial alignment. If the two books have any content similarity, then a large number of unique words following the same order are expected to be found in both texts. The sequence of unique words alignment can be performed quite fast algorithmically (12K book pair comparisons per second on a single core ¹). The experiments show that it is not necessary to align the entire texts of each pair of books to find whether they are partial duplicates. Instead, it is shown that the alignment of unique words is sufficient for partial duplicate detection purposes [119]. It is also demonstrated that the proposed approach outperforms well-known approaches such as I-MATCH [22], shingling (n-grams of words) [14, 64], DCT fingerprinting [97] and other bag-of-words text representation schemes which do not exploit the global sequence information.

The sequence of unique words alignment produces a coarse level matching between the two texts. It turns out that one can also generate the complete global alignment between two texts by splitting the input texts at positions where the unique words match. Resulting text segments are then aligned recursively in the same manner until they get short enough for dynamic programming. The alignment outputs of individual text segments are finally concatenated to produce the complete global alignment. It is shown that this Recursive Text Alignment Scheme (RETAS) is highly accurate

¹All the timing experiments are performed on a desktop computer with an Intel i5-2500 micro-processor.

and it generates the complete alignment under a fraction of a second for a book pair of length 100K each [120]. As an application, it is also demonstrated that one can use RETAS to map the duplicated portions of books which are partial duplicates of each other. Finally RETAS is used for the automatic evaluation of OCR accuracy of scanned books.

The alignment of unique words approach is also generalized for the cross-lingual case to detect translations of books. In this case, the sequence of unique words extracted from the source book is transformed to the language of the target book using a look-up dictionary. Individual words in the word sequence are replaced with their possible translations in place regardless of their translational probabilities. If there is no translation for the word, then it is kept in the sequence without any translation. There is no need to translate all the words at this stage. This approach is shown to be effective, even if some of the words are not translatable due to modest size dictionaries. Once the two representations are in the same language, one can apply the procedure discussed previously to find duplicates in the same language. It should be noted that the proposed approach does not require word translation probabilities and/or machine translation systems to find translation pairs of books unlike many other existing approaches in the literature [108, 30]. It is shown that this approach is quite effective and fast to find translation pair of books in large scanned book collections [122].

The Recursive Text Alignment Scheme is generalized for aligning long noisy texts across languages as well. Given a pair of documents written in two different languages, the task is to find the corresponding pieces of text in the form of translation despite the presence of document noise, additional and/or missing text, and, the absence of any structural information. The input documents might not necessarily be exact translations of each other (i.e., there is no 1:1 correspondence between the texts). There is also no structural information or metadata to infer the position of each

correspondence. One possible solution is to automatically translate the text of the source document to the language of the target document using a machine translation system [19]. Once the two texts are in the same language, mono-lingual text alignment and search methods become applicable. This approach requires efficient and robust machine translation systems to be available for each language pair. Another approach is to segment the input texts into sentences and find correspondences using sentence alignment algorithms. Sentence alignment techniques, however, require reliable sentence boundaries which may not be available as in the case of scanned book collections [72, 16, 30]. Besides, sentence aligners typically assume that there is a 1:1 mapping between the source and translation without any extra or missing text. Otherwise they can get very inefficient, e.g., [72]. Yet another approach is to segment the text into sentences or passages and use cross-lingual retrieval frameworks to locate translations. It should be noted all the alternative approaches discussed above directly use text structure in the form of passage and/or sentence boundaries which may not be always available. On the other hand, the proposed Recursive Translation Alignment (RTA) framework regards each book as a sequence of words without any structure. Therefore RTA does not need the text to be structured in any way. RTA first applies the dictionary transformation approach to the entire sequence of words of the source book. The source and target texts are finally aligned at the word level using RETAS. To the best of our knowledge, none of the approaches in the literature are designed for aligning long noisy texts across languages where there might be large portions of additional and/or missing text. It is demonstrated that RTA outperforms the cross-lingual and sentence alignment baselines with a very large margin for aligning long noisy texts such as OCR text outputs of scanned books.

Finally, the effective use of global font and letter sequence information is demonstrated for searching text in noisy document images. Given a query word (either in the image or text form), the task is to retrieve all the instances of the query word in

the document images. The naive approach is to automatically recognize the text and perform text search in the OCR output of the document images. However, the effectiveness of automatic text recognition systems falls drastically for degraded document images. There are also scripts for which there is no commercial OCR engine available such as Telugu and Ottoman [118]. In those cases image search mechanisms become a viable option to facilitate or improve text search in document images. Query-By-Example (also known as, “word spotting” [65]) is one common approach to searching text in document images. The basic assumption is that the word images are printed in the same or a similar font and therefore different instances of the same word can be reliably matched using visual features. In this particular case, the query word image provided by the user is matched to other word images in the entire document, book or collection. The major challenges are non-uniform document noise, differences in font and computational overload due to high dimensional image features. Exhaustive approaches using high dimensional image features are therefore not scalable for searching page images of scanned books. As a solution, a novel word spotting framework is first introduced to search text in scanned books. Local image features are first extracted from the word images and then quantized into integer values. Each word image is represented with a sequence of visual terms sorted according to their position on the horizontal axis. The observation is that the visual features extracted from different instances of the same word follow the same order on the horizontal axis. The word image similarity is simply computed by aligning the sequence of visual terms using LCS. It is shown that this approach is quite effective and fast. With visual term indexing and an efficient filtering mechanism, the proposed approach resolves a query under 10 milliseconds for an entire book [121].

The general problem with the word spotting approaches is that the user is required to find a query word example in the document images. The QBE approach is therefore not practical for searching terms which appear rarely in the document

images, such as names and places. As a remedy to this problem, we finally generalize the word spotting concept to searching arbitrary text queries in document images. A cross-media retrieval approach using a dependence model is devised for this purpose. Effectively the proposed approach trains visual features relevant to each letter bigram class present in the query term. The trained visual term distributions are used for locating the position of each letter bigram in the word image. The proposed approach not only accounts for the existence but also letter bigram sequence information to resolve arbitrary text queries. The effectiveness of this approach is first shown for improving OCR text search on Latin books with noisy OCR output. It is also demonstrated that the proposed approach effectively searches arbitrary query words in document images printed in Telugu and Ottoman scripts for which there is no OCR engine available. To the best of our knowledge, this is the only approach in the literature which allows arbitrary text queries to search document images without recognizing individual characters.

The major and minor contributions of this work may be summarized as follows:

1. a novel document representation called “the sequence of unique words” and a matching scheme for long noisy texts

Using this representation, the following frameworks are proposed:

2. Recursive Text Alignment Scheme (RETAS) for efficient alignment of long noisy texts
 - (a) an automatic OCR evaluation system using the proposed alignment scheme
 - (b) a tool for detecting/mapping the overlapping content of two books which are partial duplicates of each other
3. an efficient partial duplicate detection framework (DUPNIQ) for scanned book collections

The above approaches have also been extended for detecting duplicate content across languages:

4. a state-of-the-art translation detection framework (TRANSNIQ) for scanned book collections
5. Recursive Translation Alignment (RTA) framework for efficiently aligning document translations directly at the word level
6. Global font feature have been used for improving text search in noisy document images:
 - (a) a real-time word spotting framework for effectively searching text in noisy scanned page images of books
 - (b) a state-of-the-art dependence model approach to resolve arbitrary text queries in document images solely using visual features

The rest of the thesis is organized as follows: The next chapter (Section 2) contains a literature overview for the problem domains addressed in this work. In Section 3, the alignment of unique words and the Recursive Text Alignment Scheme is elaborated. The “sequence of unique words” text representation and its use for partial duplicate detection is introduced in Section 4. This document representation is later extended for finding translations of books as well in Chapter 5. Chapter 6 generalizes the Recursive Text Alignment scheme for aligning text across languages. Finally the image search mechanisms are investigated for searching text in document images in Section 7.

CHAPTER 2

LITERATURE OVERVIEW

A number of research problems relating to scanned book collections are reviewed in this chapter. The first area of interest is how to use the OCR text output to infer contextual similarity between books. If the OCR text output has a reasonable recognition accuracy, then one can align the words and characters of books to determine overlapping content and duplication. The algorithms for aligning long noisy texts are first discussed in Section 2.1. Conceptually, partial duplicates of books can be discovered by aligning each book against all others in the collection. However, the sequence alignment approaches are not scalable enough to perform partial duplicate detection for large scanned book collections. Even in a small collection consisting of only a thousand books, there are about half a million book pairs that need to be compared. An alternative approach is to extract textual features for representation purposes and use them for efficient text comparison, as discussed in Section 2.2. The text representation and alignment approaches can be generalized for comparing texts across languages as well. Section 2.3 discusses approaches to find translations of books in scanned book collections. Aligning long noisy texts across languages is elaborated in Section 2.4.

The secondary area of interest is how to use the scanned page images to improve the understanding of textual content in the scanned book collections when OCR is not effective. The OCR output might be quite noisy due to recognition errors and it might be insufficient for effective contextual analysis. There might also be no OCR text output available since the OCR engine can not recognize the font or the script of

the book. In those cases, visual features can be used for searching text in document images as discussed further in Section 2.5.

2.1 Alignment of long noisy texts

Sequence alignment approaches are widely used for discovering contextual similarity between texts. In this context, each input text is regarded as a single sequence of words or characters without any structural information such as page, paragraph or sentence boundaries. Sequence alignment approaches may be categorized into two classes: global and local alignment methods. The global alignment techniques try to find the alignment which optimizes the global objective function calculated over the entire input sequences. For example, finding the Edit-Distance (also referred to as Levenshtein distance) is a global alignment problem where the objective is to minimize the total cost of insertions, deletion and replacements to transform one sequence into the another. The Longest Common Subsequence and Needleman-Wunsch are other examples of global alignment approaches. Global alignment methods are most suitable for the cases where the aligning sequences have significant content overlap. On the other hand, local alignment techniques, such as Smith-Waterman, aim to localize one query sequence inside a much longer sequence and they are widely used for biological sequence analysis [32]. The global alignment approaches are preferred in the case of aligning long textual documents such as books [35, 15, 56].

In this particular work, text alignment is framed as a Longest Common Subsequence (LCS) problem. LCS is actually a special case of the Edit-Distance problem where substitutions are not allowed. In other words, the objective function aims to maximize the total number of matches in the alignment. Disallowing substitutions makes the alignment problem computationally much simpler since only the number of exactly matching words need to be accounted for. In order to speed-up the alignment, one can therefore preprocess the input sequences and remove the words which

Table 2.1. Longest Common Subsequences illustrated for two words “NEIHOORHOOD” and “NEIGHBORHOOD”. Matching characters are indicated with lines in the middle row. Insertions and deletions are shown with the “@” character. LCS length is eleven characters in this example.

N	E	I	@	H	B	O	U	R	H	O	O	D
N	E	I	G	H	B	O	@	R	H	O	O	D

do not appear in both sequences. This can be done in linear time using hashing techniques. In this way, the alignment can be carried out rapidly especially if the two input sequences do not match. Table 2.1 shows an example where the two character sequences are aligned using LCS.

There are a number of algorithms for solving the Longest Common Subsequence problem in the literature [31]. Given two arbitrary sequences of length m and n ($m \geq n$), the standard dynamic programming implementation of LCS has $O(mn)$ time and space complexity. Hirschberg [45] showed that an optimal alignment can actually be computed in $O(mn)$ time and only $O(n)$ space using binary-recursion. There is also a fast LCS algorithm available with $O(n \log \log n)$ amortized time complexity for the special case where the terms appears at most once in either input sequence [49]. This is achieved by converting the LCS problem in to a Longest Increasing Subsequence (LIS) problem and solving using a data structure called “Emde Boas” [109]. More recently Crochemore et al. have reduced the bound for LIS to $O(n \log \log k)$ where k is the length of the LIS (or in our case, LCS) [26]. Ukkonen’s suffix tree based string alignment approach can also be modified to produce the LCS [107]. This approach has previously been applied to the OCR evaluation and OCR error correction problems [89, 15, 56]. The time complexity of Ukkonen’s algorithm is $O(nd)$ where d is the edit distance between the input strings. Although Ukkonen’s algorithm is efficient for short sequences, it can be expensive especially for long sequences with potentially large gaps. In the case of books, the edit distance value d is typically large due to OCR errors, edition differences, missing or additional content.

One well-known Unix application “diff” [48] uses a Longest Common Subsequence algorithm to find out the lines which differ between the two text files. This is achieved by hashing each line and calculating LCS over the generated sequences of numeric values. Clearly this application is not suitable for our purposes because lines are typically not preserved across different versions of books. In addition, there might be OCR errors, spelling differences and formatting changes which causes the lines to obtain different hashcodes for alignment. The proposed text alignment scheme (Section 3) aligns the texts at the word level instead and it has a principled way to reduce the computational complexity for aligning texts written in some natural language.

In the case of text alignment, the cost of alignment can be reduced if the corresponding portions of the two texts are known a priori. Specifically, this is achieved by splitting the two input texts into smaller segments using the correspondence information. The total cost of aligning those text segment pairs is significantly lower compared to the case where the two input texts are aligned entirely without any splitting. The overall cost reduction is due to the quadratic time complexity of LCS for aligning texts. However, there is no such correspondence information available in the case of scanned book collections. The page, paragraph and even sentence boundaries might not be preserved across different editions, prints and versions of the same book. The OCR text output of the scanned book is characterized as long noisy texts without any particular structure. As a solution, Feng and Manmatha’s [35] proposed the use of “unique” words as anchor points for splitting the text into smaller text segments. Next, each subproblem is solved separately using a HMM alignment model. Inspired by this approach, our framework also generates a number of problems smaller in size but instead uses a recursive approach followed by an edit distance based alignment model. It is shown that the proposed approach is robust

and faster than the aforementioned techniques for aligning long texts written in some natural language.

Sequence alignment approaches have been widely used in the area of bioinformatics as well. Biological sequences include the amino-acid sequences of different proteins or the nucleotides of DNA sequences [32]. These sequences might be quite long including billions of characters as in the case of DNA. The alphabet for biological sequences is much more restricted as compared to texts. For example, the DNA character set includes only four elements (A, T, G and C). The correspondences between different biological sequences are not exact due to several factors such as biological mutations and other variations in the expression of the genome. The inexact nature of the alignment is typically incorporated into the alignment models accounting for each type of modification/change across input sequences. In the case of text, white spaces are used to designate logical and/or textual elements such as words, sentences and paragraphs. However, biological sequences do not have any type of explicit breaks and/or boundaries. Due to these fundamental differences, most of the heuristics applied in bioinformatics are not directly applicable to align texts.

In bioinformatics, a number of approaches have been proposed to locate correspondences (also referred to as “seeds” or “anchors”) between the sequences. Those seeds are then used for guiding the latter stages of the local or global alignment. A sliding window of size n is widely used to extract local features (n -grams of nucleotides or amino acids) from the input sequences. Typically the window size n is set to a small number, such as eight. The n -grams are then indexed for efficiently finding the corresponding locations across the input sequences. Variations of these anchor or seed based approaches include ACANA [47], BLAST [5], BLASTZ [95], FASTA [80], PatternHunter [62] and YASS [76]. Delcher et al. [29] locate unique matches between the input sequences and produce an initial alignment. Those unique matches are aligned at the top level using dynamic programming and used for guiding the

global alignment of genomes. This process is referred to as the Maximal Unique Match (MUM) decomposition. Unlike biological sequences, text is naturally split into words. The proposed Recursive Text Alignment Scheme exploits the fact that some kinds of words lead to unique matches and those words are used for aligning long noisy texts efficiently.

2.2 Duplicate detection

Most of the work in near duplicate detection involves using either fingerprinting algorithms or using relative frequency techniques (based on using words with similar frequencies) [14]. The fingerprint techniques [14, 18] assume that each document can be broken up into “distinctive” chunks or shingles and two documents which have a large number of chunks in common are likely to be similar to each other compared to documents which only have a small number in common.

Chunks are created using n -grams of words or characters. Note that this is more likely to make the chunks unique since an n -gram is less common than the original word. The chunks are later indexed and used to match duplicate documents. Since a document can contain a large number of chunks, most algorithms subsample this set of chunks and differences in sampling strategy distinguish the various approaches [46]. Several sampling techniques have been tried such as full sampling, random sampling and picking every k^{th} chunk [43]. “0 mod p ” is one of the most widely used sampling approach which hashes each chunk to a discrete value and chooses the chunks whose hash value mod p is equal to zero for matching documents. Here p refers to the sampling factor which is empirically determined. This method is later used as one of the baselines in the experimental section. Another approach windows the chunks [94] by picking the chunk with the lowest hash-key as a window is moved over the document. Bernstein and Zobel [14] use the fact that every sub-chunk of a duplicated chunk must be non-unique to reduce the number of chunks in multiple

passes. Other chunking algorithms include those by [17, 64, 99]. The sub-sampling required in practice means that many of these algorithms do not work as well when the documents are only partial duplicates or noisy OCR output is considered [14]. I-Match [22] extracts words with certain statistical characteristics and hashes their aggregation. These hash values are compared to determine duplication. Talent [25] similarly finds certain kinds of content words and hashes them. Note that both I-Match and Talent use collection statistics rather than individual document statistics as done here. Charikar [20] applied a random projection based method - essentially locality sensitive hashing on the terms of a document - to find near duplicates and Henzinger [44] applied this to the web domain. Hajishirzi et al. [40] also worked on near duplicates by representing each document as a sparse n-gram vector and learning the weights depending on the similarity measure being optimized.

Shivakumar and Garcia-Molina [98] use relative frequency techniques to detect duplicated digital documents. The assumption is that two documents with similar words and frequencies must be similar or duplicated. Hoad and Zobel [46] also explore a similar approach for finding plagiarized or versioned documents. The relative frequency techniques are claimed to be more accurate than chunking based methods [46]. Local text reuse is a related problem where the duplicates may not be exact. Seo and Croft [97] used a Discrete Cosine Transform (DCT) fingerprinting algorithm for this problem. DCT fingerprinting is explored further in Chapter 4.

In plagiarism detection sequence alignment techniques have been used to find plagiarized passages but it is seen to be impractical for long documents and large collections [24]. For example, eTBLAST [34] was used to search the Medline database by aligning only the abstracts since the whole document alignment was computationally prohibitive. Instead Errami et al.[33] proposed an essentially chunking based approach to searching whole documents in the Medline database.

2.3 Translation detection

The related problem of near duplicate detection in the same language has been well studied especially for web documents as discussed in the previous section. It should be noted that n-grams of words (shingles) are not well preserved across languages since the order of words in a sentence can change across translations. Word frequency distributions are also not preserved across translations especially when there is additional or missing text. These approaches are therefore not directly applicable for finding translations.

There has been work on finding comparable corpora for training machine translation models. Much of this work has focused on finding parallel sentences from small corpora [102] or web pages [74, 88, 102, 124]. Nie et al. [74] and Resnik [88] utilized structural information - HTML markup such as anchors, links, filenames - to find parallel resources. Alignment was specifically rejected as being too expensive. Yang and Li [124] limited the alignment to titles and used a translation dictionary to find parallel texts. There is also a significant amount of work on the extraction of bilingual dictionaries [38]. However, there is much less effort on detecting which documents are translations of each other in large corpora. One of the few papers on identifying translations of documents is by Smith [102]. The paper uses several translation dictionaries and then computes the word overlap. Filtering was done based on document length for efficiency. The method was tested on a small dataset of about 1000 sentence pairs and another dataset of 325 web document pairs. Resnik and Smith [87] combined structural and content features to mine web pages for parallel corpora. Ma and Liberman [63] also used structural features paired with a content filtering scheme to find parallel corpora on the web. Koroutchev and Cebri [52] used the idea that similar texts would have similar graph structures after compression to find translations of portions of texts.

Uszkoreit et al. [108] is one of two papers to find translations of books. They use Google’s large computing resources to translate all the books in the collection to English. This transforms the problem of finding translations to monolingual duplicate detection. Next, they match chunks (n-grams) of words in translated texts to determine translation pairs. One drawback of this approach is that it requires building machine translation systems for all languages and automatic translation of books is computationally expensive. Ideally, one should be able to find translations of books without having to translate them explicitly. The success of their approach is evaluated partially on a small dataset. Uszkoreit et al.’s method is further discussed in the experimental section. Krstovski and Smith [53] use words which are common between translations of books to find translations of books. Each book is represented as a vector in a high dimensional space and the translational similarities between books are defined by several distance measures such as Cosine distance. They use Locality Sensitive Hashing (LSH) to efficiently compute the translational similarity scores. Our technique is compared to their approach on the publicly available datasets and we demonstrate that our approach is more accurate.

There has been work on cross-lingual plagiarism detection. Sequence alignment, word sampling and variants of chunking methods have also been tried for cross-lingual plagiarism detection. Please refer to [82] for a recent survey of those methods. It should be noted that cross-lingual plagiarism and translation detection for scanned book collections are different problem domains. Scanned book collections include long documents with potentially a lot of OCR errors which prohibit the use of conventional approaches.

2.4 Aligning texts across languages

Cross-lingual retrieval and sentence alignment are two tasks related to the cross-lingual text alignment problem. The former approach first segments the input texts

into logical units such as sections, paragraphs or sentences and performs cross-lingual retrieval on them. The latter approach segments texts into sentences and aligns them using sentence alignment algorithms. These approaches are directly relevant to the task of aligning long noisy texts across languages and are discussed further below.

Given a query in one language, cross-lingual retrieval systems try to retrieve (rank) documents in another language. A number of these techniques do word-by-word translation using a dictionary [9, 57] or a machine translation system [10, 77] or by inferring translation probabilities using a bilingual corpus [116]. Since translation coverage can be poor, the retrieval is augmented by query expansion techniques as suggested by Ballesteros and Croft [10] and Levow et al.[57]. Lavrenko et al.[55] uses a relevance modeling approach for cross-lingual information retrieval. In general, it has been found for translation systems that frequent words are less useful than less frequent ones and in fact a number of systems eliminate stop words as in monolingual information retrieval.

Sentence alignment is usually the first step in using these bitexts to infer (weighted) translation dictionaries, translation models, and evaluation data for machine translation systems. Sentence alignment approaches in the machine translation literature primarily focus on extracting a sufficient number of high-precision sentence pairs to train effective machine translation systems. The focus on the precision is therefore reflected in the empirical performance of tools developed for bilingual sentence alignment. Moore [72] described a system that extracts high-precision sentence pairs using only length statistics, estimates a weighted lexicon from those seed pairs, and realigns the bitext using that lexicon. Researchers later on have proposed improvements to both the dynamic programming and model estimation components (for example Braune and Fraser [16]). Deng et al. [30] take a more top-down approach. For a given span of text, they choose a pair of split points, in source and target, based on whether the source text before the point is a good match, under a bag-of-words translation

model, for the target text *either* before or after the target split point; similarly, the source text after the split point must be a good match of the target text after or before the split point. Once the best pair of split points is chosen, the corresponding source and target spans are recursively aligned. In effect, their model implements a greedy, top-down version of an inversion transduction grammar introduced by Wu [114], without the prohibitive $O(n^6)$ time complexity. This recursive approach bears some resemblance to the recursive alignment scheme proposed in this paper. The most important difference is that their exhaustive split point evaluation step is much more expensive than the sequence of unique words alignment method described in the next section.

Word similarities across different languages have also been used for improving the effectiveness of the sentence alignment. The words which are similar both in form and meaning across languages are used for this purpose. Those words are called “cognates” and they are shown to improve the sentence alignment accuracy [100]. For example, the words “Curious” and “Curioso” are cognates in English and Spanish, respectively. Instead of automatically discovering the cognates from the input texts themselves, Chen [21] uses an external lexicon to improve the alignment accuracy. Melamed [69] uses both cognates and an external lexicon to improve sentence alignment. The shortcoming of the approaches which relies on the existence of cognates is that they are highly dependent on the language pair of the dataset [101]. It is therefore desirable to incorporate external language sources such as lexicons into the alignment process. It should be noted that there is no existing work for aligning long noisy texts (such as OCR text outputs of books) across languages. The above approaches are designed for aligning texts at the sentence level.

Unlike sentence alignment approaches for training statistical machine translation systems, our specific task is to map text passages across translations along with their *context*. In some cases, the passages that *do not* align are the primary focus of interest,

since they provide evidence on what has been added, deleted, or changed between different editions. Similar to most bitext alignment methods—and in contrast to most comparable corpus extraction procedures—our approach assumes that passages align monotonically. Unlike many approaches that prune the search space for dynamic programming (such as Moore’s approach [72]), however, our focus is on texts with substantial amounts of extraneous material such as scanned book collections. Sentence aligners such as Moore’s tend to get much slower with extra or missing text.

The proposed approach (Section 6.1.1) makes use of an external lexical source (a look-up dictionary). If the translation lexicon can not translate the source word, then the translation is assumed to be the source word itself. In some cases, the source word appears to be exactly in the same form in the target text. These words are actually a special type of cognate which are spelled exactly the same. Here they are shown to be useful for aligning long noisy texts across languages.

2.5 Searching text in noisy document images

Searching text in document images without explicit character or word recognition is referred to as “word spotting” in the literature [85]. More specifically, a query word image is given and the task is to search for other instances of the same word in other documents using raw image features. Several word spotting approaches have been proposed for printed [92, 93] and handwritten documents [83, 84, 105, 6, 112, 8, 4]. These approaches have been shown to be effective especially for searching text in degraded historical documents. The most important drawback for word spotting systems is that a query image is required for each query word. Therefore a user can not search for arbitrary text unless the word image is available.

Word spotting frameworks mainly differ from each other in three ways: the word image segmentation method, the image features used, and the word image matching/retrieval approach. Projection profiles [86], scale-space approaches [66], Hough-

based methods [59] and gaps metrics [68] have been applied for automatically segmenting text lines and word images. Several image features have been proposed for representing word images including variants of projection profiles, DFT features extracted from projection profiles, ink transitions, gray level variance and local gradient histograms [86, 90]. Rath and Manmatha [83] showed that Dynamic Time Warping (DTW) is particularly effective for matching word images represented by projection profiles. Other methods include aligning the word images and computing a similarity based on pixel wise comparisons using XOR, Sum of Squared Distances (SSD), Euclidean Distance Mapping (EDM) [28] and many other distance metrics [96].

The word spotting paradigm has also been extended to perform holistic word recognition. Given a query word image whose text content is known, one can propagate the text label to other visually similar word images in the document set. Marinai et al. [67] and Pramod et al. [93] use clustering techniques to group similar word images and the word images are labeled based on which clustering they belong to. However, manual labeling of the word image clusters is not practical for large datasets with diverse fonts and writing styles. In addition, these approaches have limitations in the sense that they can not label word images which are not in the vocabulary of recognizable words.

Lu et al. [61] and Bai et al. [7] adopt a word shape coding approach for searching text in document images given a text query. Word shape coding approaches account for the character ascenders and descenders, character holes, and character water reservoirs. Shape codes extracted from the word images are later indexed and used for matching purposes. The problem with word shape coding approaches is that some words may end up having exactly the same shape code although they are different. It is reported that these shape code collisions happen 28% of the time for a dictionary of size 50K words [61]. Shape collisions therefore causes ambiguous search results. In addition, shape codes are sensitive to subtle ink deformations.

Metzler and Croft [70] propose a general Markov Random Field framework (also referred to as “Dependence Models”) for the text retrieval task. Each word in the document and query is regarded as a random variable and the joint probability of the words in the document and the query terms is estimated efficiently. The general MRF framework for retrieval has also been used for image retrieval as well by Feng and Manmatha [36]. As discussed in Section 7.3, we adapt the general MRF framework by Metzler and Croft [70] for searching text in noisy document images. Given a text query, all the word images in the collection are ranked using visual features. The proposed approach models each letter bigram explicitly to avoid collusions and searches arbitrary words in the document collection with a speed of 5 milliseconds per query. The proposed approach also uses local interest points which are known to be robust to subtle ink deformations [121]. To the best of our knowledge, this is the only approach in the literature which allows arbitrary text queries to search document images without recognizing individual characters and shape code collisions.

In this chapter, the literature is reviewed for a number of research problems relating to scanned book collections. The proposed approaches are elaborated in the following chapters.

CHAPTER 3

ALIGNMENT OF LONG NOISY TEXTS

The global alignment paradigm can be used for solving several problems defined over long texts such as scanned books and government documents. Some of the applications include finding duplicates of documents, comparing different versions of texts, OCR error detection and correction [111]. However, conventional sequence alignment approaches are not suitable because of the high computational cost for input sequences of several hundred thousand of words. In this chapter we describe a general text alignment framework which efficiently aligns long texts at various coarseness levels using a novel text representation scheme referred to as “the sequence of unique words”. The proposed alignment approach is shown to effectively align book length documents in a fraction of a second on a single core.

There are several applications of the proposed alignment scheme. This is first demonstrated for evaluating OCR accuracy of real scanned books at the end of this chapter. In Chapter 4, the proposed approach is also used for mapping duplicated portions of texts which are partial duplicates of each other. In Chapter 6, the proposed alignment scheme is generalized for aligning texts across languages as well. The ideas presented here provide motivation for the partial duplicate and translation detection frameworks presented in Chapters 4 and 5. The details of the proposed text alignment scheme are elaborated in the following subsections.

3.1 The Recursive Text Alignment Scheme

The aim is to break the $O(mn)$ alignment problem into a number of smaller problems each of which can be solved efficiently. This is achieved by breaking both input sequences into corresponding regions which follow the same order in both texts. This process generates a number of chunks over the text and aligns them at the top level instead of directly aligning the entire word sequences. The challenge is how to identify the anchor points for breaking the sequences and generating corresponding pieces of texts.

The problem of finding the anchor points is trivial if the corresponding portions of the two texts are known a priori. However, this type of information is not available for scanned books. The OCR text output of the scanned book is a long noisy text without any particular structure. The page, paragraph and even sentence boundaries might not be preserved across different editions, prints and versions of the same book. The proposed approach therefore generates its own anchor points automatically from the input texts by relying on the statistical properties of the texts written in some natural language. The words that appear only once in the entire text (i.e., the unique words) are used as candidate points for cutting the input sequences into smaller pieces. The properties of unique words and their efficient use for the text alignment task are elaborated in the next subsections.

3.1.1 The properties of unique words

According to Zipf's law, in a text corpus, word frequencies are inversely proportional to their corresponding rank in the word frequency table. This holds if the documents are written in some natural language. Figure 3.1 shows the word frequency as a function of rank as defined by the Zipf's law. As seen in the graph, the majority of the words in the vocabulary are expected to be quite rare in the text. Some of the words actually appear only once in the entire text. Here they are re-

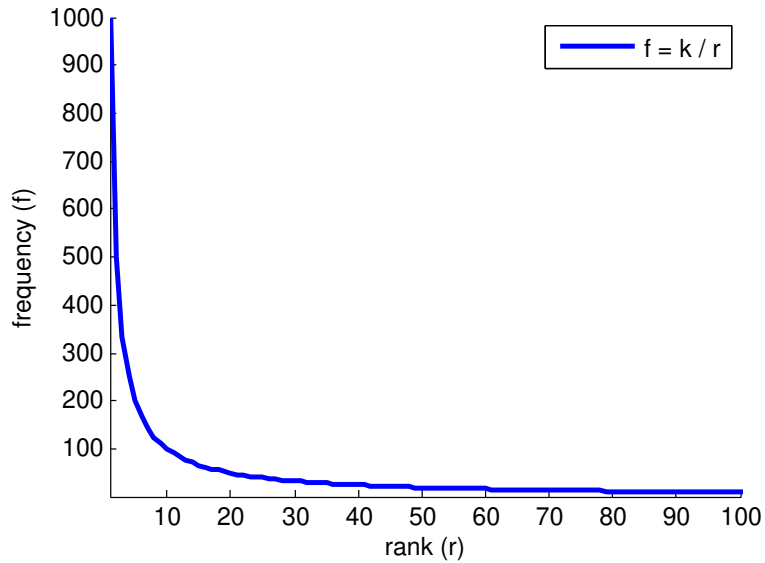


Figure 3.1. Word frequency as a function of word rank as defined by Zipf’s law.

ferred to as “unique words”. According to Zipf’s law, unique words always exist if the documents are written in some natural language. For English, half of the vocabulary of the text (or corpus) is composed of words which appear only once in the entire context [27]. In English, about 2-5% of all the words are expected to be unique for book length documents with no OCR errors. In other words, every second sentence in a book is expected to contain a word which is unique in the entire context.

Unique words typically correspond to names, places and other infrequent words in the language, such as “aliens”, “light” and “barely” as seen in Figure 3.3. These words are highly descriptive of the content and therefore strong candidates for being anchor points unlike stop words such as “the”. The property of appearing only once in the entire text makes the unique words specifically suited for serving as anchor points. If the word appears only once in both texts, then there is only one way to split the input texts into two pairs of corresponding text segments. Otherwise, if the candidate words appear multiple times in the input word sequences, then there are several possible ways to split the two texts. Among those splits, only one of them is actually correct and it needs to be determined. The unique words do not have this

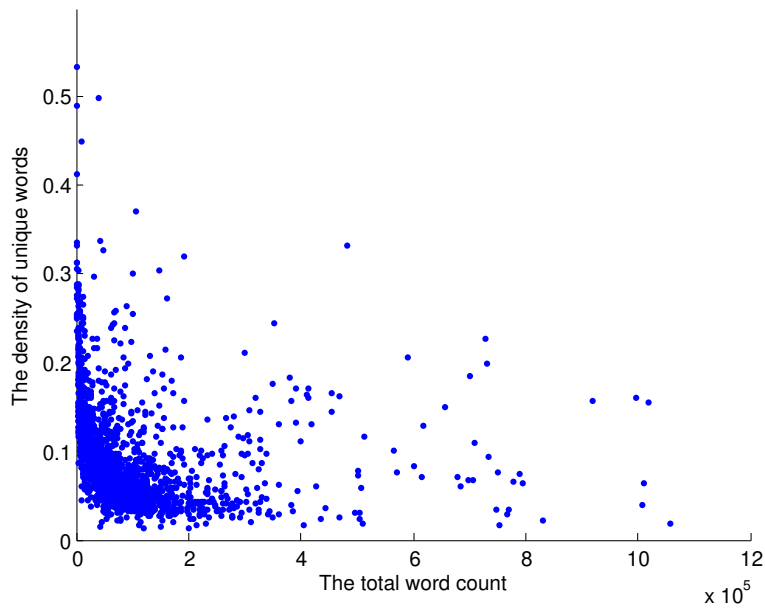


Figure 3.2. The scatter plot for the ratio (density) of unique words in the text as a function of total word count in the OCR output.

problem and therefore they are ideal for finding corresponding locations between the input texts.

From a sampling point of view, one could still use words which appear more than once for splitting the texts. The proposed approach can be simply generalized by incorporating the words whose frequency is below a “maximum term frequency” threshold. However, this approach is not as effective and efficient as using only the unique words as anchor points. This is discussed further in Section 3.2.

An important property of unique words is that the words appearing only once in the original text tend to be unique in the OCR output as well if the word is recognized correctly. The reason is that OCR errors are quite unlikely to create words which are in the vocabulary of the book. This property is therefore well-preserved across the original content and the corresponding OCR output of the same text.

The total number of unique words extracted from an OCR output is expected to be higher compared to the number of unique words in the original text without any character recognition errors. The reason is that OCR errors tend to create unique

words which are typically not even in the language itself. One could potentially eliminate these noisy words using a dictionary and/or language modeling approach. However, later we see that there is no need to detect and/or eliminate them for the alignment and matching tasks discussed in the context of scanned books. Figure 3.2 shows a scatter plot for the density of unique words as a function of the total number of words in the recognized text. The unique word density is defined to be the number of unique words divided by the total number of words in the entire text. 1700 real scanned books downloaded from the Internet Archive website are used for generating the plot. The average number of words per scanned book is 103.3K and 9.45% of those appear only once in their own context. As seen in the figure, all the books contain a number of unique words in their own context. The lowest and highest unique word density values are 0.013 and 0.71, where the unique word density variance is 0.0039. A large proportion of the unique words are expected to correspond to OCR errors if the text recognition accuracy is low. As also discussed later, these OCR errors do not affect the alignment process of the proposed approach unless the OCR error rates are very high.

3.1.2 Alignment of unique words

The unique words must follow exactly the same global order in both input texts in order for them to be valid anchor points. Otherwise the resulting splits might generate text segment pairs which do not correspond to each other. For this purpose, the unique words which are common in both texts are first identified using a hashtable in linear time. This stage eliminates most of the unique words which are created by OCR errors. Second, the sequence of unique words which are common in both texts are aligned using LCS. Only the words which are in the LCS are used as anchor points. Each chunk of texts between the consecutive anchor points is then associated with the corresponding chunk in the other word sequence. The set of corresponding

pieces of text are later forwarded to the recursive stage for generating the complete alignment.

Feng and Manmatha’s [35] use of unique words follows a different strategy for selecting the anchor points. Instead of aligning the unique word sequences, a window of four words is placed around each unique word and they are checked to see if they are the same in both sequences regardless of their original order in the input sequences. In other words, the anchor points which are used for dividing the text are not verified whether they follow the same order in both texts. Their approach is therefore more sensitive to the OCR errors (which may corrupt the word n-grams), and the edition differences between the texts where the textual content and/or the order of sections might not be exactly the same.

3.1.3 The recursive stage

The alignment of unique words produces a large number of text segments each of which is much shorter than the original input sequences. However, the stretch between two unique words may still be very large in certain cases especially for very long documents. In those cases, the alignment of unique words procedure is applied to each text segment separately in a recursive manner. It should be noted that some words which are not unique in the entire sequence become unique in the corresponding text segment. These unique words are used for dividing the text segment further into finer segments and this helps reduce the overall computational cost of the alignment. The recursion stops when the text segments gets small enough at the leaf level for dynamic programming.

Figure 3.3 depicts the proposed alignment scheme for two sample texts. In Figure 3.3a a small portion of the OCR generated text and its ground truth is shown. Unique words are colored for both texts. Aligning the unique words allows us to determine that the underlined unique words (i.e., “aliens”, “light”, “barely”) match with each

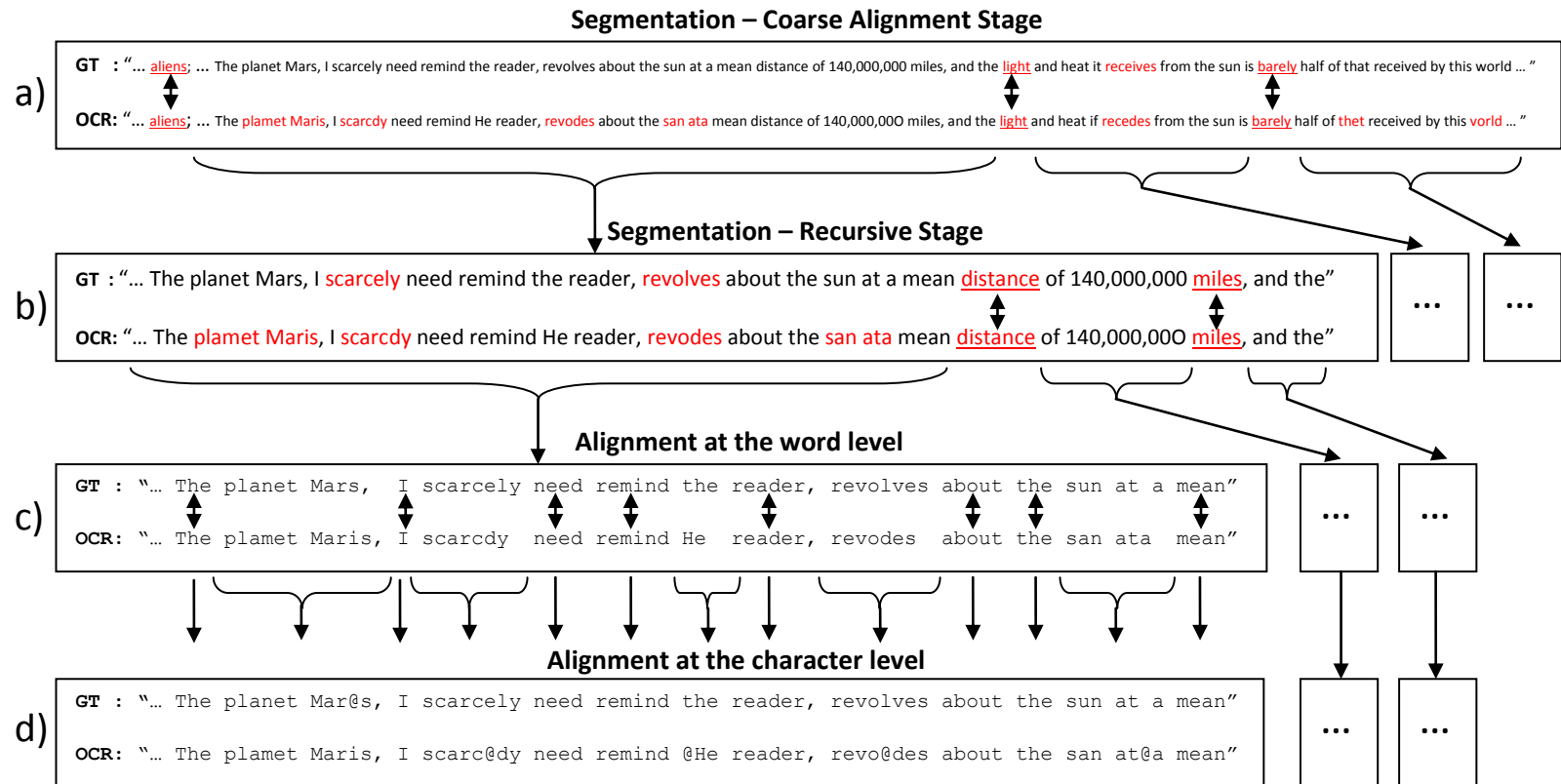


Figure 3.3. The Recursive Text Alignment Scheme (RETAS) depicted for two short texts. “...” stands for the skipped content for illustration purposes. Double headed arrows indicate matching words. “@” is a “null” indicator used for designating character insertion and deletions.

other and are used as anchor points to segment the texts. Thus the text between “aliens” and “light” forms a segment and the text between “light” and “barely” another. Notice that OCR errors generate a number of unique words such as “Plamet” and “Maris” but this does not affect the algorithm since they do not align with the other sequence and hence are not used as anchors. The next step is to align each text segment recursively. Figure 3.3b depicts the recursion for the text segment between the words “aliens” and “light”. Notice that the words “distance” and “miles” are now unique for this text segment although they are not unique in the entire book and they can be used for segmenting the text further into shorter segments.

One important point is the stopping criteria for the recursion. One could give a predetermined limit on the depth of recursion or the maximum size for a text segment. In our case, we continue text segmentation recursively until each segment become smaller than a given size K (in our case 200 words). One should avoid using a small K since stopwords become unique at the sentence level and this may yield segmentation errors. Yet another stopping condition is the absence of unique words which are common in both text segments. At the end of the recursion, a large number of short text segments are generated for the word and character alignment.

3.1.4 Word and character level alignment

Corresponding pairs of the short segments produced in the recursive stage are now aligned at the character level using an edit distance based algorithm. First, each pair of segments is aligned at the word (Figure 3.3c) level. The word alignment maps words which are the same across the pair of segments. However, due to OCR errors some words do not align and these are later aligned at the character level to produce the final alignment (Figure 3.3d). Notice in this case the word “ata” in the OCR string is aligned with “at a” in the ground truth by introducing a null character “@”

to correspond to the missing space character. It is observed that aligning words and then characters is more efficient than aligning segments directly at the character level.

At this stage the sequences are short and so the choice of the alignment algorithm usually does not make a significant difference in terms of processing time. In this work we use the standard dynamic programming algorithm for Edit-Distance [32]. The costs for insertion, deletion and replacement are taken as $[1, 1, 2]$ respectively [89]. To make sure that the the size of the dynamic programming table in memory is sufficient, it is set to have a threshold of (2 million) at both word and character level. This implies that two text segments each of which has 1000 words can be aligned even if there is no common unique word. Large stretches without any common unique word are more likely due to missing or extra text. Hence if the size of the table is likely to be over this limit then the characters are aligned with “null” indicators.

3.2 Verification of the proposed alignment scheme

The effectiveness of the proposed alignment approach is tested using texts with synthetic document noise. The synthetic noise model introduced in [35] is adopted for this purpose. In a nut-shell, this model applies basic string edit operations at the character level iteratively until the desired amount of noise is reached. The ground truth for the alignment itself is determined uniquely since the position of each deleted, inserted or replaced character in the synthetic text is known precisely. This information is used for automatic evaluation of the alignment output itself.

For the synthetic experiments, an electronic copy of the book “The Critique of Practical Reason by Immanuel Kant” (English) was obtained from the Project Gutenberg website [2]. The book is converted into a sequence of words each of which is separated by a single space character, letter cases are preserved and all punctuation letters are removed. In this form, the book has around 350K characters (including spaces) and 63K words. The noise level of a synthetic text is defined by the percent-

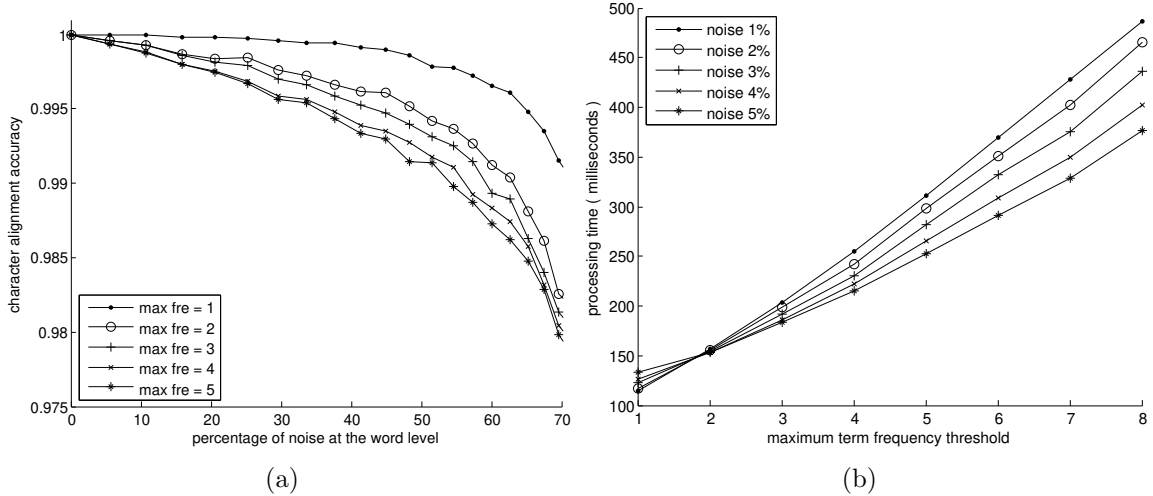


Figure 3.4. a) The accuracy and b) the speed of the character alignment versus the document noise for different values of maximum candidate term frequency threshold.

age of randomly inserted, deleted and replaced characters where the distribution of insertion, deletion and replacement operations is $[1/3, 1/3, 1/3]$ for each text. The percentage of changes (noise) is varied from 1 to 20 in steps of 1. All the experiments are repeated 100 times with different random seeds and the statistics are averaged.

In the first synthetic experiment, the accuracy of the character alignment is evaluated as a function of document noise. The “maximum term frequency threshold” is introduced to investigate the effect of term frequency in the selection of candidate anchor words. For this purpose, the proposed text alignment approach is generalized so that it uses not only the unique words but also other rare words whose term frequency is less than or equal to M as candidate anchor points. Figure 3.4a) shows that the character alignment accuracy is $\geq 98\%$ correct for different values of maximum term frequency threshold even if there exists 20% character level document noise in the synthetic text. Notice that, for 20% noise, the word error rate is over 70%. The OCR accuracy on real books is actually much higher. The alignment accuracy is maximized when the maximum term frequency threshold is set to one. This result is actually parallel with the argument that unique words are ideal for finding correspon-

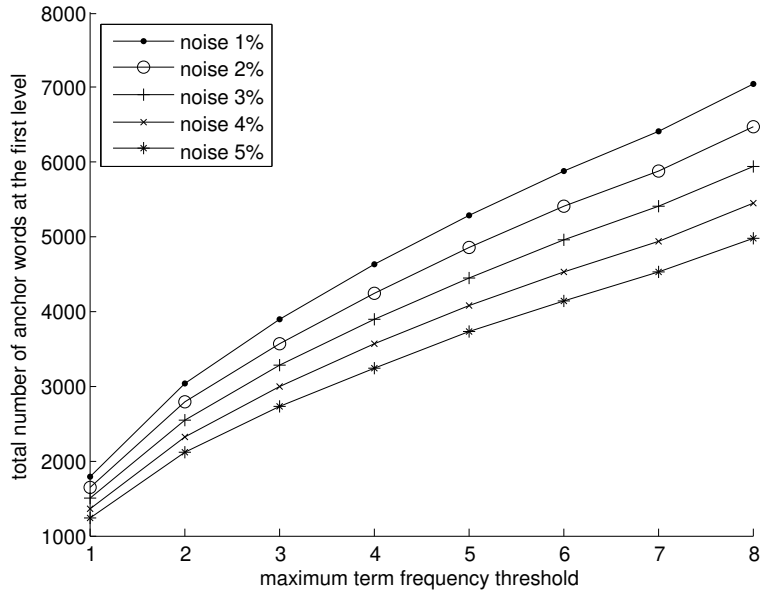


Figure 3.5. The total number of anchor words as a function of the maximum candidate term frequency threshold for different amounts of synthetic character level document noise.

dences between the two texts. As the maximum term frequency threshold increases, the character alignment accuracy falls.

Alignment errors may occur when an OCR error transforms a unique word to a legal unique word which is also present in the ground truth. For example, transforming “ball” to “call” could possibly lead to segmentation errors. However, this is unlikely to lead alignment errors. First, most OCR errors lead to words which are not present in the ground truth or even in the language. Second, even if an OCR error creates a unique word present in the other sequence, it must occur in the right place in the sequence for it to cause an alignment error.

Figure 3.4a) shows the total processing time per document as a function of document noise and the maximum term frequency threshold. It is clear that the overall processing time increases as the maximum term frequency threshold is increased. The total processing time is minimized when only the words whose frequency is equal to one (i.e., the unique words) are used as candidate anchor points. It should be noted

the speed versus the document noise behaves differently for $M = 1$ and $M > 1$. The total processing time increases as the document noise increases, when the unique words are only used for anchoring purposes. For $M > 1$, the processing time decreases as the document noise increases. Further analysis reveals that the total processing time is dominated by the first level alignment for $M > 1$. The reason is that the total number of candidate terms for the first level alignment increase rapidly as the maximum term frequency threshold increased as seen in Figure 3.5. This makes the first level LCS alignment computationally expensive despite the fact that the candidate anchor word lists of the two input texts are intersected prior to alignment for efficiently purposes. As the amount of document noise increases, the total number of candidate anchor words to be aligned by LCS decreases for a given value of M . In other words, document noise helps reduce the cost of LCS alignment at the first stage whereas it increases the cost of alignment at the leaf level. When only the unique words are used, there are fewer candidate anchor words and therefore the total processing time is dominated by the cost of alignment at the leaf level. The document noise causes the resulting text segments to be larger, and in turn, the total processing time increases as the document noise increases. In this particular synthetic experiment, for noise levels 1% and 5%, there are about 1800 and 1200 anchor words respectively to split the text of length 65K words if only the unique words are used. As a result, for these noise levels, the average text segment size aligned at the leaf level are 36 and 54 words respectively. For the maximum term frequency threshold equal to eight, there are many more anchor words to split the text. The average text segment sizes at the leaf level of the recursion therefore become much shorter (9 and 14 words, respectively).

3.3 Computational complexity

The overall cost of the Recursive Text Alignment Scheme is characterized by the total cost for the word and character level alignment at the leaf level of the recursion. For the average case, assume that each text segment is divided into k subsegments at each level of the recursion and the length of the text segments at the leaf level is K . Then, the total cost becomes $O(nK)$ since there are n/K text segments each of which takes $O(K^2)$ time to align. The alignment of unique words at each level of the recursion can also be computed asymptotically faster. There exists a LCS algorithm with an amortized cost $O(n \log \log k)$ to align two input sequences where an element does not appear more than once in either sequence although it is not used here. In theory, the worst case running time is achieved when there are no common unique words between the OCR output and the ground truth and in this case the texts have to be aligned using an exact alignment algorithm at the leaf level (i.e., $K = n$). As mentioned in Section 3.1.1, this is a very unlikely scenario and never happens in practice.

3.4 Efficiency

The first stage of the proposed approach is to extract the unique word sequences from each input file. For this purpose, the frequency of words in each input text is determined in linear time using a hashmap data structure. The unique words are then sorted according to their original order in the text. Sorting takes $O(n \log n)$ time. The sorted sequence can also be generated in linear time using a memory based approach. A Boolean array of size n can be used to mark each unique word where n is the total number of words in the text. The unique word sequence can be simply recovered by iterating over the boolean array. Determining the sequence of unique words for each text segment can be performed in the same fashion at different levels of recursion.

The sequences of unique words are aligned at every stage of recursion using LCS. The asymptotically faster $O(n \log \log n)$ LCS algorithm [49] is applicable to this particular case since the words in each input sequence appears at most once in either sequence. Instead, the standard $O(n^2)$ LCS algorithm is preferred in our case since it is quite fast for aligning short sequences of size a few hundred or thousand words. The standard dynamic programming implementation can be speeded up drastically. If the word does not appear in the other sequence, then it can not be in the LCS. The words which does not appear in the other sequence can be therefore eliminated before the LCS alignment. The word elimination process can be achieved in linear time using a hash table. This helps reduce the length of the unique word sequences before the alignment. It should be noted that the word elimination procedure is not applicable to other sequence alignment approaches such as Needleman-Wunsch and Edit-distance. In those cases, the entire input sequences are necessary to compute the alignment. LCS is therefore the ideal choice for aligning unique word sequences.

At the end of recursion, the resulting text segment pairs are relatively short - typically a few hundred words each. At this scale, the alignment of text segments can be performed efficiently using any alignment approach depending on the specifics of the application. In the case of OCR evaluation and error correction tasks, Edit distance might be more desirable since it explicitly accounts for the character/word replacements in the cost function. In our framework, the alignment is performed using Edit distance by assigning the insertion, deletion and replacement costs to [1,1,2] respectively. This cost function does not allow replacements and the generated alignment output is exactly the same as LCS at the leaf level.

3.5 Evaluation of OCR accuracy for real scanned books

A number of scanned books in different languages are downloaded from the Internet Archive's website [1] and their OCR accuracies are evaluated. According to the

metadata, these books are recognized using ABBYY FineReader 8.0. The original texts of these books (i.e., the ground truth texts) are also obtained from the Project Gutenberg website [2]. These public domain books have been proofread by volunteers and are, therefore, mostly free of OCR and transcription errors. One issue with these books is that formatting information (line, page breaks) has been removed so that we are essentially left with one long string of possibly half a million characters. Our approach is therefore to align the OCR output with the Gutenberg version of the text as suggested in [35]. The OCR accuracy metric is defined as follows:

$$OCR_{acc} = \frac{m}{c} \tag{3.1}$$

where m is the total number of matching characters/words in the alignment and c is the total number of characters/words in the ground truth. This metric accounts for the containment of the ground truth text in the OCR output. The rationale behind this approach is to obtain a statistical evaluation of the OCR accuracy for the portion of the text for which we have ground truth. Note that the scanned text may have extra portions (e.g. an extra introduction) and the metric is not sensitive to such text. With the reasonable assumption that the rest of the book is similar we can assume that the estimated OCR accuracy is true for portions for which we have no ground truth.

The first experiment is to evaluate the accuracy of the OCR evaluation framework. A number of synthetic texts at varying amounts of noise are generated as described in the previous section. Next, the synthetic texts are aligned with the original text using the proposed alignment scheme and the corresponding OCR accuracy values are estimated. Figure 3.6 shows both the ground truth and estimated character, word and stopword accuracies using the proposed alignment approach for OCR evaluation purposes. The stopword list consists of the top 100 most frequent words in English trained using fifteen books from the Project Gutenberg. Note that the curve for

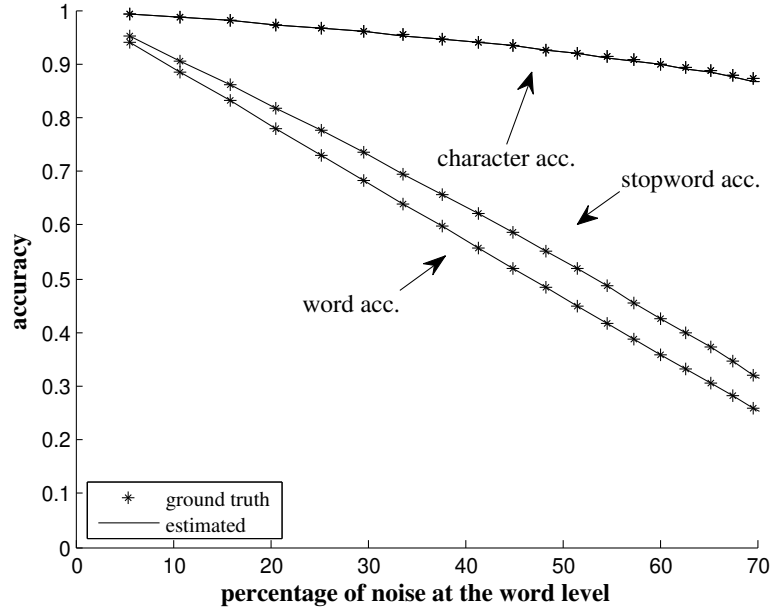


Figure 3.6. Estimated and ground truth OCR accuracies for characters, words and stopwords versus document noise.

accuracy estimations are almost overlaid over the plot for the ground truth values. This implies that the proposed methodology can be successfully used for estimating OCR accuracies as demonstrated for real scanned books.

Estimated character and word accuracies for the real scanned books are shown in Table 3.1. for four languages using the Latin alphabet. Both word accuracies and character accuracies are directly estimated. English is the most accurately recognized. The word accuracy for Spanish is slightly higher than for French but the character accuracies are reversed (this reflects the fact that word and character statistics depend on language). It is clear that the average OCR word error rate is about 7% for English and more than 10% for other languages. The highest character recognition accuracy is reported for English. There has been more work on recognizing English than other languages. The lowest word recognition accuracy is for German. Notice that the average word length in German is longer than the other languages listed in the table. Clearly there is scope for substantial improvement in preprocessing and the OCR

Table 3.1. Estimated character and word OCR accuracies for books in English, French, German and Spanish from the Internet Archives. Punctuations are ignored.

Dataset	#books	average word length	OCR word accuracy	OCR character accuracy
English	100	4.45	0.934	0.973
French	20	4.91	0.883	0.961
German	20	5.66	0.878	0.949
Spanish	20	4.83	0.900	0.959

itself for non-English languages. The character accuracy rates even for English do not reach 99% indicating that there is potential for improvement there too.

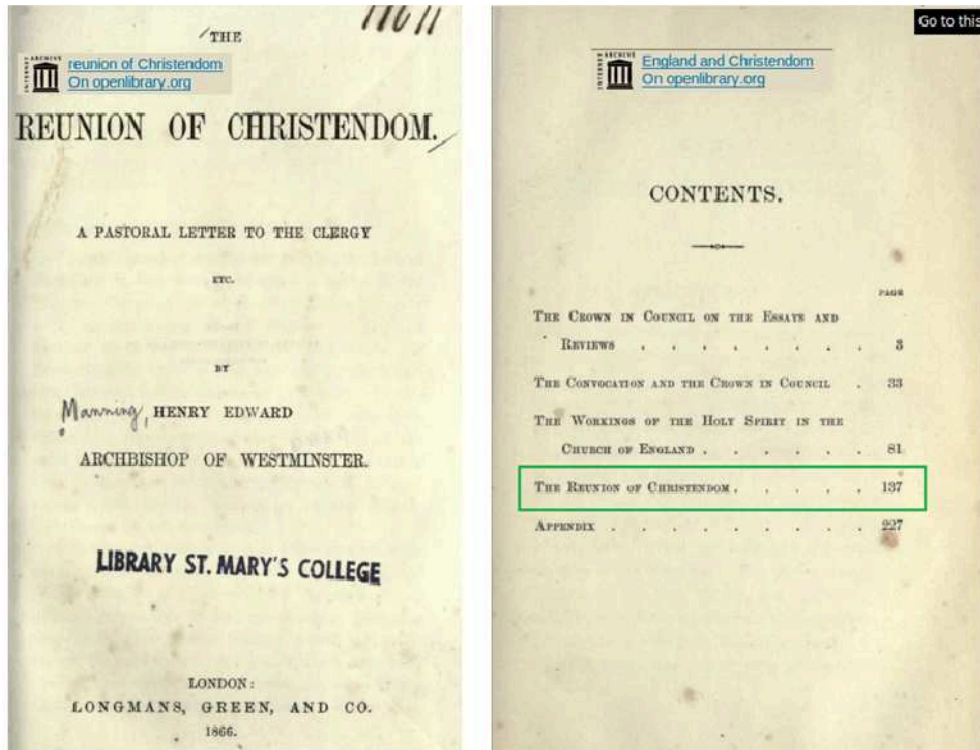
In this chapter an efficient text alignment framework is proposed for long noisy texts and it is used for evaluating OCR accuracy of scanned books. The next chapter introduces a framework which uses the sequence of unique words alignment concept for efficient detection of partial duplicates in scanned book collections. In the next chapter it is also shown that the proposed text alignment approach can be used to visualize the duplicated portions of text for books which are known to be partial duplicates.

CHAPTER 4

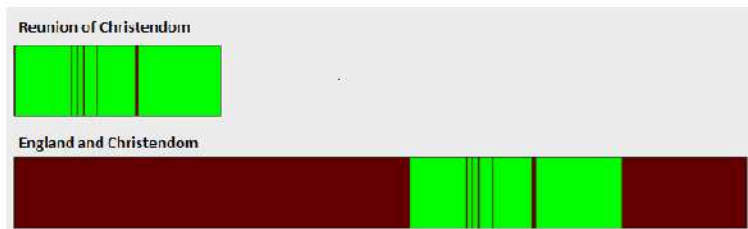
PARTIAL DUPLICATE DETECTION FOR LARGE SCANNED BOOK COLLECTIONS

A pair of books is a partial duplicate if there is a significant amount of content overlap in the form of duplication or a slight modification. Examples of partial duplicates include different editions, versions, prints and compilations of books. A book may include an entire book, the main text of a play, a story from a selection of short stories or long excerpts. Figure 4.1 shows an example where a book is entirely subsumed by another one. A small amount of duplication at the passage level or in the form of short quotations might exist in almost any book and this information is not sufficient for books to be partial duplicates. For example, a physics book might have a quote from Shakespeare’s Hamlet but this does not make it a partial duplicate of Hamlet (see [97] on finding quotations). Books on the same topic (e.g. optics) are not necessarily partial duplicates either. Topic detection techniques are therefore not applicable to this problem domain.

Scanned books are different from traditional web or born electronic documents. Different editions, versions, prints and compilations of books are often not straight copies or near duplicates but may show a lot of variation. These variations include not only OCR errors and language differences, but also additional, missing or modified sections. The amount of variation may therefore be quite significant. For example, Figure 4.2 shows an example where both versions of Shakespeare’s “Othello” contains the entire main text whereas the variorum edition is three times longer because of additional content in the form of footnotes on each page. Figure 4.3 shows another example where one of the two versions of the book includes an additional essay written

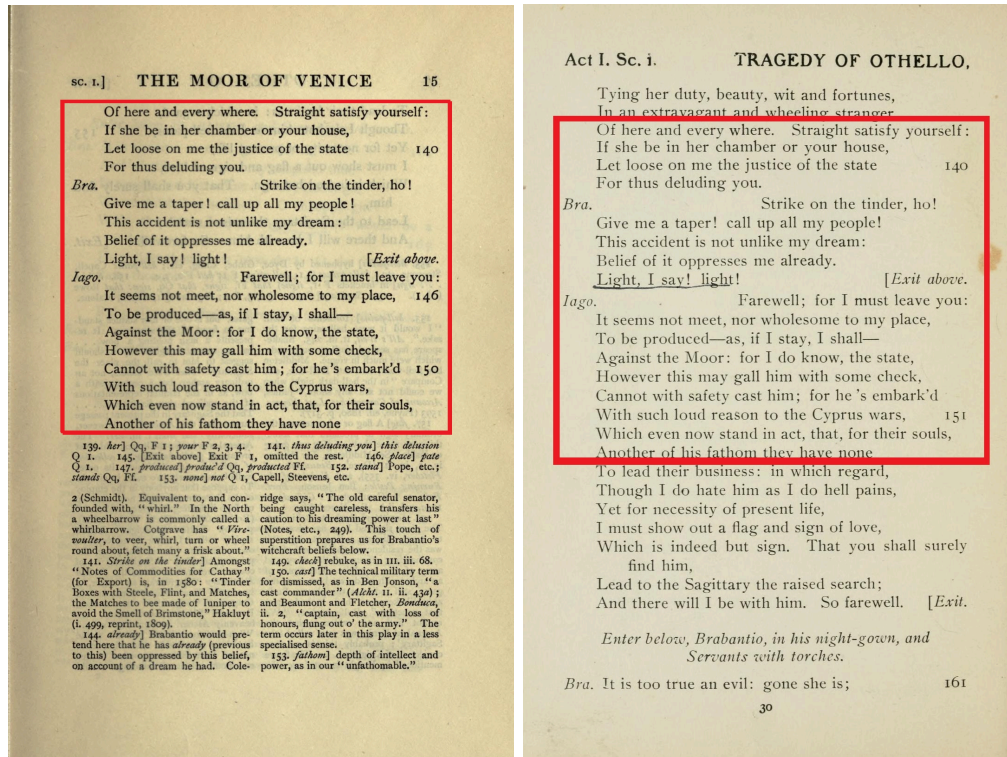


(a) The book covers for books “Reunion of Christendom” (left) and “England and Christendom” (right)



(b) The overlapping portions of the texts are visualized in green. The width of the rows indicate the relative lengths of the books.

Figure 4.1. An example pair of books which are partial duplicates of each other. One book subsumes the other.



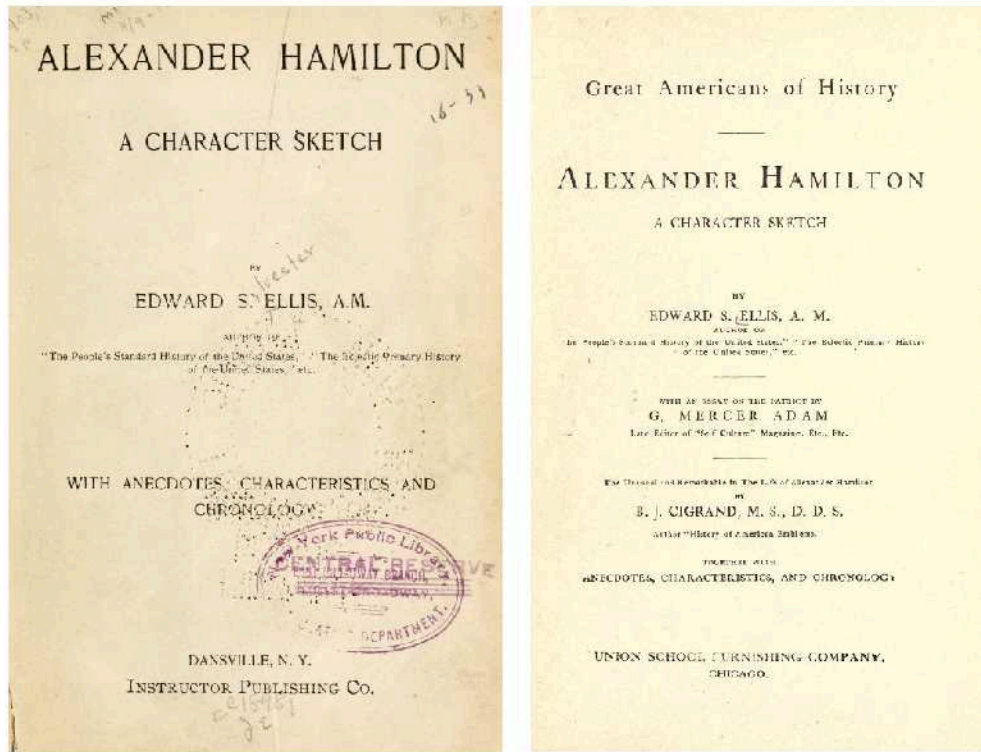
(a)

(b)

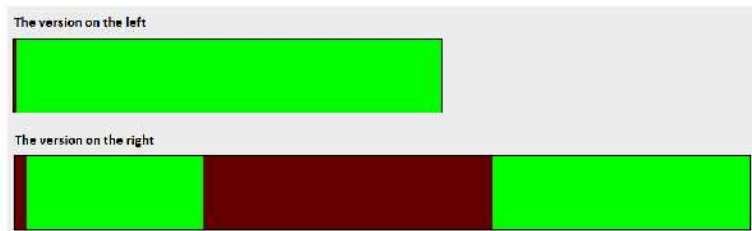
Figure 4.2. Two sample pages from two different versions of Shakespeare's Othello downloaded from the Internet Archive's website. The duplicate text is shown with red rectangles.

by a different author. Finding near duplicates in document collections (where the content overlap is much higher, say 80% or more [115]) have been extensively studied in the literature. It should be noted that exact and near duplicates are both special cases of partial duplicates.

Knowing the editions, versions and duplicates of books helps us understand the textual contents. This type of background information is essential especially for scholars in social disciplines. Search engines may therefore display different versions of a book as part of the results. For example a reader may want to read the original version of Shakespeare's Othello whereas some other reader may choose to read the modern English version with footnotes. There are several other uses of this type of information. For example, one can propagate annotations from one book to its



(a) The book covers for two different versions of the book “Alexander Hamilton: A character sketch”



(b) The overlapping portions of the texts are visualized in green. The width of the rows indicate the relative lengths of the books.

Figure 4.3. An example pair of books which are partial duplicates of each other. The version on the right contains an extra essay in the middle section.

#	Creator	Title
1	No Author	Shakespeare's Tragedy of Othello
2	William Shakespeare and H. H. Furness	A New Variorum Edition of Shakespeare
3	William Shakespeare, Tommaso Salvini and James Henry Mapleson	Othello: A Tragedy in Five Acts
4	Walter E. Hoffman	A modern Othello

Table 4.1. Example records for *Othello* from the Internet Archive's catalog. The first and the third records include the original text of Othello with significant amount of additional text in the form of commentary and footnotes. The second book consists of a number of works written by Shakespeare also including the full text of Othello. The last record is an entirely different work written by a different author. The first three books are partial duplicates of each other.

duplicates and translations in the collection. It is also possible to improve text search by refining the results based on the linkage between the books in the collection. Duplicates of books can also be used to correct text recognition errors. The humanities and library communities also have a great interest in aggregating works. IFLA's Functional Requirements for Bibliographic Records (FRBR) requires that the next generation of cataloging systems include works aggregation where different versions, editions, translations and copies of each book is documented [71]. However, no specific technique is proposed to create FRBR catalogs and it is implicitly assumed that the metadata will be sufficient.

Scanned book collections provide metadata for each book in the form of title, author, year and language. However, the metadata is not completely reliable and includes a large number of typos, mistakes and incomplete information. Such information is typically transferred manually from the library catalog or entered by the people who actually scan the books. Besides, certain types of information can not be inferred directly from the fields of the metadata. Table 4.1 shows an example where the metadata is not sufficient to find the partial duplicates of books. Finding the translations of books using the book metadata is even more complicated since it requires matching titles and names across languages. Yet a more challenging problem

is to line up duplicated and/or translated portions of texts given a pair of books. In all these cases, it is necessary to use the textual content of the books directly in the process.

The problem is to find all the partial duplicates in a given collection of scanned books. A solution is to compare all $n(n - 1)/2$ pairs of books to check whether they are partial duplicates where n is the size of the collection. One can use text alignment approaches to locate overlapping content for each book pair. However, this approach is expensive since large scanned book collections typically contain millions of books and each book has hundreds of thousands of words. For example, a small collection with one thousand books has approximately half a million book pairs. Assuming that the alignment takes 0.1 second per book pair (as in the case of Recursive Text Alignment Scheme introduced in Chapter 3), it would take 13 hours on a single core to find all the partial duplicates even for such a small collection.

In a nut-shell, our proposed approach uses “the sequence of unique words” representation introduced in Chapter 3. Given a pair of books, the partial duplication is efficiently determined by aligning their representations using a Longest Common Subsequence algorithm. It should be noted that only the first level alignment of the Recursive Text Alignment Scheme presented in the previous chapter is used. The LCS length is expected to be larger if the books compared have a significant content overlap. Two duplicate scoring functions are defined over the LCS length for identifying partial duplicates of books. It is shown that the sequence of unique words alignment approach is sufficient to find partial duplicates efficiently at a rate of 12K book pair comparisons per second per core. On a collection of 100K scanned English books the proposed framework detects partial duplicates in 30 min using 350 cores. In term of effectiveness, it outperforms several baselines including shingling and Discrete Cosine Transform fingerprinting approaches. The technique works on other languages as well and is demonstrated for a French dataset.

Yet another problem is to locate duplicated portions of texts for a given pair of books which are known to be partial duplicates. In Section 4.3, we introduce an alignment based approach to map and visualize the overlapping contents of books. The details of the proposed approaches are elaborated in the following subsections.

4.1 The proposed framework

The “sequence of unique words” representation introduced in Chapter 3 is used for efficient analysis of the textual contents of scanned books. Each book pair in the collection is compared using a Longest Common Subsequence algorithm. If the book pair has some content overlap, then the LCS length is expected to be significantly higher than for a non-duplicate pair. Therefore the LCS length is used to score each book pair as to whether they are duplicates as described in Section 4.1.2. Figure 4.4 depicts the proposed framework for two short poems. Having a large number of unique words following the same order is a clear indication of duplicate text. It should be noted that unique words are much sparser for book length documents. Partial duplicates of books typically contain hundreds or even thousands of unique words common to both texts which follow the same order. The details of the sequence of unique words and the proposed score functions are discussed in the following subsections.

4.1.1 The document representation

The sequence of unique words representation incorporates the discriminatory power of infrequent words. According to Zipf’s law, the frequency of a word is inversely related to its rank in the word frequency table of a given document. This is true for documents written in some natural language. For example, the word “the” is the most frequent word in English and it constitutes about 6-7% of all the words in a document. The second most frequent term is “of” whose expected frequency is about 3-3.5%. These frequent terms alone give very little evidence about the content

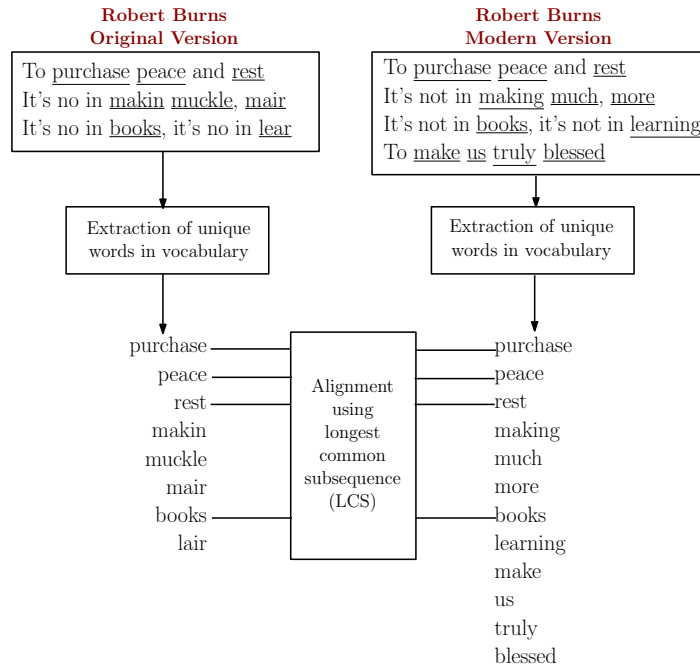


Figure 4.4. Illustration of partial duplicate detection for the two versions of Robert Burns' poem. Unique words are underlined and listed according to their original order in the text. The two sequences of unique words are finally compared using LCS alignment.

of the text. On the other hand, some terms such as names and places appear rarely and they are very descriptive of the content. It is typically desirable to give more importance to those infrequent terms in various search and retrieval tasks. The most discriminative terms are generally the least frequent terms, namely the ones which appear only once in the entire context of a document. Along with the sequence information, unique words are highly descriptive of the content and flow of ideas in the book.

One of the implications of Zipf's law is that unique words always exist if the documents are written in some natural language, as discussed in Section 3.1.1. Approximately half of the words in the vocabulary of a document appear only once if it is written in English. In English, about 2-5% of words are expected to be unique for a book length document with no OCR errors. The sequence of unique words

representation is therefore compact and efficient for comparing books. The sequence of unique words has a storage overhead of a few kilobytes per book after hashing each unique term into a discrete value. OCR errors also tend to create words which may not even be in the vocabulary of the language. Therefore the number of unique words is expected to be higher for scanned books but it is not a problem for the proposed technique. There will still be a large number of unique words recognized correctly in the sequence.

4.1.2 Scoring schemes for document pairs

The LCS length is used to detect duplicates. However, the LCS length is a function of book length and needs to be normalized. Below two normalization schemes are proposed. Other approaches involving normalizing the LCS length by the minimum or average of the lengths of the two sequences did not work well and therefore, are not reported.

4.1.2.1 Correlation Score (CS)

The CS score for two (unique word) sequences X and Y is defined by analogy with correlation as:

$$cs(X, Y) = \frac{|LCS(X, Y)|}{\sqrt{(|X||Y|)}} \quad (4.1)$$

where $|LCS(X, Y)|$ is the LCS length of the two sequences. $|X|$ and $|Y|$ denote the length of X and Y respectively. The range of values is $[0,1]$ and the highest score is obtained when the two sequences are identical.

4.1.2.2 Information Theoretic Score (ITS)

From an information theoretic point of view, the similarity between two objects X and Y maybe defined as [58]:

$$similarity(X, Y) = \frac{\log \Pr(common(X, Y))}{\log \Pr(description(X, Y))} \quad (4.2)$$

Table 4.2. DUPNIQ-cs and DUPNIQ-its scores for three pairs of books. X and Y refer to the sequences of unique words for Book X and Book Y respectively. GT is the ground truth.

Book X	Book Y	#words in X	#words in Y	$ X $	$ Y $	$ X \cap Y $	$ LCS $	DUPNIQ		GT
								cs	its	
Shakespeare’s Law; S. G. Greenwood	Shakespeare’s Law; S. G. Greenwood	14967	15079	2419	2421	2016	2009	0.8301	0.9568	Yes
Departmental ditties and other verses; R. Kipling	Departmental dit- ties ballads bar- rackroom ballads & other verses; R. Kipling	48637	27880	9292	5698	2596	1783	0.2450	0.7889	Yes
Metrical Transla- tions; J. Muir	Pride and Preju- dice; J. Austen	15372	129647	3247	9192	337	51	0.0093	0.4172	No

where X and Y are any objects generated by a probabilistic model. According to the formula, two objects are more similar if they share more features. The similarity value is maximized when the two objects are identical.

In our case, X and Y are sequences of unique words. The overlapping content between X and Y is defined by the longest common subsequence:

$$common(X, Y) = LCS(X, Y) \tag{4.3}$$

and the total information content (description) of X and Y is defined by the alignment of X and Y . Assuming that the probability of any word sequence is inversely proportional to its length, then Eq.4.2 simplifies as:

$$its(X, Y) = \frac{\log |LCS(X, Y)|}{\log (|X| + |Y| - |LCS(X, Y)|)} \tag{4.4}$$

The score has a range $[0,1]$ and the maximum value is obtained when the sequences are identical. If $|X|$ and $|Y|$ have no common words, then the score is assumed to be zero.

Table 4.2 shows two duplicates (first two rows) and a non-duplicate (last row). The names of the scoring schemes are preceded by the word DUPNIQ to denote duplicate

detection using the sequence of unique words representation. Duplicates of books have higher duplication scores. The thresholds are 0.12 and 0.72 for DUPNIQ-cs and DUPNIQ-its respectively.

4.2 Duplicate detection experiments

4.2.1 Datasets

A number of datasets with different characteristics are created using the scanned books downloaded from the Internet Archives’s website [1]. The OCR text outputs are extracted and preprocessed for duplicate detection purposes. More specifically, punctuations are ignored, letter cases are folded and empty spaces are collapsed. The hyphenated words at the end of each line are also merged if the hyphen is followed by white space characters terminated by a new line character. In a typical book, this helps recover hundreds of words from the text. In the case of DUPNIQ-its and DUPNIQ-cs, the numeric letters are also ignored. The reason is that page numbers typically become unique in the entire text and they follow a specific order. The ground truths for the datasets are obtained using a manual technique except for the 100K set. A pooling technique is used for generating the ground truth of the 100K set due to its large size and it is elaborated below. All the datasets except the 100K are publicly available ¹. Details about the datasets used in the experiments are given below.

- The **Training Set** consists of 151 English books containing 67 duplicate pairs labeled manually. This set is used to learn duplicate detection score thresholds.
- The **1K Set** consists of 1,092 English books containing 258 duplicate pairs.
- The **3K Set** consists of 2,884 French books containing 483 duplicate pairs.

¹<http://books.cs.umass.edu/downloads/dup-detect/>

- The **Partial Set** contains 458 books. There are 460 partial duplicates in total. The content overlap in this dataset ranges from 15% to 80%. This dataset is used for evaluating the success for the case when the content overlap is small.
- The **100K Set** is a large dataset consisting of 103,455 English books containing 45,485 duplicate pairs. The metadata alone is not sufficient to create the ground truth since it may be missing, incorrect or incomplete. Instead a pooling approach is used to create the ground truth. The outputs of the three different techniques (i.e. DUPNIQ-cs, DUPNIQ-its, and Shingling) are used to determine a set of candidate book pairs. This is achieved using a much looser threshold on their respective translational similar scores. Next the entire text of the candidate pairs are aligned as shown in Figure 4.8. The candidate pairs are labeled as partial duplicates or not after the visual inspection of the alignment output.

4.2.2 Baselines

A number of baseline approaches are implemented and tested on the scanned book collection datasets. The baselines primarily differ in the way they represent the text as elaborated below.

- **Unique Word Overlap (UWO)**: Each text is represented by the set of words which appear only once in the entire text without any word sequence or position information. This is a bag-of-words approach and it does not exploit the global word order in the text. This baseline is primarily designed to test the importance of the sequential information for the proposed text representation scheme.
- **Vocabulary Overlap (VO)**: Each text is represented by the set of words which appear in the vocabulary of the text itself. This approach is to test the effectiveness of the vocabulary based duplicate detection approaches. It

should be noted that a significant portion of the vocabulary words are actually composed of unique words. Specifically, half the vocabulary words are expected to appear only once in the entire text. Therefore the VO text representation scheme uses a superset of the words used by the UWO baseline.

- **Shingling ($0 \bmod p$)** [14, 64]: A moving window of size n words is used to extract a number of shingles (n -grams of words) from the text. For a text of length m words, there are $m - n + 1$ number of shingles in total. There are a large number of shingles for book length documents. A common approach is to use the zero mod p sampling scheme to represent the text using a subset of its shingles. Basically, a hashcode is computed for each shingle and the ones whose mod p value is equal to zero are selected for representation purposes. In this particular application, MD5 is used as the hash function. Several values for n (2, 4, 8) and p (10, 25, 50) are tested and the best parameters ($n = 4, p = 50$) are used for experiments.
- **Discrete Cosine Transform (DCT) fingerprinting** [97]: The DCT fingerprinting approach first splits the input text into a number of non-overlapping text segments. This splitting operation is called “hash-breaking”. It is achieved by computing the first 32-bits of the MD5 hashcode for each word and finding the ones whose mod p value is equal to zero. Those selected words are later used to split the text into a number of non-overlapping text segments. Each resulting text segment might have an arbitrary length. The DCT fingerprinting approach eliminates text segments of length smaller than the parameter p . For the remaining text segments, a 32-bit hashcode is generated using an approach called DCT fingerprinting. In a nut-shell, a 32-bit MD5 hashcode is computed for each word in the text segment. The sequence of hashcodes is regarded as a time series signal and each hash value is mapped to a value in the interval $[-1,1]$.

The coefficients of the DCT transform of the resulting signal is used to set the bits in the 32-bit DCT fingerprint. The text is represented by the set of DCT fingerprints computed for each text segment. The parameter p is typically set to be a small number (2,3 or 4) to detect local text reuse and this parameter is set to four in our experiments. The expected number of text segments is therefore equal to $N \div p$ where N is the total number of words in the entire text. Overall, DCT fingerprinting is less sensitive to local word changes compared to the shingling approach. Two different text segments with minor differences might end up getting the same DCT fingerprint. This is a desirable effect in the context of text reuse detection task for which the DCT fingerprinting technique is primarily designed. Insensitivity to the minor modifications in the text also makes the DCT fingerprinting approach viable in the context of finding duplicates in scanned book collections. Edition differences and OCR errors also introduce word and/or character insertions, deletions and replacements in the text. However, the local changes might be severe if the OCR error rates are high.

- **I-Match** [22]: [22] selects a subset of the words in the vocabulary of the text, sorts them according to their lexicographical order and finally hashes the aggregate string into a value. These hash values are compared to determine duplication. The word selection process is based on the inverse document frequency values (IDF) of the terms computed over the entire text collection. Different word selection approaches are presented in the original paper. Notice that if the input text includes an additional or missing word in the vocabulary, it is very likely that the text obtain a different hash value from the original text. I-Match is primarily designed for finding near-duplicates of web pages and it is not sensitive to the type of text duplication found on the web. However, in the context of scanned books, the partial duplicates of books might have signifi-

cantly different vocabulary words due to OCR errors, additional or missing text. The global hash function of I-match is quite sensitive to even small numbers of vocabulary words being wrong. The vocabulary of a book typically contains tens of thousands of words and some of them are not recognized correctly in different versions of the same book. As a result, I-Match performed poorly in all experiments and it is not discussed further.

Except for I-Match, all the baselines use Jaccard similarity² as the duplicate detection score. The duplicate detection thresholds are estimated by maximizing the F-measure (the harmonic mean of recall and precision) over the Training set for all baselines, DUPNIQ-its and DUPNIQ-cs.

4.2.3 Evaluation of duplicate detection results

Table 4.3 shows the precision, recall and F-measure scores for all baselines, DUPNIQ-its and DUPNIQ-cs over the Training, 1K, 3K and Partial set. The estimated duplicate score thresholds for DUPNIQ-its and DUPNIQ-cs are 0.72 and 0.12, respectively. The same thresholds are used for all datasets including the 3K French dataset. For 1K, 3K and Partial test sets, DUPNIQ-its outperforms all other approaches in terms of F-measure. It should be noted that the score threshold trained on a set of English books generalized on the 3K French dataset. DUPNIQ-cs provides slightly lower F-measure scores compared to DUPNIQ-its. Vocabulary overlap (VO) and DCT fingerprinting techniques performed much worse compared to other approaches on the 3K French and Partial duplicates sets. Between UWO and VO, UWO is the best on most counts showing that the unique words are more discriminative than the entire vocabulary. Besides, UWO is much faster since it has many fewer words to deal with. Shingling performed reasonably well on the 1K and 3K sets, however, it missed a

² $Jaccard(A, B) = |A \cap B| / |A \cup B|$

Table 4.3. Precision (P), recall (R) and F-measure (F) scores for the Training, 1K, 3K and Partial Duplicates sets.

Dataset		DUPNIQ cs	DUPNIQ its	UWO	VO	Shingling	DCT
Training	P	0.984	1	0.970	1	0.971	1
	R	0.940	0.955	0.970	0.836	1	0.925
	F	0.962	0.977	0.970	0.911	0.985	0.961
1K set	P	1	1	0.987	1	0.980	0.971
	R	0.938	0.953	0.915	0.818	0.938	0.911
	F	0.968	0.976	0.950	0.900	0.958	0.940
3K set	P	0.913	0.954	0.928	0.976	0.926	0.958
	R	0.938	0.946	0.878	0.583	0.882	0.703
	F	0.925	0.950	0.902	0.730	0.903	0.811
Partial	P	0.924	0.995	0.919	0.990	0.989	0.996
	R	0.957	0.904	0.959	0.674	0.761	0.570
	F	0.940	0.948	0.938	0.802	0.860	0.725

Table 4.4. Precision (P), recall (R) and F-measure (F) scores for the 100K dataset.

Approach	P	R	F
DUPNIQ-cs	0.903	0.933	0.912
DUPNIQ-its	0.996	0.833	0.907
Shingling	0.992	0.720	0.834

large number of positives in the Partial set. The overall performance of the DCT fingerprinting approach is worse than the Shingling approach, although previously it has been shown to perform better than other n-gram (Shingling) based approaches in the local text reuse problem domain. [97]. The observation is that the hash-breaking approach used for splitting the text into smaller segments is quite sensitive to OCR errors. As a result, a smaller number of text segments are preserved across different books, despite the fact that the actual DCT fingerprints computed for each segment are less sensitive to minor modifications or OCR errors. The negative effects of the hash-breaking approach is discussed further in the synthetic experiments section. It should be noted that the DCT fingerprinting approach is the most computationally expensive approach among others since it generates a large number of fingerprints to represent the document.

The results for the top three best performing approaches are shown on the large 100K set in Table 4.4. DUPNIQ-cs and DUPNIQ-its did much better than any other technique in terms of F-measure (DUPNIQ-cs - 0.912, DUPNIQ-its - 0.907, UWO - 0.724, VO - 0.552, Shingling - 0.834). As the results show, both DUPNIQ techniques perform better than Shingling in terms of the F-measure and recall. Higher recall values are desirable since more exhaustive approaches can be employed as a post processing step to eliminate false positives, if necessary. Further analysis on the 100K experiments show that some duplicates are missed by DUPNIQ-its and DUPNIQ-cs because their score is slightly lower than the threshold even though they have a high LCS length. This indicates that there is scope for further improvement in the score normalization stage. 14% of the false negatives are due to dictionaries and encyclopedias. It should be noted that the technique is not meant to work on such books because of their alphabetical ordering. 16% of the false matches are religious book pairs (e.g. gospels, hymns, and prayer books), 12% are literary books, and 6% of them are technical book pairs. The flavor of results obtained is similar to that shown in Table 4.1. There are also some unusual partial duplicates. For example, two technical reports published by Johns Hopkins University matched because they contained very similar mailing distribution lists at the end.

4.2.4 Word sampling experiments

The sequence of unique words text representation scheme uses the words that appears only once in the entire text for representation purposes. From a sampling point of view, one could also select other rare words such as the words which appear twice or three times in the entire text. The effects of word sampling based on their frequency in the text are investigated next. The “maximum term frequency threshold” M is introduced for this purpose. If a word’s frequency is equal or less than M , then it is used for text representation purposes. All the selected words are sorted according

Table 4.5. Precision (P), recall (R) and F-measure (F) scores of the proposed approach (DUPNIQ-its and DUPNIQ-cs) for the Training, 1K, 3K and Partial Duplicates sets. M and T correspond to the maximum term frequency and duplicate score thresholds respectively.

Approach	M	T	Training Set			1K Set			3K Set			Partial Set		
			P	R	F	P	R	F	P	R	F	P	R	F
DUPNIQ-its	1	0.72	1	0.955	0.977	1	0.953	0.976	0.954	0.946	0.95	0.995	0.904	0.948
DUPNIQ-its	2	0.74	1	0.955	0.977	1	0.953	0.976	0.954	0.940	0.947	0.995	0.830	0.905
DUPNIQ-its	3	0.75	1	0.955	0.977	1	0.950	0.974	0.963	0.925	0.944	0.994	0.75	0.855
DUPNIQ-its	4	0.74	0.956	0.970	0.963	0.996	0.953	0.974	0.928	0.938	0.933	0.992	0.796	0.883
DUPNIQ-its	5	0.74	0.956	0.970	0.963	0.992	0.950	0.970	0.928	0.938	0.933	0.992	0.776	0.870
DUPNIQ-cs	1	0.12	0.984	0.940	0.962	1	0.938	0.968	0.913	0.938	0.925	0.924	0.957	0.940
DUPNIQ-cs	2	0.13	0.970	0.955	0.962	1	0.950	0.974	0.907	0.932	0.919	0.950	0.943	0.947
DUPNIQ-cs	3	0.13	0.970	0.955	0.962	0.996	0.950	0.972	0.905	0.934	0.919	0.958	0.935	0.946
DUPNIQ-cs	4	0.12	0.956	0.970	0.963	0.98	0.950	0.965	0.890	0.936	0.912	0.960	0.937	0.948
DUPNIQ-cs	5	0.12	0.956	0.970	0.963	0.98	0.950	0.965	0.894	0.929	0.911	0.966	0.913	0.939

to their original order in the entire text and they are aligned using LCS as described for DUPNIQ. Notice that the sequence of unique words scheme is equivalent to using $M = 1$. For the values of M greater than one, each word might appear more than once in either sequence. Clearly the asymptotically faster $O(n \log \log n)$ LCS alignment approach is not applicable in those cases. It should be noted that a space efficient LCS algorithm is used in all our duplicate detection experiments [45].

Table 4.5 shows the precision, recall and F-measure scores for the Training, 1K, 3K and Partial sets where the maximum term frequency threshold M is varied from one to five. DUPNIQ-its and DUPNIQ-cs thresholds are retrained for different values of M from the Training set and the same threshold is applied on the test sets. The F-measure score is maximized for DUPNIQ-its when $M = 1$, although $M = 2$ and $M = 3$ also give comparable results on the Training and 1K sets. In the case of DUPNIQ-cs, there is no significant correlation between the F-measure score and the threshold M . In terms of F-measure, DUPNIQ-its using only the unique words for text representation purposes outperforms DUPNIQ-cs for all values of M except the Partial set where the F-measure scores are equal.

To sum up, the duplicate detection experiments indicate that including additional words from the original text into the text representation does not have a positive impact in the effectiveness. On the contrary, as M increases, the processing time increases drastically. The efficiency of DUPNIQ is further discussed in the Section 4.2.6.

4.2.5 Synthetic experiments

In the first synthetic experiments, a number of synthetic documents are generated for investigating the effect of OCR errors and the amount of overlapping text between two books. Pairs of texts are created as follows: A clean (without OCR errors) book from the Project Gutenberg website [2] is used as a reference text. The second text

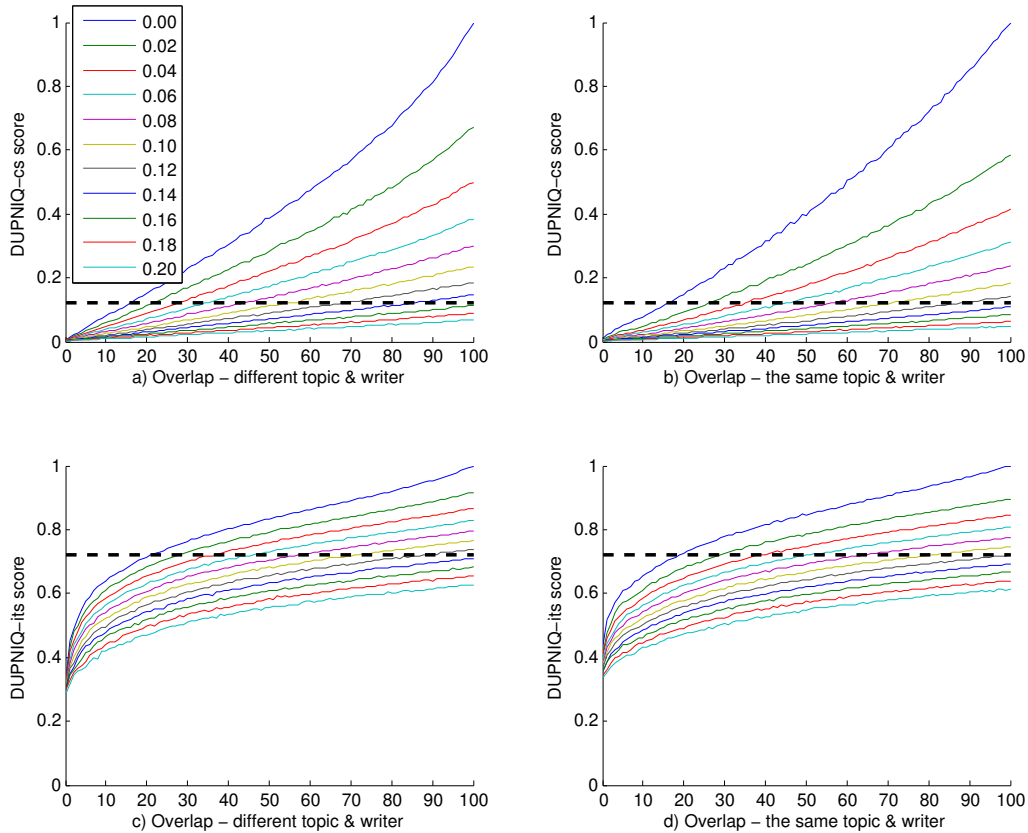


Figure 4.5. DUPNIQ-its and DUPNIQ-cs versus percentage of overlap and character level document noise. Each line in the plot represents results for the given amount of noise. Dashed horizontal lines corresponds to the duplicate detection score thresholds. Note that the word error rate is around 70% for 20% character level noise.

is created by replacing a random portion of the reference text with a sample from another book from the Gutenberg website while ensuring that its length remains the same. A specified amount of character level document noise is added to the second text to simulate OCR errors creating a synthetic document [35]. DUPNIQ-cs and DUPNIQ-its scores are computed between the synthetic and reference book. Experiments are performed 100 times and the scores are averaged. Two different scenarios are investigated: In the first scenario two different books on different topics by different authors are used to generate the reference and the synthetic documents. In the second scenario, the same process is applied using two different novels from the same writer (Joseph A. Altsheler) on the same subject (Civil War). Figure 5.3 shows

the duplication scores as the amount of noise and the percentage of text overlap are varied for both scenarios. The word error rate is estimated with the assumption that an average word in the original text contains 5.45 letters including one white space character. Misrecognized white space characters also causes OCR errors in the form of word merge and splits. It is seen that the duplication score does not significantly depend on which scenario is used. The typical character error rate in scanned book collections is around 2-3% for English, as discussed in Section 3.5. At this level of character error rate, both scoring functions are able to detect duplicate pairs with 10-20% content overlap. As the character error rate increases, Both DUPNIQ-its and DUPNIQ-cs can detect duplicates with higher amounts of content overlap. For a given amount of content overlap, it is clear that DUPNIQ-its is relatively better than DUPNIQ-cs in detecting partial duplicates of texts with higher character error rates. This is indicated by the number of lines above the specified score threshold for a given content overlap ratio.

The second synthetic experiment is designed for investigating the effects of OCR errors on different text representation schemes. For this purpose, two synthetic texts are generated for a book downloaded from the Project Gutenberg website. The original book contains no OCR errors. The two synthetic texts are generated so that they contain the same amount of character level noise but with different random seeds. The document noise model used for generating the synthetic texts are the same as the previous synthetic experiments. All the experiments are repeated 100 times and the results are averaged.

Figure 4.6 shows the ratio of preserved text elements across the two synthetic texts and the original copy. In this particular case, the term “text elements” refer to the unique words, shingles and DCT fingerprints. The preservation ratio corresponds to the total number of preserved text elements across the synthetic texts and the original copy divided by the total number of text elements in the original text. This

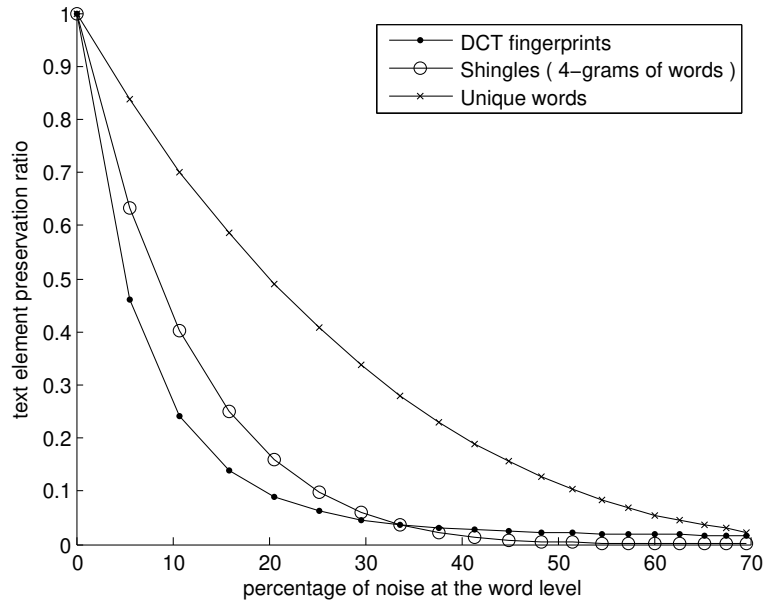


Figure 4.6. The ratio of preserved text elements across the synthetic texts as a function of document noise.

metric indicates the ratio of text elements that remains uncorrupted in the synthetic texts for the purposes of duplicate detection. As the amount of document noise increases, the ratio of preserved elements fall in all cases. In the case of Shingling (4-grams of words without any sampling) and DCT fingerprinting ($p = 4$), the fall is more drastic compared to the unique words. Both Shingling and DCT fingerprinting generate chunks of texts and therefore the chances of corruption due to OCR errors are higher. On the other hand, unique words are unigrams of words and they are better preserved across two noisy texts especially if both texts contain significant amount of OCR errors. In this particular case, the average length of text segments generated by the DCT fingerprinting approach is greater than or equal to four. The reason is that the text segments whose size is smaller than the parameter p are ignored by the technique. That is why the rate of fall as a function of document noise is higher for DCT fingerprinting compared to the Shingling. As the percentage of noise increases, the preservation ratio for DCT becomes slightly higher than the Shingling approach. The reason is that the Discrete Cosine Transformation is used for generating the

fingerprints and it causes collisions more frequently compared to the Shingling. It should be noted that both approaches create 32-bit fingerprints for the text segments and the first 32-bit of MD5 hashcode is used in the case of Shingling.

4.2.6 Efficiency

The unique words are precomputed and stored in binary files for efficiency purposes. Each unique word is represented by a 32-bit hashcode which is generated using a product sum algorithm over the entire text of the string. For batch processing, the sequences of hashcodes are appended one after another in to binary files which are referred to as “barrels”. A barrel containing 2K books occupies 25-35 megabytes of disk space. Alternatively, one could also index unique words and assign a term ID for each unique word. However, it would be a two-pass approach with large memory and computation requirements since the vocabulary of scanned book collections becomes arbitrarily large as the size of the collection grows.

Given a pair of unique word sequences, DUPNIQ aligns them using LCS. The standard dynamic programming algorithm of LCS has quadratic time and space complexity. For long input sequences, the standard LCS algorithm has very large memory requirements. There are also asymptotically faster LCS algorithms for input sequences where no element appears more than once within either input string [49, 26]. These algorithms are shown to be asymptotically faster for aligning very long sequences. Hirschberg’s $O(mn)$ time and linear space LCS algorithm is adopted for our purposes. In our case, we are only interested in the length of LCS. LCS length can be computed efficiently by storing only two rows of the dynamic programming table at a time without computing the LCS itself. This approach is quite efficient if the following efficiency improvements are done prior to alignment.

First of all, it is not necessary to compute LCS over the entire input sequences. One can disregard the words which do not appear in both sequences since a word must

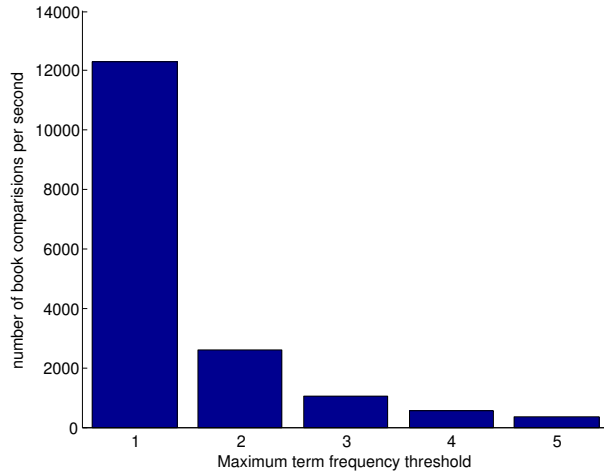


Figure 4.7. The total number of book pairs compared per second by DUPNIQ as a function of maximum term frequency threshold.

appear in both sequences at least once in order to be in the LCS. This elimination procedure reduces the cost of alignment drastically. It should be noted that the intersection of elements between two sequences can be computed in linear time using a hashtable.

Another improvement is to avoid LCS computation entirely when certain constraints apply. The LCS length is upper bounded by the total number of common unique words between the input sequences. Given two input sequences, even if all the common words are assumed to follow exactly the same order, the duplicate score might still be below the duplicate detection threshold. In those cases, there is no need for alignment since the resulting score after LCS computation is guaranteed to be lower than the threshold. In this way, 99% of the LCS alignments are avoided in the case of DUPNIQ. These improvements provide significant speed-up.

Figure 4.7 shows the overall speed of the proposed duplicate detection framework for different values of maximum term frequency threshold M . Using only the unique words for representation purposes ($M = 1$), over 12K book pairs are compared per second on a single core. However, in the case of $M = 5$, only 357 book pairs are compared in a second. The reason is that the total number of words in the representation

increases drastically as M increases. LCS alignment is much slower for longer word sequences. In terms of scalability and effectiveness, the sequence of unique words representation scheme achieves practical bounds for finding partial duplicates in large scanned book collections.

4.3 Mapping duplicated portions of texts

Given a pair of books which are partial duplicates of each other, the aim is to locate and map the duplicate portions of the texts. Our approach is to align the books using the Recursive Text Alignment Scheme introduced in Chapter 3. The duplicate portions of the texts are expected to have a higher number of matching words in the alignment. A binning approach over the alignment output is therefore adopted for visualizing the overlapping content. More specifically, both texts are divided into a number of bins. Each bin contains a fixed amount of words. All the matching words in the alignment are put into the corresponding bins in their respective text. If the ratio of matching words is greater than a predefined threshold, then the bin is colored green. Otherwise the bins are rendered in red. Figure 4.8 shows examples of duplicated portions of books. In these particular examples, each bin has 200 words and the threshold value is set to 50%. The horizontal length of each bar indicates the relative lengths of the books. For these examples the overlap is quite small and the metadata gives very little or no clue about the duplicated text. It should be noted that the duplicated portions may not always follow the same order in both texts. Figure 4.9 shows an example where the order of the chapters is different for two anthologies of Mark Twain’s short stories containing four stories in common. In this case, the alignment algorithm matches the stories which follow the same order in both books. The story “The German Chicago” does not follow the same order in both texts and therefore is not seen in the visualization.

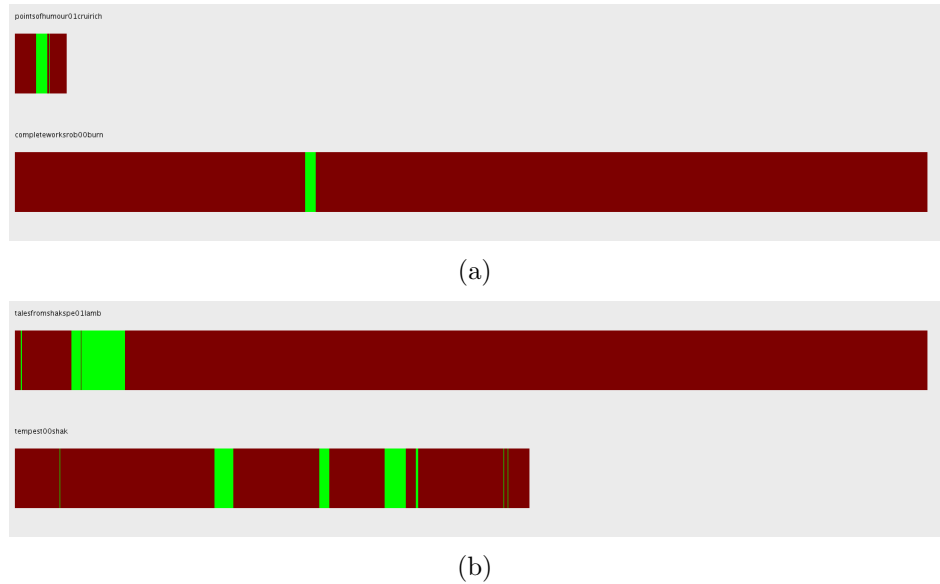
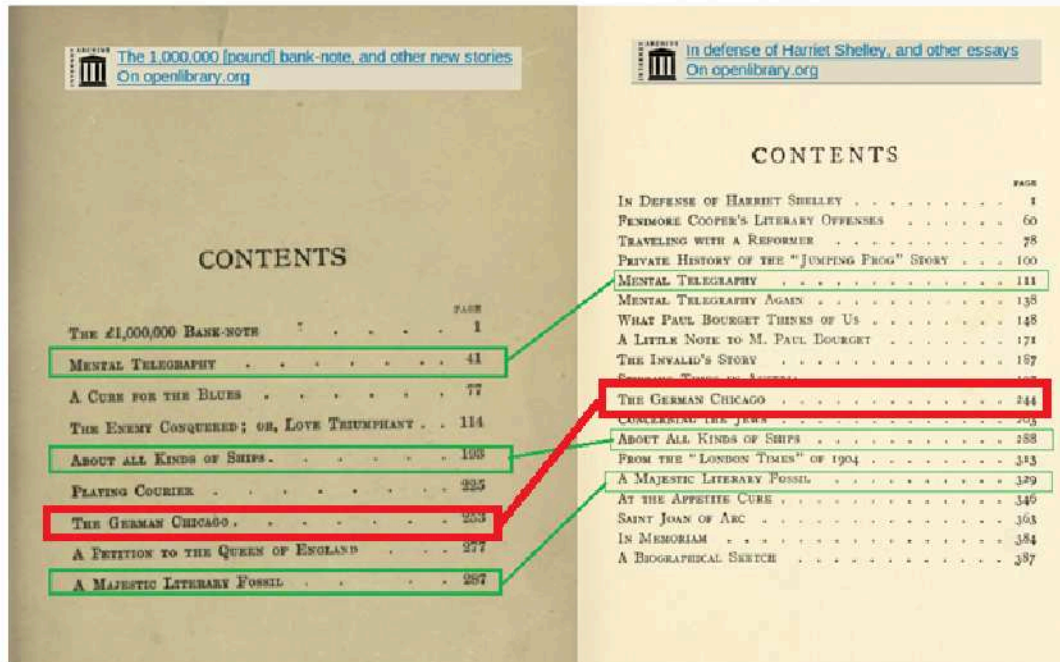
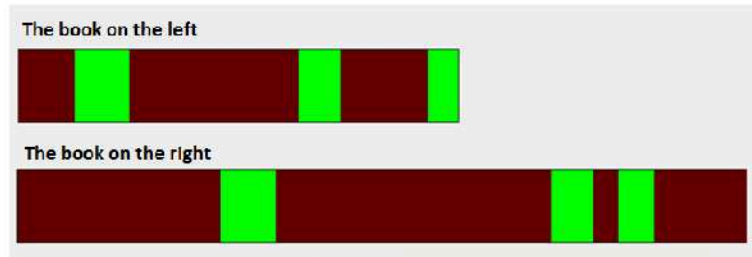


Figure 4.8. Examples for partial duplicates of books. Green bars show matching portions of text with 50% or above word overlap. (a) The book *Points of Humour* (top) contains a selection of verses from the *Complete Works of Robert Burns* (bottom). (b) *Tales from Shakespeare* (top) contains selections from Shakespeare’s plays including the *Tempest* (bottom).

In this chapter the sequence of unique words text representation scheme is proposed for detecting partial duplicates of book in scanned book databases. In the next chapter these concepts are extended for finding translations in scanned book collections.



(a) The index pages for two different books each of which contain a number of short stories.



(b) The overlapping portions of the texts are in green.

Figure 4.9. Mapping duplicated portions of two books which contain a number of stories in common. The story “The German Chicago” is not colored green in the visualization output since it does not follow the same order in both books.

CHAPTER 5

FINDING TRANSLATIONS IN SCANNED BOOK COLLECTIONS

The idea of duplicate detection may be extended to find duplicates across languages, that is translations. Consider Figure 5.1 which illustrates an English verse by Oscar Wilde and its German translation. Unique words are underlined for the two versions of the poem. Unique words in the German version are first transformed into English using a dictionary look-up approach. At this point both sequences are in the same language and the translation detection task is transformed to the mono-lingual duplicate detection problem. As in the case of partial duplicate detection, the resulting word sequence is later aligned with the unique words extracted from the English version. The words in the LCS are indicated with single headed arrows. It is seen that a large number of words follow the same order in both sequences. This is a clear indication for texts being translations.

As for the duplicate detection problem, translation detection is challenging for a number of reasons. Corresponding books may only partially overlap due to edition differences or extra notes and the books may have OCR errors. Another difficulty peculiar to translations is that the dictionary may only provide translations for some of the words. In spite of all these problems we later show that translation detection can be successfully done. Figure 5.2 shows an example of a German book on Johann Wicliff and its English translation. The overlapping content of these two books is automatically determined by the proposed cross-lingual text alignment approach described in the next chapter. The lengths of each row reflect the relative sizes of the two books. Blue denotes aligned portions of texts. The German version contains the

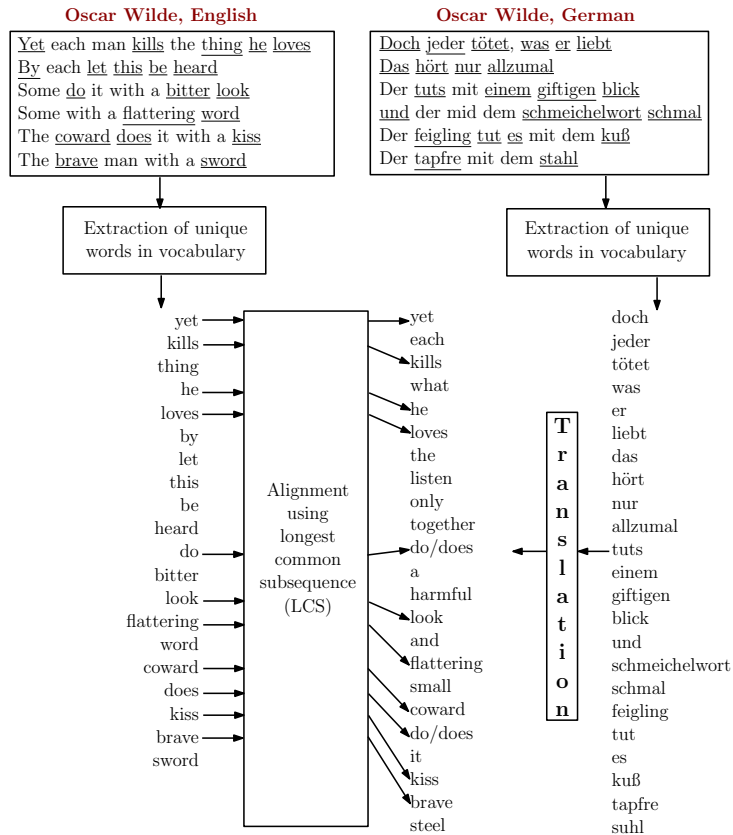
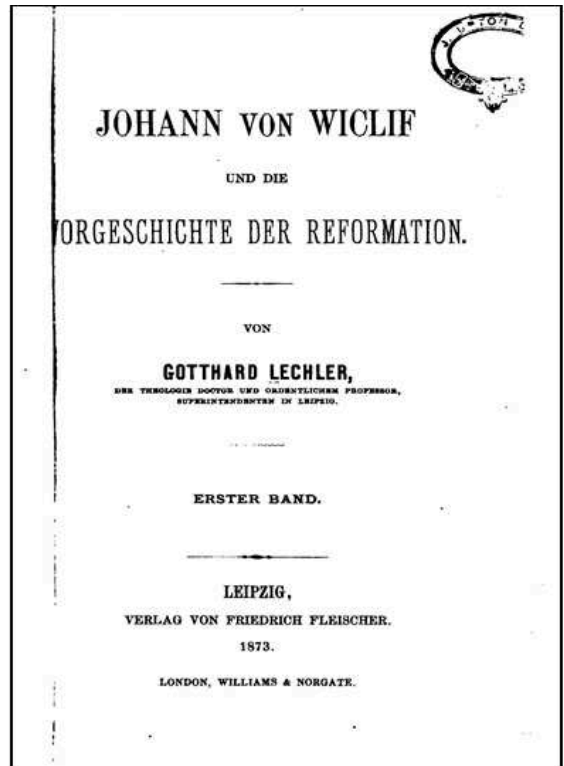
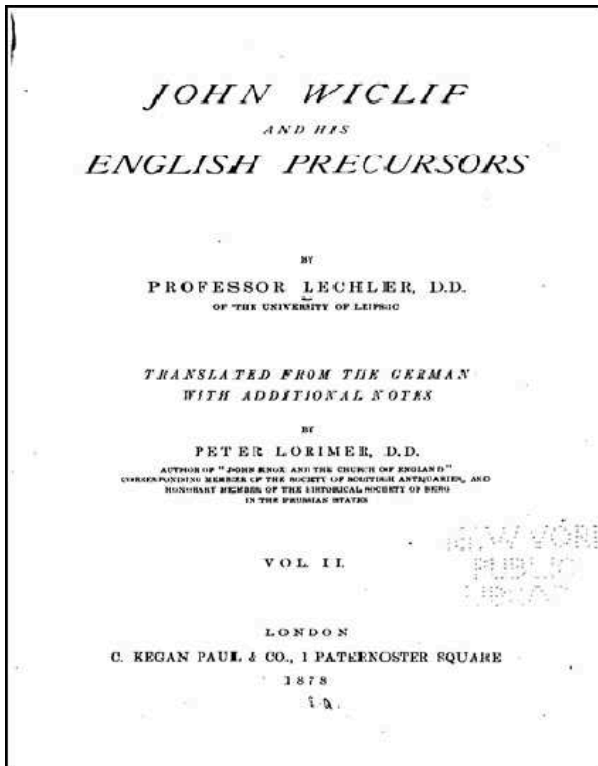


Figure 5.1. Illustration of the proposed translation detection framework.

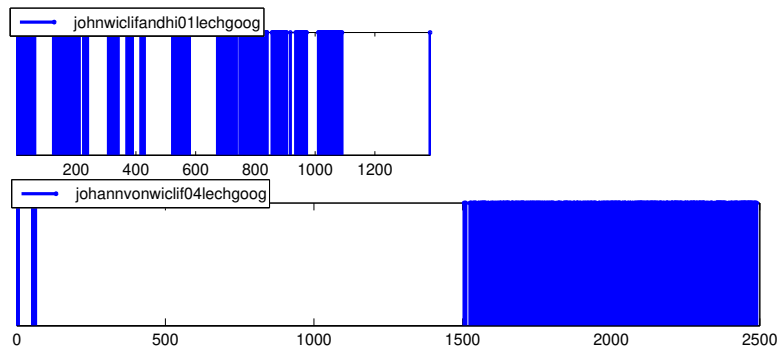
complete text while the English version is only Volume II and hence the big gap in the lower bar. The English version has additional notes and these are reflected in the gaps in the upper bar.

Translation detection has many applications. One can consider translations of books as a parallel corpus and train machine translation models. It is also possible to learn a translation lexicon automatically from translations of texts. Another application would be to transfer annotations of one book to the other translations. One can define several other use cases for a collection which consists of translations of books.

The proposed approach is described next in Section 5.1. The experimental results and evaluations are given in Section 5.2.



(a)



(b)

Figure 5.2. a) Covers of the English translation and the German original of John Wiclif's biography. b) the approximate overlap between the two translations (upper bar English, lower bar German).

5.1 The proposed framework

There are $O(mn)$ distinct book pairs in a bilingual collection which contains n books in one language and m books in some other. In the ideal case, all the book pairs must be checked to see whether they are translations of each other. This is expensive for large n and m especially if the book comparisons take a significant amount of time. In the framework proposed here, the book comparison time is minimized using the sequence of unique words representation introduced in Chapters 3 and 4. Each book in the collection is represented by the sequence of its unique words. At this point, the books are not comparable since the sequence of unique words contain words in two different languages. The solution is to convert each unique word in one language to the other using a look-up dictionary. Each word is mapped to its possible translations and a transformed representation is used for comparing books across languages. There are also words such as names and places which may be preserved across languages. The unique words which appear in both sequences but not translated by the dictionary are also incorporated in to the transformed sequence. The transformed word sequence and the sequence of unique words of the other book are compared using LCS. If there are a significant number of words in the LCS, the books are very likely to contain translated text. The problem is that the LCS length depends on the length of the books to be aligned. Therefore the two score normalization schemes proposed in Chapter 4 are adopted. In this context, TRANSNIQ-its and TRANSNIQ-cs scores refer to the *its* and *cs* scoring schemes in Section 4.1.2.

Unlike the example in Figure 5.1, books are much longer texts and may include hundreds of thousands of words. Detailed statistics are shown in Table 5.1 for three example pairs of books ¹. Egmont and Faust are different books written by J. W.

¹Goethe's Egmont in English: <http://www.gutenberg.org/cache/epub/1945/pg1945.txt>,
Goethe's Egmont in German: <http://www.gutenberg.org/cache/epub/2146/pg2146.txt>,
Goethe's Faust in English: <http://www.gutenberg.org/files/14591/14591.txt>,
Kant's the Critique for Pure Reason: <http://www.gutenberg.org/files/4280/4280.txt>

Table 5.1. Detailed statistics for three pairs of books compared using the proposed translation detection framework.

English Book X	German Book Y	#words						TRANSNIQ	
		in X	$ X $	$ Y $	$ X \cap Y $	$ X_T \cap Y $	$ LCS $	its	cs
Egmont	Egmont	29177	2395	3224	32	492	251	0.643	0.090
Faust	Egmont	37131	3706	3224	27	416	43	0.426	0.012
Critique	Egmont	209383	2625	3224	6	270	31	0.396	0.011

Goethe. The other book is “The Critique of Pure Reason” by Immanuel Kant. The length for each book is given for the English version of these three books. The German version of Egmont has 25,743 words in total. $|X|$ and $|Y|$ corresponds to the number of unique words in books X and Y respectively. Approximately 10% of the words are unique for the English versions of the books Egmont and Faust since they are relatively short books. However, the book “The Critique of Pure Reason” is much longer and less than 1% of the words are unique in the text. It should be noted that the ratio of unique words decreases as the length of the text increases, as discussed in Section 3.1.1. $|X \cap Y|$ corresponds to the number of common unique words between X and Y without any translation. $|X_T \cap Y|$ refers to the number of common words after translating the words in X to the language of book Y. $|LCS|$ refers to the LCS length. The first book pair is a translation pair whereas the other two are not. It is clear that the translation pair has a significant number of words in the LCS compared to the non-translation pairs of books. The thresholds for TRANSNIQ-its and TRANSNIQ-cs scores are 0.49 and 0.023. Table 5.1 shows that the translation of Egmont is correctly detected while different books by the same author or different authors are not confused as translations.

Table 5.2 shows statistics on the size of the dictionaries used in our experiments [3]. All the dictionaries provide translations for different forms of the word (such as plural, gerund, past participle etc.), whereas the English-German 5K and English-Finnish dictionaries lack this feature. We also provide the average translation success as a percentage for each dictionary. These statistics are generated for the EUROPARL

Table 5.2. Dictionary statistics after ignoring phrasal translations.

Dictionary	Words	Translation success		
		Unique words	Vocabulary words	Entire document
English-German 62K	62242	79.8%	85.5%	93.8%
English-German 5K	5487	19.7%	27.1%	56.0%
English-Finnish	2997	11.9%	26.8%	52.1%
English-French	17326	54.2%	57.6%	79.7%
English-Spanish	23377	53.2%	57.2%	83.5%

corpus. We also tried a number of lemmatization techniques in order to improve translation success. Even if we observed improvements in the total number of translated words, no improvement is observed in the precision and recall figures for translation detection. Dictionary size and OCR error rate are the determinants of the overall success of the dictionary based alignment approach.

5.2 Experiments

The effectiveness of the proposed translation detection framework is tested for several scanned book collections of varying size. Another set of experiments are performed for four language pairs on a public dataset called “EUROPARL” which contains noise free (i.e., no OCR errors) government documents. A number of synthetic experiments are also carried out to investigate the impact of document noise and content overlap in translation detection. Three evaluation metrics are devised for different search scenarios. The proposed method is compared to three baselines one of which uses metadata for finding translations of books. The details are elaborated in the following subsections.

5.2.1 Datasets

Books downloaded from the Internet Archive (IA) [1] were used to construct the scanned book datasets. English-German training and the 2K datasets are publicly available ².

An English-German training set containing 30 scanned books (16 English, 14 German) from the IA database. It is manually verified that a book has at least one translation in the set. There are 31 true translation pairs in total. This set is used to estimate the translational similarity threshold for the scanned book experiments.

The 2K dataset is an English-German collection of 2K scanned books and is one of the datasets used by Krstovski and Smith in [53]. There are 18 translation pairs in this dataset. The dataset is originally created by downloading a random selection of 1K German and 1K English books from the IA website and embedding 17 book translation pairs in it. However, our approach discovered that there are actually 18 translation book pairs in the dataset. TRANS found three additional translation pairs and rejected two translation pairs which were initially in the ground truth. After manual investigation, the ground truth for this dataset has been corrected and it is used for the experiments along with the updated results obtained from Krstovski & Smith.

The 50K dataset is a collection of 50K books in German randomly selected from the IA database. Using the language identifier [123], it is verified that the OCR outputs are not garbage and that the dominant language of these texts is German. This set is used only for ranking experiments. A set of 20 famous books in English is used for querying. Query books are chosen in a way that there exists at least one translation for each of them in the entire collection. The ground truth for the query set is obtained as follows: for each query book, books in the 50K collection are ranked

²<http://ciir.cs.umass.edu/downloads/trans-detect/> and <http://books.cs.umass.edu/downloads/trans-detect/>

according to the TRANSNIQ-cs, TRANSNIQ-its and metadata scores. Each of these techniques produces a ranked list for each query. The top 200 ranking entries from all three lists were pooled for each query and then manually judged. This pooling approach provide a basis to determine the relative effectiveness of the systems being compared. In total, 52 translation pairs were labeled for all 20 queries.

The EUROPARL parallel corpus is a standard collection of text documents from the proceedings of the European Parliament [50] used for machine translation. These documents are clean - since they have no OCR errors. Version 3 is used for our experiments in order to compare the results with the baseline approach described in [53]. It contains speeches from the period 04/1996-10/2006. There are over 600 documents each of which is translated in to 11 languages. Unlike the scanned book collections, these texts do not include any document noise since they are translated and typed by humans. We use four language-pairs: English-Finnish, English-French, English-German and English-Spanish. Notice that Finnish is a member of the “Uralic” group of languages whereas English, French and Spanish belong to the Indo-European language group. The average number of words per document in the English collection is 50360 after removing the tags. Many of these documents are much shorter than most books.

5.2.2 Evaluation metrics

Three different evaluation methods are defined to elucidate different aspects of the problem and also depending on what kind of ground truth is available. For large datasets, it is not possible to obtain manually labeled ground truth. In such cases, a retrieval approach must be adopted as described below.

Retrieval of Translations: In this approach, each book in the source language (English in our example) is regarded as a query and all the books written in the target language (German) are ranked according to their translational similarity score.

MAP (Mean Average Precision) is calculated over the rank lists and this score serves as a metric for the retrieval of translations task. The retrieval approach is feasible especially for large datasets since the evaluation is practical. One can adopt a pooling approach in analogy with the traditional IR ranking paradigm to obtain relevance judgments. The details are described in the experimental section.

Ranking All Book Pairs: Krstovski and Smith [53] rank all the book pairs in a single list according to some similarity score and compute Average Precision (AP) over the entire ranked list. This is different than the retrieval of translations approach. Consider the following list of English books E1, E2, E3 and German books G1, G2. Assume that the following ranked list is produced after comparing all the source-target book pairs (E3G1, E1G2, E2G2, E1G1, E3G2, E2G1). The retrieval of translations approach instead use E1, E2 and E3 as queries and compute the AP for each ranked list (E1G2, E1G1), (E2G2, E2G1) and (E3G1, E3G2) and average all the AP values to compute a MAP score. The ranking all book pairs approach is reasonable as long as the ground truth for the entire dataset is available. One may still go over the entire ranked list and annotate each pair manually. However, this is not feasible for large datasets since the number of book pairs to be checked is significantly larger than for the retrieval approach.

Binary Classification: This measure requires the system to classify each book pair as a translation or not. In this context this is done using a threshold over the translation scores. If the ground truth is available for the entire dataset, then precision and recall values can be generated.

From now on, references to MAP and AP will be taken to imply the retrieval of translations and ranking all book pairs experiments respectively. It should be noted that binary classification is the most restrictive metric since it requires translational scores to be comparable between different book pairs and a careful selection of the score threshold. Even if the other two evaluation metrics MAP and AP are both

equal to 1.0, binary classification may have precision and recall values below 1.0. It happens when the score threshold is either too high or too low. The least restrictive evaluation metric is the MAP score for the retrieval task since it does not require the translational scores to be comparable between different queries.

5.2.3 Baselines

Most work on creating parallel corpora has been focused on small datasets and using either structural information or the alignment of individual sentences [102] with two exceptions: Uszkoreit et al. [108] and Krstovski & Smith [53]. Uszkoreit’s approach is not used as a baseline since the datasets and the translation system they used are not available to us. Here we use three baseline systems: metadata search, IBM MODEL 1 and where available numbers from Krstovski and Smith [53].

META refers to using metadata search for finding translation pairs in a collection of books. Here we use the title and author information from the IA database as follows: first all the punctuations in the author and title fields are removed and all the characters are lowercased. Numeric characters are also ignored only for the author field since the date information leads to false matches. The title of the query book is also translated from English to German using the Google Translate API. The set of tokens in the author field of the query book is compared against the books in the collection of 50K German books using Jaccard similarity. If the similarity is greater than zero, then the translated title is also compared against the title of each candidate book in the same way. The “metadata score” for a single pair of books is defined to be the average of the title and author Jaccard similarities. The metadata score is used to detect/rank books pairs for being translations. Notice that the metadata may not be fully reliable since it is manually entered.

IBM M1 refers to the widely-used IBM Model 1 used for aligning words given two sentences in different languages [19]. It is used for different tasks over parallel

corpora and essentially gives an estimate for the probability of a target sentence T in some language given a source sentence S in another language. There are several simplifying assumptions in this model. It does not incorporate any information about the long range order of words in the source and target sentences unlike the sequence of unique words. This approach is therefore ideal to demonstrate the effectiveness of bag-of-words models over long texts. Since this model is effective for ranking, we use it only for retrieval and ranking experiments. For fairness, the same dictionary is used for all techniques. Transition probabilities are estimated by assuming that all translations are equiprobable.

Krstovski & Smith use an approach for generating a ranked list of book translation pairs without the use of a bilingual dictionary or a machine translation system [53]. Each book in the collection is represented in the vector space and cosine similarity is used to rank all the book pairs in the collection. The vector representation only accounts for the words which appear in both languages without any translation. For each book, the weights of the vector representation are calculated by multiplying the frequency of the term in the book with the inverse document frequency of the term in the collection of books in the same language, i.e. (TFxIDF). The Locality sensitive hashing (LSH) approximation algorithm is used to calculate cosine similarity to reduce the time complexity. We use their datasets and results which are publicly available.

5.2.4 EUROPARL experiments

The EUROPARL dataset is used to test the effectiveness of our approach for documents with no OCR errors. There are roughly 650 documents per language each of which has a translation in the other language. For each language pair we selected 50 translation pairs at random as a training set and used the remaining as a test set. The training set is used only for training the score threshold (a different threshold

Table 5.3. Translation retrieval and ranking all-pairs experimental results for the EUROPARL dataset using the proposed approach with dictionary transformation.

Dataset	Retrieval Experiments (MAP)		All-Pairs Experiments (AP)		
	TRANS-its	TRANS-cs	TRANS-its	TRANS-cs	Krs.&Smith
Eng-Fin	1.0	1.0	1.0	1.0	-
Eng-Fre	1.0	1.0	1.0	1.0	-
Eng-Ger	1.0	1.0	1.0	0.994	0.986
Eng-Spa	1.0	1.0	1.0	1.0	-

Table 5.4. Classification experiments for the EUROPARL dataset using the proposed approach with dictionary transformation.

Dataset	TRANSNIQ-its			TRANSNIQ-cs		
	precision	recall	F-measure	precision	recall	F-measure
Eng-Fin	1.0	0.998	0.999	0.973	1.0	0.986
Eng-Fre	1.0	1.0	1.0	1.0	1.0	1.0
Eng-Ger	1.0	0.997	0.998	0.992	0.995	0.993
Eng-Spa	1.0	1.0	1.0	0.992	1.0	0.996

for each language since dictionary sizes vary significantly). For English-German, the 62K dictionary is used. The evaluations are done on the test set. The retrieval and ranking all pairs experiments are shown in Table 5.3. Binary classification results are given in Table 5.4. We notice that TRANSNIQ-its has a MAP score of 1.0 and an AP of 1.0 for both the retrieval and ranking of all pairs evaluations. TRANSNIQ-cs performs slightly worse on the English-German ranking of all pairs evaluation. We also list Krstovski and Smith’s result for the English-German pair from their paper (their splits are different but the results are indicative). Krstovski and Smith do not provide numbers for the other language pairs but they have graphs which clearly show that the AP score must be less than 1.0.

The binary classification experiments indicate that threshold selection is a hard problem compared to the ranking and retrieval paradigms. TRANSNIQ-its has a precision of 1.0 for all language pairs. TRANSNIQ-its also ranks all the document pairs perfectly since the AP score is 1.0. However, the recall values are slightly lower than 1.0 for English-Finnish and English-German datasets. The reason is that one

pair in the English-Finnish and two pairs in the English-German dataset are below the score threshold although they are relevant. Further analysis of the results show that missing document pairs are actually very short (a few hundred words). Our technique is quite robust for longer documents. Precision and recall values for TRANSNIQ-cs are both lower than 1.0 for the English-German dataset, which indicates there are relevant documents below the score threshold while there are false positives with a score higher than the threshold. Clearly TRANSNIQ-its performs very well on all metrics followed by TRANSNIQ-cs.

The impact of term selection based on frequency and the dictionary based word sequence transformation approaches have been investigated. As in the case of DUP-NIQ, the “maximum term frequency threshold” (M) is introduced for sampling words from the text. All the terms whose frequency is equal to M or below are used to represent the text *without* the dictionary transformation. The sequence of sampled words are directly aligned using LCS and translational similarity scores are computed as in the case of TRANSNIQ. The experiments are repeated for four language pairs and different values of the maximum term frequency threshold. The value “ALL” corresponds to the case where all the terms in the text are used regardless of the number of appearances in the text. In this particular brute-force scenario, the Recursive Text Alignment Scheme (Section 3) is used for alignment purposes. The top 500 stopwords in the language are removed prior to alignment for further efficiency. The stopwords for each language are trained using 15 noise-free books downloaded from the Project Gutenberg website. It should be noted that the holistic text alignment approach without any dictionary transformation relies on the fact that there exist some common words whose meaning and spelling are the same across translations. However, this might not always be true for some language pairs such as English-Chinese or English-Arabic [101]. The advantage of the proposed approach over this brute force

alignment approach is that it uses an external source of linguistic information in the form of a dictionary.

The results given in Tables 5.5 and 5.6 indicate that the proposed approach without the dictionary transformation performs reasonably well for the EUROPARL collection (1:1 content overlap and no document noise). The MAP and AP scores for the translation retrieval and all-pairs ranking experiments are quite close to 1.0 except for the English-Finnish set (AP = 0.973) where the entire texts are used for the alignment ($M = ALL$). However, for the binary classification task, precision and recall scores are significantly lower compared to TRANSNIQ-its with dictionary transformation using the sequence of unique words (See Tables 5.3 and 5.4). Increasing the maximum term frequency threshold does not necessarily improve the effectiveness of the proposed approach. Instead, the LCS alignment gets prohibitively slow due to the size of the word sequences to be aligned. The global alignment approach without any term sampling (i.e., $M = ALL$) is therefore not practical. In the case of the EUROPARL dataset, the experimental results are consistent across the four language pairs. Further analysis of the alignment results indicate that EUROPARL government documents contain a large number of terms such as names of people, events, concepts and places which are preserved exactly across translations. This type of evidence might not always be available for finding translations in other document collections. TRANSNIQ-cs slightly outperforms the TRANSNIQ-its score normalization scheme in most cases, especially for the retrieval and ranking all-pairs experiments.

The global word sequence information across document translations provides additional evidence for finding translations. In the case of the EUROPARL collection, the global alignment approach improves upon the bag-of-words approach employed by Krstovski and Smith [53] for finding translations. Effectively, Krstovski and Smith also use cognate words which are spelled exactly the same across translations. However, they adopt a bag-of-words approach to represent the texts in the vector space.

Table 5.5. Precision (P), recall (R) and F-measure (F) scores of TRANSNIQ-its and TRANSNIQ-cs for the EUROPARL dataset **without the dictionary transformation**. M and T correspond to the maximum term frequency and translational similarity score thresholds, respectively.

Language pair	M	TRANSNIQ-cs				TRANSNIQ-its			
		T	P	R	F	T	P	R	F
eng-fin	1	0.005	0.896	0.995	0.943	0.28	0.851	0.995	0.917
eng-fin	2	0.005	0.967	0.989	0.978	0.31	0.911	0.995	0.951
eng-fin	3	0.005	0.982	0.986	0.984	0.315	0.869	0.996	0.928
eng-fin	5	0.005	0.985	0.968	0.976	0.34	0.965	0.991	0.978
eng-fin	10	0.005	0.988	0.88	0.931	0.35	0.986	0.986	0.986
eng-fin	ALL	0.005	0.984	0.774	0.866	0.37	0.786	0.973	0.870
eng-fre	1	0.02	1	0.998	0.999	0.345	0.851	0.998	0.919
eng-fre	2	0.015	0.997	1	0.998	0.38	0.932	0.998	0.964
eng-fre	3	0.015	0.995	1	0.997	0.4	0.979	0.998	0.988
eng-fre	5	0.015	1	1	1	0.46	0.997	0.997	0.997
eng-fre	10	0.01	0.977	1	0.988	0.46	0.977	0.997	0.987
eng-fre	ALL	0.01	0.933	1	0.965	0.48	0.998	0.998	0.998
eng-ger	1	0.01	0.992	0.993	0.992	0.32	0.903	0.997	0.948
eng-ger	2	0.01	0.993	0.993	0.993	0.34	0.87	0.997	0.929
eng-ger	3	0.01	0.993	0.993	0.993	0.345	0.74	0.997	0.849
eng-ger	5	0.01	0.998	0.993	0.995	0.38	0.985	0.997	0.991
eng-ger	10	0.01	0.998	0.992	0.995	0.405	0.966	0.995	0.980
eng-ger	ALL	0.005	0.637	1	0.778	0.42	0.924	0.993	0.957
eng-spa	1	0.01	0.94	1	0.969	0.33	0.938	1	0.968
eng-spa	2	0.01	0.968	1	0.984	0.355	0.954	1	0.976
eng-spa	3	0.01	0.97	1	0.985	0.365	0.944	0.998	0.970
eng-spa	5	0.01	0.993	1	0.996	0.38	0.953	0.998	0.975
eng-spa	10	0.01	0.993	0.997	0.995	0.42	0.993	0.997	0.995
eng-spa	ALL	0.01	1	0.972	0.986	0.44	0.985	0.993	0.989

Table 5.6. Ranking all-pairs (Average Precision - AP) and translation retrieval (Mean Average Precision - MAP) results for TRANSNIQ-its and TRANSNIQ-cs on the EUROPARL dataset **without the dictionary transformation**. M corresponds to the maximum term frequency threshold.

Language pair	M	TRANSNIQ-cs		TRANSNIQ-its	
		AP	MAP	AP	MAP
eng-fin	1	0.995	1	0.996	1
eng-fin	2	0.996	1	0.996	0.999
eng-fin	3	0.996	1	0.997	1
eng-fin	5	0.996	1	0.996	1
eng-fin	10	0.988	1	0.996	0.999
eng-fin	ALL	0.978	0.999	0.973	0.998
eng-fre	1	1	1	1	1
eng-fre	2	1	1	0.999	1
eng-fre	3	1	1	0.998	1
eng-fre	5	1	1	0.998	1
eng-fre	10	1	1	0.998	1
eng-fre	ALL	1	1	0.998	1
eng-ger	1	1	1	0.997	0.998
eng-ger	2	1	1	0.997	0.999
eng-ger	3	1	1	0.998	0.999
eng-ger	5	1	1	0.997	0.999
eng-ger	10	1	1	0.998	0.998
eng-ger	ALL	0.999	1	0.995	0.998
eng-spa	1	1	1	1	1
eng-spa	2	1	1	1	1
eng-spa	3	1	1	1	1
eng-spa	5	1	1	0.999	1
eng-spa	10	1	1	0.998	1
eng-spa	ALL	1	1	0.998	0.998

Their approach does not exploit the word sequence information for finding translations. The English-German EUROPARL experiments indicate that aligning the entire texts of documents without any translation or word sequence transformation ($M = ALL$) gives an AP score of 0.995 on the test collection. Krstovski and Smith’s average AP results (0.984) are reported across ten random splits for the same language pair. Despite this difference, the results are indicative that the global sequence information provides additional information for finding translations. The use of global word sequence information is further discussed for scanned book collections in the next section.

5.2.5 Experiments with real scanned books

Table 5.7 shows results for the retrieval and all pairs experiments on real scanned books for the English-German datasets. The best scores are shown in bold face. In all the tables below “Dict” refers to the size of the dictionary. TRANSNIQ-its (using the large dictionary) is the most successful system among all others in providing the highest scores. Note that the results are worse when the smaller dictionary is used. Metadata search (META) performs well in ranking books for both the training and 2K test sets, but not as well for retrieving translations on the 50K dataset (MAP = 0.821). TRANSNIQ-cs is much worse indicating the importance of LCS length normalization. In all cases, IBM-M1 performs poorly. The all pairs experiment is not performed on the 50K dataset because it would require judging several thousand entries. Krstovski and Smith obtained an AP = 0.945 for the 2K dataset (their precision, recall and MAP results are not available for the 2K dataset).

Binary classification is performed by learning the score threshold from the training set and it is used for the 2K dataset. As seen in Table 5.8, TRANSNIQ-its with the 62K dictionary gives perfect precision and recall values for both datasets. TRANSNIQ-cs and TRANSNIQ-its both provide perfect scores on the training set

Table 5.7. Translation retrieval and ranking all-pairs experimental results for the scanned book datasets.

		Retrieval Experiments (MAP)			All-Pairs Experiments (AP)	
Method	Dictionary	Training	2K	50K	Training	2K
TRANSNIQ-its	62K	1	1	1	1	1
TRANSNIQ-cs	62K	1	1	0.717	1	1
TRANSNIQ-its	5K	1	1	0.714	1	1
TRANSNIQ-cs	5K	1	1	0.669	1	0.943
META	-	0.99	1	0.821	0.959	0.916
IBM-M1	62K	0.302	0.008	<0.001	0.148	<0.001
Krs.&Smith	-	-	-	-	-	0.945

Table 5.8. Binary classification results on English-German datasets using the sequence of unique words representation with dictionary transformation. “T”, “P”, “R” are threshold, precision, recall and F-measure scores respectively.

Approach	Dict. Size	T	Training Set			2K Set		
			P	R	F	P	R	F
TRANSNIQ-its	62K	0.49	1	1	1	1	1	1
TRANSNIQ-cs	62K	0.023	1	1	1	0.782	1	0.877
TRANSNIQ-its	5K	0.395	1	1	1	0.122	1	0.201
TRANSNIQ-cs	5K	0.0085	1	1	1	0.01	1	0.019
META	-	0.275	0.882	0.968	0.923	0.739	0.944	0.829

even with a small dictionary. Precision values for the 2K dataset fall if the small dictionary is used. The drastic fall in the precision figures for the 2K dataset is due to the low score threshold. This indicates that there is a need for a better threshold selection paradigm since both score functions actually perform well in the all pairs ranking experiment as shown in Table 5.7. Surprisingly metadata search does not provide high detection scores (precision = 0.739, recall = 0.944) for either the 2K set or the train set.

Uszkoreit et al.’s [108] best published result (using an oracle to choose the threshold) for a dataset of 103 books (English-French) with 30 matching pairs has a precision of 1.0 and a recall of 0.71. Although it is not directly comparable, we note that TRANSNIQ-its has both precision and recall 1.0 on a 2K book dataset. The propose framework also does not require the complete translation of books. Unfortunately,

their machine translation system and datasets are not publicly available to us to be able to make a direct comparison.

The impact of the dictionary transformation approach has also been investigated for scanned book collections. The results are given in Tables 5.9 and 5.10. It is clear that, on the training set, the proposed approach without the dictionary transformation provides the highest scores on all tasks (1.0) for different values of M except for TRANSNIQ-its where $M = 3$ and $M = 5$. However, the translational similarity score learned from the training set does not generalize for the 2K test set. The precision scores are below 0.01% for $M \leq 10$ although the recall values is quite high. Despite the low precision scores, over 98% of book pairs are successfully eliminated for being a match after applying the translational similarity threshold. This approach might therefore serve as a filter for a more exhaustive translation detection approach. The ranking all-pairs and translation retrieval experiments also indicate that the effectiveness of the proposed approach has been severely degraded without the dictionary transformation. The corresponding AP and MAP values are much lower than 1.0. It should be noted that TRANSNIQ using the dictionary transformation approach (where $M = 1$) provides the highest scores (1.0) for all evaluation metrics for the 2K dataset (See Tables 5.8 and 5.7).

The translation retrieval experiments on the scanned book datasets also indicate the importance of global word sequence information for finding translations. Krstovski and Smith’s bag-of-words approach provides an AP score of 0.945 for the 2K test set. The proposed approach without the dictionary transformation where $M = ALL$ provides a higher score of 1.0. In this particular case, it should be noted that both approaches only use the words which are preserved exactly across document translations without any external source of linguistic information. The global word sequence information provides additional evidence for finding translations.

Table 5.9. Precision (P), recall (R) and F-measure (F) scores of the proposed approach (TRANSNIQ-its and TRANSNIQ-cs) **without the dictionary transformation** for the Training and 2K sets. M and T correspond to the maximum term frequency and translational similarity score thresholds respectively.

Approach	M	T	Training Set			2K Set		
			P	R	F	P	R	F
TRANSNIQ-its	1	0.33	1	1	1	0.002	1	0.003
TRANSNIQ-its	2	0.36	1	1	1	0.001	1	0.003
TRANSNIQ-its	3	0.37	0.969	1	0.984	0.001	0.944	0.002
TRANSNIQ-its	5	0.39	0.969	1	0.984	0.001	0.889	0.002
TRANSNIQ-its	10	0.41	1	1	1	0.001	1	0.001
TRANSNIQ-its	ALL	0.45	1	1	1	0.002	1	0.004
TRANSNIQ-cs	1	0.005	1	1	1	0.003	1	0.006
TRANSNIQ-cs	2	0.005	1	1	1	0.003	1	0.005
TRANSNIQ-cs	3	0.005	1	1	1	0.002	1	0.005
TRANSNIQ-cs	5	0.005	1	1	1	0.002	1	0.004
TRANSNIQ-cs	10	0.005	1	1	1	0.002	1	0.003
TRANSNIQ-cs	ALL	0.01	1	1	1	0.225	1	0.367

Table 5.10. Ranking all-pairs (Average Precision - AP) and translation retrieval (Mean Average Precision - MAP) results for TRANSNIQ-its and TRANSNIQ-cs on the scanned book datasets **without the dictionary transformation**. M corresponds to the maximum term frequency threshold.

Approach	M	Training Set		2K Set	
		AP	MAP	AP	MAP
TRANSNIQ-its	1	1	1	0.479	0.771
TRANSNIQ-its	2	1	1	0.636	0.831
TRANSNIQ-its	3	1	1	0.504	0.710
TRANSNIQ-its	5	1	1	0.480	0.707
TRANSNIQ-its	10	1	1	0.440	0.610
TRANSNIQ-its	ALL	1	1	1	1
TRANSNIQ-cs	1	1	1	0.452	0.76
TRANSNIQ-cs	2	1	1	0.458	0.889
TRANSNIQ-cs	3	1	1	0.452	0.732
TRANSNIQ-cs	5	1	1	0.450	0.767
TRANSNIQ-cs	10	1	1	0.440	0.651
TRANSNIQ-cs	ALL	1	1	1	1

The effectiveness of the proposed approach without the dictionary transformation is limited for scanned books collections unlike the EUROPARL dataset. As discussed earlier, the EUROPARL dataset does not contain any document noise in the form of OCR errors and there is 1:1 content overlap between the translations. In addition, the parliamentary documents contains a number of speeches including a large number of proper names, places and concepts which are spelled exactly the same way across translations in order to avoid translational ambiguity. These words serve as a strong evidence for matching document translations. In the case of scanned book collections, however, the OCR errors, partial content overlap and additional ambiguity due to the translation of literature works make the problem much more challenging. The proposed approach is still able to cope with these challenges using the dictionary transformation approach.

5.2.6 Synthetic Experiments

The effect of OCR errors on translation detection is investigated by generating synthetic errors in texts. A pair of texts is created as follows: Two error-free (no OCR errors) books are downloaded from the Project Gutenberg website [2] - one in the source language (the reference text) and a second in the target language. The latter is used for generating synthetic texts by adding a specified amount of random character level document noise to simulate OCR errors. Unique words in the reference text are translated in to the target language. TRANSNIQ-its and TRANSNIQ-cs scores are computed for the reference and synthetic texts using different amounts of character level document noise from 0% to 20% with 1% increments. Experiments are repeated one hundred times - each time with different random seeds - and the scores are averaged.

The noise model introduced in [35] is adopted for generating the synthetic texts. The model basically performs string edit operations (insertion, deletion and replace-

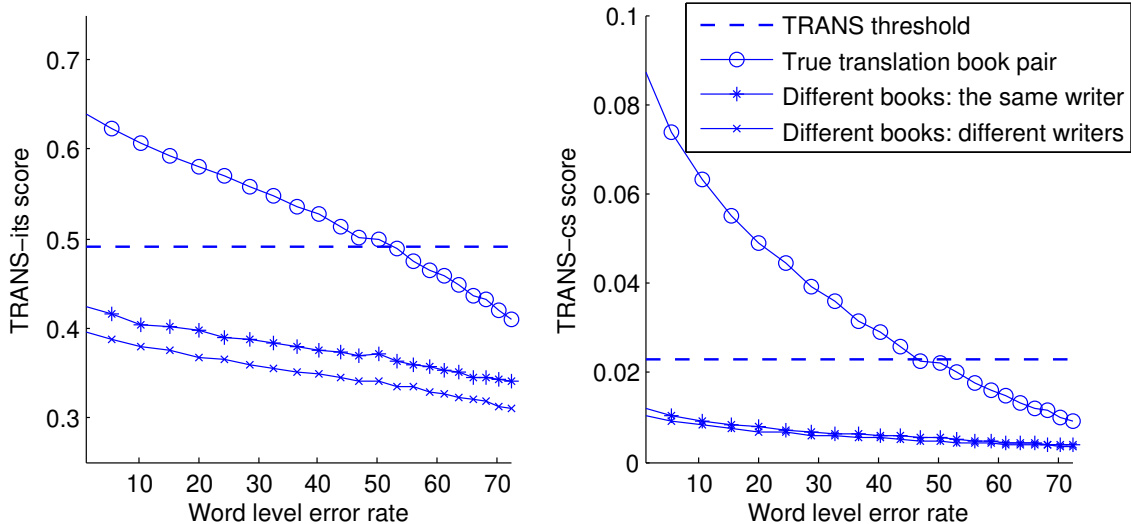


Figure 5.3. The effect of OCR errors on the translation scores are investigated for three different scenarios. TRANSNIQ-its (left) and TRANSNIQ-cs (right) scores are shown as a function of word level synthetic document noise. Both measures are able to classify the book pairs correctly for the given thresholds even for high rates of character level document noise.

ment) over the entire text for the given amount for each type of noise. The total amount of noise is defined to be the total percentage of characters deleted, replaced and inserted over the entire string. The distribution of edit operations is defined to be uniform, i.e., $[1/3, 1/3, 1/3]$ respectively. Case is folded and all punctuations and numerals are removed. The English-German dictionary used in the synthetic experiments contains 62K words including inflections.

Three different scenarios are investigated. In the first scenario, we evaluate the effect of OCR errors for true translation pairs. In this case, the reference book is chosen to be “Egmont” which is written in German by Johann Wolfgang von Goethe and synthetic texts are generated using the English translation of the same book. In the second scenario, the same process is applied to two different books which are known not to be translations of each other but written by the same author - the German original of Goethe’s “Egmont” and an English translation of ‘Goethe’s “Faust”. The purpose of this scenario is to test the robustness of the proposed method

Table 5.11. Detailed statistics for the three pairs of books examined in Figure 5.3. $|X|$ and $|Y|$ corresponds to the number of unique words in books X and Y respectively. $|X \cap Y|$ corresponds to the number of common words between $|X|$ and $|Y|$ without any translation. $|X_T \cap Y|$ refers to the number of common words after translating the words in $|X|$ to the language of book $|Y|$. $|LCS|$ is the length of the longest common subsequence between the word sequence representations. TRANSNIQ-its and TRANSNIQ-cs scores are also shown where the corresponding thresholds are 0.49 and 0.023, respectively.

Char err. rate(%)	Word err. rate (%)	English Book X	German Book Y	$ X $	$ Y $	$ X \cap Y $	$ X_T \cap Y $	$ LCS $	TRANS its	TRANS cs
0	0	Egmont	Egmont	2395	3224	32	492	251	0.643	0.090
1	5.33	Egmont	Egmont	2395	4232	34	474	235	0.623	0.074
3	15.29	Egmont	Egmont	2395	5109	34	438	209	0.593	0.055
5	24.38	Egmont	Egmont	2395	7395	33	455	187	0.570	0.044
10	43.68	Egmont	Egmont	2395	10256	30	337	128	0.514	0.026
0	0	Faust	Egmont	3706	3224	27	416	43	0.426	0.012
1	5.33	Faust	Egmont	3706	4232	29	406	42	0.415	0.011
3	15.29	Faust	Egmont	3706	5109	35	388	40	0.401	0.008
5	24.38	Faust	Egmont	3706	7395	37	363	37	0.388	0.007
10	43.68	Faust	Egmont	3706	10256	41	306	35	0.374	0.006
0	0	Kant	Egmont	2625	3224	6	270	31	0.396	0.011
1	5.33	Kant	Egmont	2625	4232	9	263	30	0.387	0.009
3	15.29	Kant	Egmont	2625	5109	10	250	30	0.374	0.007
5	24.38	Kant	Egmont	2625	7395	11	236	29	0.364	0.007
10	43.68	Kant	Egmont	2625	10256	12	205	26	0.345	0.005

for texts having similar style and vocabulary. The third scenario investigates the case in which two different books are written by different authors - the German version of Goethe's *Egmont* and an English version of "The Critique of Pure Reason" by Immanuel Kant. In a collection the most common scenario is one where the books are not translations of each other and the authors are also different.

In Figure 5.3, it is clear that TRANSNIQ-its and TRANSNIQ-cs scores are substantially larger for the true translation pair compared to the other two non-translation pairs. For all scenarios, the translation scores are the highest when there is no document noise and they gradually fall as the amount of noise is increased. In this case, TRANSNIQ-cs score's rate of fall is higher compared to TRANSNIQ-its. For the true translation pair, TRANSNIQ-its and TRANSNIQ-cs scores fall below the given thresholds at approximate word error rate levels 49% and 44% respectively. Notice that these word error rates are very high and unlikely to happen in practice for printed books. As discussed in Section 3.5, the estimated OCR word error rate of scanned books in the IA database is typically less than 15%. The proposed method is robust to the OCR errors found in scanned book collections.

Table 5.11 provides further detail. In all scenarios, it is seen that the number of unique words increases as the amount of noise increases. The reason is that document noise (or OCR errors) tend to produce arbitrary words which are not in the vocabulary of the book (or even the language).

It is seen that the non-translation book pair having the same author has more common words and higher translation scores compared to the third scenario where the non-translation book pair has different authors. The reason is that different books written by the same author are likely to have more common words in the vocabulary, even though one of them is translated by someone else. Despite this effect, the proposed method successfully discriminates both non-translation book pairs from the true translation pair.

The length of the sequence of words following the same order in both contexts is a clear indication of translation. This can be seen more clearly for the book pairs having the same writer (scenarios 1 and 2). See Table 5.11. Both book pairs have comparable numbers of common words in their representations. This information alone does not help discriminate these two cases. However, the length of the LCS is considerably higher for the true translation pair. This means that there are a large number of words following the same order for the true translation pair whereas it is not the case for the non-translation pair. The word sequence information is therefore a strong feature to detect translations. It is sufficient to have a small number of words in common preserving the same order compared to the total number of unique words in the book.

5.2.7 Efficiency

Without the dictionary transformation, over 12K book pairs can be compared by aligning the the sequence of unique words as described for the case of monolingual partial duplicate detection in Section 4. After the dictionary transformation, however, the word sequence representation for the source text typically get longer. The increase in length depends on the fertility rate of the bilingual dictionary used ³. In the case of English-German experiments using the 62K dictionary, transformed word sequences are typically two times longer than the original. It should also be noted that the transformed sequence might include terms more than once. Therefore asymptotically faster $O(n \log \log n)$ LCS algorithms [49, 26] are not applicable in this particular case.

The dictionary transformation is performed only once for every book pair. The resulting word sequences are hashed into 32-bit integers and put into binary files in the same order as they appear in the text as described in Section 4.2.6. Despite the increase in word sequence length after the dictionary transformation, the proposed

³Fertility is defined to be the average number translation entries for each term in the dictionary.

approach compares 10K book pairs per second on a single core and therefore it is quite scalable. The proposed approach becomes much slower for the maximum term frequency values higher than $M = 1$ (2K pairs / sec for $M = 2$). In the case of aligning the entire texts without any term sampling (i.e., $M = ALL$), aligning a single book pair takes approximately 100 milliseconds (10 book pairs / sec) using the Recursive Text Alignment scheme. The proposed approach using only the sequence of unique words with dictionary transformation is a thousand times faster and more accurate than the brute force text alignment approach.

In this chapter a translation detection framework is proposed for scanned book collections. The proposed system uses the sequence of unique words representation to match translation pairs of books in the collection. In the next chapter, an efficient cross-lingual text alignment scheme is proposed to map translated portions of texts across document translations.

CHAPTER 6

ALIGNING LONG NOISY TEXTS ACROSS LANGUAGES

Given a pair of documents written in two different languages, the task is to find the corresponding pieces of text in the form of translation despite the presence of document noise, additional and/or missing text, and, the absence of any structural information. More specifically, it is assumed that the input documents are *not* necessarily exact translations of each other (i.e., there is no 1:1 correspondence between the texts) and there is no structural information or metadata to infer the position of each correspondence. The input documents might be quite long hundreds of thousands words. Moreover, the documents might contain a considerable amount of noise due to Optical Character Recognition (OCR) errors and/or version differences between the original and translated text. It should be noted that OCR errors often corrupt sentence and paragraph boundaries and as a result conventional cross-lingual alignment and retrieval approaches become ineffective. In addition, there is always an inherent noise due to the quality and style of the translation. These complications make the task quite challenging and there is a need for an efficient and robust solution.

There are several applications of mapping corresponding portions of text across document translations. In the context of machine translation, cross-lingual alignment approaches can be used to create a parallel corpus. Corresponding portions of text can be used for training bilingual dictionaries and machine translation models. Given a bilingual corpus, one can also detect document translation pairs by aligning the contents of the documents. Linguistic annotations of one book can be projected to

its translated version if the correspondences are known. Another application is to search text across document translations. One can retrieve the corresponding text in the translated version if the correspondences are known. Automatic approaches for mapping document translations can also be used for automatic evaluation of cross-lingual search engines. One can also define several other potential applications.

The problem of mapping corresponding portions of text across document translations also arises in many other disciplines including education, languages, comparative literature, business management and law. For example, many people in their work or leisure often have to look at a document and its translation. Legal experts may want to examine the legal documents, treaties, or contracts across languages but may not have the language expertise in the other language. Humanities scholars, religious scholars, historians, and others often need to compare different versions of texts to see where the translated version agrees and where there is either interpolated or missing text. Students studying Latin may want to look at the English translation of Virgil's Aeneid to better understand the material. In the case of translations of books, it is common for one book to have some extra material in the form of footnotes, endnotes, introductions, glossaries, and commentaries. In addition, the translations may not be exact. Figure 6.1 shows an example where the translation includes additional text which is not present in the original book. The OCR text output of scanned books may also have recognition errors which destroy the structure of the text such as sentence and paragraph boundaries. In this respect, scanned book collections are particularly challenging compared to other electronic documents as discussed further in the experimental section. In all the use cases mentioned above, automatic cross-lingual text alignment approaches would be quite useful to find the corresponding passages across document translations.

Mapping translated portions of text across document translations can be performed in a variety of ways. One possible approach is to automatically translate the

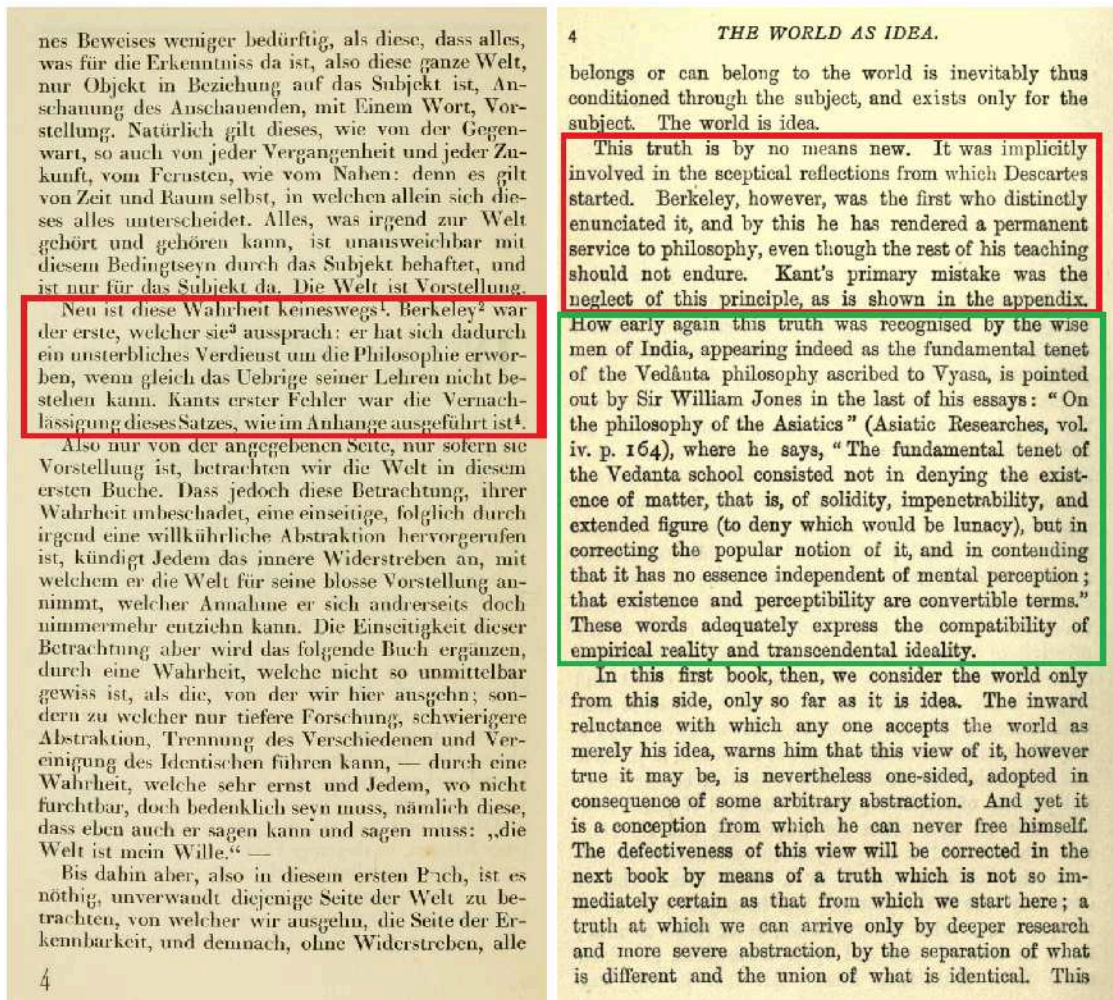


Figure 6.1. The red box on the left is a paragraph from the original version of the book “The world as will and idea” by A. Schopenhauer. The corresponding passage (red box) in the English translation on the right includes additional text (green box) which is not present in the original book.

text of the source document to the language of the target document using a machine translation system [19]. Once the two texts are in the same language, mono-lingual text alignment and search methods become applicable. This approach requires efficient and robust machine translation systems to be available for each language pair. Another approach is to segment the input texts into sentences and find correspondences using sentence alignment tools, which are primarily developed for the purpose of training machine translation systems. Most of the sentence alignment techniques

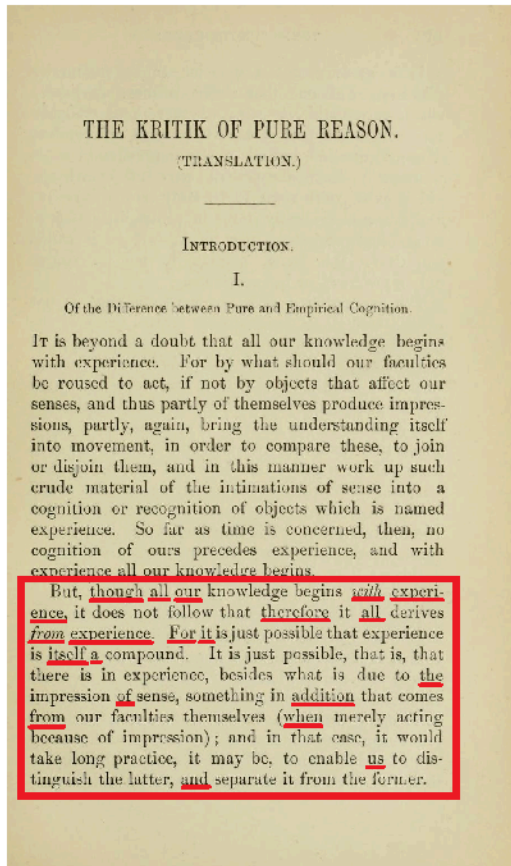


Figure 6.2. The red box on the left denotes a query passage from the English translation of Kant’s Critique of Pure Reason. On the right the corresponding passage (red box) is automatically derived in the German original of the book. The passages can be derived because of the matching words (underlined in red) between the two books derived by alignment. Note that the German book contains a lot of extraneous material - comments - which are not found in the English version.

require reliable sentence boundaries which may not be available as in the case of scanned book collections. Besides, sentence aligners typically assume that there is a 1:1 mapping between the source and translation without any extra or missing text. Otherwise they can get very inefficient, e.g., [72]. Yet another approach is to segment the text into sentences or passages and use cross-lingual retrieval frameworks to locate translations. It should be noted all the alternative approaches discussed above directly use text structure in the form of passage and/or sentence boundaries which may not be always available. On the other hand, the proposed approach aligns the

input documents directly at the word level and therefore does not need the text to be structured in any way. To the best of our knowledge, none of the approaches in the literature are designed for aligning long noisy texts such as noisy OCR transcripts of books which may include large portions of additional and/or missing text.

Since the documents are translations (although not necessarily literal ones), the *global* order of ideas must be preserved; however, word order in a sentence may not be preserved across languages. Further, a long sentence in one language such as German may be split up into many smaller sentences in a second language. Often the two documents are not identical because there is additional material. Books for example, typically have extra material such as footnotes, endnotes, introductions, glossaries, and commentaries, which are present in one book but may not be present in the other. The left-hand side of Figure 6.2 shows a query passage (red box) from the English translation of Kant’s Critique of Pure Reason. The right-hand side shows the automatically mapped passage (red box) from the German original. Note that the book and its translation are not identical because of extra material. This makes the problem of finding corresponding portions somewhat challenging.

Clearly, one must have the document and its translation to perform cross-lingual text alignment. It is assumed that a book and its translation are already available. The techniques for finding translations in scanned books collections are elaborated in Section 5. Some of these techniques can be run rapidly even over large collections of books.

Our proposed cross lingual text alignment scheme incorporates insights from the recursive text alignment and the translation detection frameworks proposed in Chapters 3 and 5 respectively. More specifically, the book in the source language is transformed to the language of the target book using the dictionary look-up approach as described in Chapter 5. The original words in the source book are mapped to their possible translations in the transformed word sequence. The transformed word

sequence is later aligned with the target book using the Recursive Text Alignment Scheme as proposed in Chapter 3. Word correspondences are later used to generate sentence and paragraph level alignment of translated texts. It is shown that the proposed approach successfully aligns translations of books where there is no structural information or metadata available. Experiments also show that the proposed approach outperforms the cross lingual retrieval and sentence alignment baselines for searching translations in scanned books datasets. The processing time of the proposed scheme is less than 30 milliseconds (100ms with I/O time) on a single core for aligning book length documents. This is several orders of magnitude faster than the sentence based alignment baseline.

The rest of this section is organized as follows: The proposed framework and the evaluations are discussed in detail in Sections 6.1 and 6.2, respectively. The proposed cross-lingual text alignment approach is later used to visualize the translated portions of texts as described in Section 6.3.

6.1 The proposed framework

An efficient alignment-based approach is proposed for mapping translated portions of two input texts. The core idea is that once a text is translated, a large number of words typically follow the same global order of the individual passages and sentences. The proposed framework first aligns the two texts at the word level as described in Section 6.1.1. These word to word correspondences are later used to align individual passages for search and retrieval purposes as described in Section 6.1.2. The details are elaborated in the following subsections.

6.1.1 Recursive Translation Alignment (RTA)

The dictionary look up approach described in Section 5 is adopted to match translated content across languages. More specifically, the book in the source language

is transformed in to the language of the target book using a look-up dictionary and aligned with the target book using LCS. However, conventional sequence alignment techniques do not scale for aligning these long word sequences. The Recursive Text Alignment Scheme proposed in Chapter 3 is therefore extended for this purpose. In this context, the new alignment scheme is referred as Recursive Translation Alignment (RTA) algorithm.

In Figure 6.3, the recursive translation alignment framework is depicted for a short section of the book “The Miser” written by Molière. The words “Martin”, “esclave”, “beautés”, “sévéritiés”, “résolution” and “danis” are the unique words in the French version of the book because they appear only once in the entire text. In the English version, the unique words include “Martin”, “harshness”, “resolution”, “disguised” and “serwant”. All the unique words are colored in red in the corresponding context. In the top level, the sequence of unique words are aligned and the matching ones are used as anchor points to divide the entire sequence into a set of corresponding texts. In this particular example, the unique words “Martin”, “sévéritiés” and “résolution” match the words “Martin”, “harshness” and “resolution” respectively in the same order. The word “Martin” is matched because the word is preserved across translations. The other two anchor words are matched since the dictionary includes them as possible translations. The unique words selected as anchor points are underlined. It should be noted that OCR errors might generate a number of unique words such as “danis” and “serwant” which are not even legal words in the language. Such words are quite unlikely to be anchor points since they do not align with any word in the other text.

In the next stage, the texts in between each consecutive anchor word are aligned recursively as shown in Figure 6.3b. The words “Élise”, “esclave”, “beautés” and “violence” became unique words since they appear only once inside their own text segment. Similar to the first stage, these unique words are aligned with the corre-

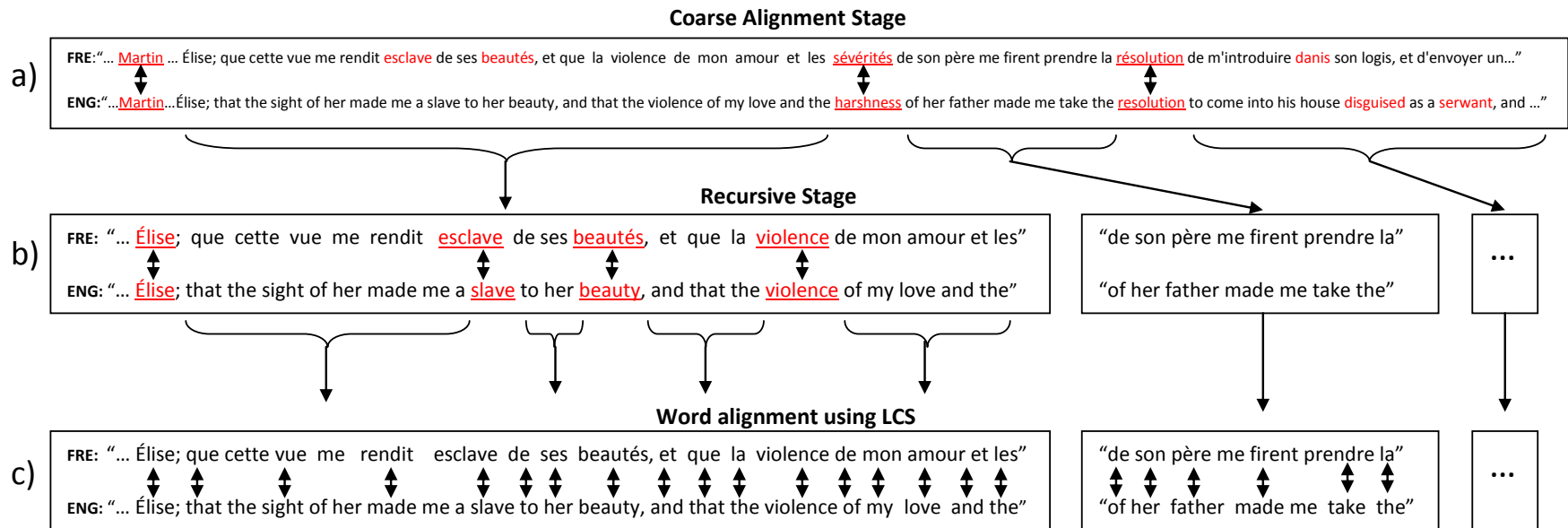


Figure 6.3. The recursive translation alignment scheme (RTA) depicted for two short texts in English and French. “...” stands for skipped content for illustration purposes. Double headed arrows indicate matching words.

sponding words in the English version of the text. The recursion terminates once the text segments in between matching words become short enough for dynamic programming (i.e., shorter than 400 words). At the leaf level text segments are aligned using the standard dynamic programming implementation of LCS. Aligned text segments are later concatenated to generate the complete global alignment.

In this particular example, there are a large number of matching words following the same order in both texts. This may not always be the case because the order of words or even sentences may significantly change after translation. Despite this, a large number of words are expected to follow the same order after alignment if the two texts are translations of each other. These matching words are sufficient to locate the translated portions of the texts reliably.

6.1.2 Passage level alignment

The Recursive Translation Alignment algorithm produces a word level alignment for a given document translation pair. It should be noted that some of the words may not align with any word in the target sequence. It happens especially when the dictionary does not have any translation entry for the words in the source document, or, the local word order is not preserved across translations. This is not a problem for the purposes of passage level alignment. In fact, a small number of matching words in the alignment output are sufficient to locate the translated text in the target document. The aim now is to convert the word level alignment to passage level. It is more convenient for users to search and visualize the translated texts on the passage level. Passage level analysis also enables us to evaluate the effectiveness of the proposed system against the sentence based alignment and the CLIR baselines which are used to rank the passages.

For each passage in the source document, passage level alignment is produced by counting the total number of matching words for each passage in the target document.

The passage which has the highest number of matches is matched to the source passage. In the experimental section we will see that this method actually works quite well. In order to improve this simple scheme, we tried two modifications. The first one is to ignore stopword matches and use only the remaining ones to determine the matching passages. However, this approach did not perform well. This actually implies that matching stopwords are actually useful since their respective order may also be preserved in the local context. Alternatively, one can also use term weighting approaches to assign a matching score for each passage and rank them accordingly. The weight w_i for each matching word i is defined as

$$w_i = \frac{1}{\log(f_i + 1)} \quad (6.1)$$

where f_i is the frequency of the word in the target document. In Section 6.2 we show that this weighting scheme showed minor improvement over the simplest word counting approach. Therefore the framework uses only the matching word counts to align passages.

6.2 Experiments

The effectiveness of the proposed approach is evaluated for various datasets with different characteristics including e-books, government documents, scanned books and synthetic texts. Different evaluation metrics are adopted for various retrieval tasks. The results are compared against two baselines including a sentence aligner and a cross-lingual information retrieval baseline. Details are elaborated next.

6.2.1 Datasets

The **EUROPARL parallel corpus** is a standard text collection from the proceedings of the European Parliament [50]. Version 3.0 contains speeches from the period 04/1996-10/2006. There are more than 600 documents each of which is trans-

lated in to eleven languages. This corpus is different from the book datasets since each sentence is translated and the pairwise correspondence between sentences is provided. Since they are e-texts they do not have any document noise unlike the scanned book collections. This dataset is primarily used for training machine translation models and sentence alignment algorithms. Experiments are performed for the English-German language pair. After removing the tags, the average number of words per document is 50,360 in the English collection. These documents are much shorter than a typical book.

The Gutenberg dataset consists of four translation pairs of e-books downloaded from the Project Gutenberg website. It includes Molière’s “Avare” (“The Miser”) in English-French, and Shakespeare’s masterpiece “Othello” in three language pairs: English-German, English-French and English-Finnish. It should be noted that Finnish is not a Indo-European language unlike the other languages listed. In contrast to the EUROPARL corpus, these translations are not literal. These plays are written in verse form and do not contain any document noise (i.e., OCR errors). Unlike Othello, Avare is written in prose form and the speeches are relatively longer. Manual evaluation of the search results is prohibitive in this context since the books typically contain a large number of passages. Instead we propose a novel automatic approach to annotate the books. The inherent structure of the plays is exploited for this purpose. In a play, each character takes turns and this is indicated by the character’s name or initials. For example the following excerpt from Hamlet shows Hamlet and Ophelia taking their turn as the speaker (... indicates the rest of the text).

Hamlet: To be, or not to be, ...

Ophelia: Good my lord, ...

Hamlet: I humbly thank you; ...

Ophelia: My lord, I have remembrances of yours,..

In a play, the global order of the characters in the scene is strictly preserved across translations. Therefore, one can align the speech tags of the two books to obtain the corresponding passages between the two versions of the play. The speech tags and/or the names of the characters may vary across books. In such cases some manual preprocessing may be required to equate the names and their translated versions in the text. Speech tags are removed at all other stages to make sure that they are not exploited for retrieval purposes. The use of plays is merely to enable quantitative evaluation.

A set of synthetic texts are generated for evaluating the effectiveness of the proposed approach for various levels of document noise and content overlap between the two texts. The approach used to generate this dataset is elaborated further in the appropriate section.

The Scanned Book dataset includes the OCR output for 30 translation pairs of books downloaded from the Internet Archives website [1]. These texts are basically long strings of text containing varying amounts of OCR errors. Document structure such as paragraph and sentence boundaries are not necessarily preserved in the OCR output. The amount of content overlap between the original and the translated version of the book may vary significantly because of edition differences and extra material such as annotations. For evaluation purposes, five query sentences are defined for each book pair. There are 150 queries in total. The position of the corresponding text in the target document is manually labeled for each query. The scanned book dataset is useful to evaluate the effectiveness of the proposed approach for long noisy texts with varying amounts of content overlap. This is the most challenging dataset.

6.2.2 Baselines

Two baselines are adopted for testing the effectiveness of our approach.

The **Cross-Language Information Retrieval (CLIR)** baseline ranks the paragraphs of the target book against a query passage selected from the source book as follows:

$$t_{match} = arg \max_t \Pr(t|q) \quad (6.2)$$

where q refers to the query paragraph in the source book and t_{match} refers to the matching paragraph in the target book written in the other language. The translation model probability $Pr(t|q)$ is estimated using IBM Model 1 translation model as described in [19]:

$$\Pr(t|q) = \frac{\epsilon}{(l+1)^m} \prod_{j=1}^m \sum_{i=0}^l \Pr(t_j|q_i) \quad (6.3)$$

where q_i and t_j refers to the i th and j th word in the query and target passage respectively. The length of the query passage is denoted with l whereas the target passage has the length m . In a nut-shell, this model regards each passage as a “bag of words” and tries to estimate $\Pr(t|q)$ based on the translation probabilities between the individual words in the query and target passages. Although the IBM Model 1 has been primarily used for building statistical machine translation systems, this approach has been adapted to several other domains for different types of documents, namely, in information retrieval(IR) [13, 73], cross-lingual IR [81], cross-lingual plagiarism detection [11] and bilingual text classification [23]. In the experimental section, the same dictionary is used for both RTA and CLIR approach for a fair comparison. Given a word in the dictionary, the translation probabilities are assumed to be uniform over all its possible translations. The following smoothing approach is used to estimate the translation probabilities for words which are not covered by the dictionary:

$$\Pr(t_j|q_i) = (1 - \lambda) \Pr(t_j|q_i) + \lambda \Pr(t_j) \quad (6.4)$$

$Pr(t_j)$ is estimated as $\frac{1}{|V_t|}$ where $|V_t|$ refers to the vocabulary size of the book written in the target language. λ is a constant weighting factor between zero and one and it is typically set to a small value.

Sentence Level Alignment (SLA) baseline aligns the sentences in the source and target book using Moore’s sentence alignment algorithm [72]. Basically Moore’s sentence aligner uses sentence lengths and word correspondences to align sentences in a given pair of documents written in different languages. It does not require any translation lexicon to be given. The word correspondences and translation probabilities are estimated from the given texts. The first stage of the algorithm is to align the sentences based solely on their relative lengths. Sentence pairs which are likely to be translations are forwarded to the next stage where the word correspondences are determined. The last stage combines both sentence length and word correspondences to determine the matching pairs of sentences. The design principle of this method is to provide high alignment accuracy for the automatic generation of parallel corpora task for machine translation. An automatic sentence segmentation tool is used for finding sentence boundaries ¹.

One drawback of the baseline approaches is that they require the sentence and/or paragraph boundaries to be available. This structural information may not always be available especially for scanned book collections because of OCR errors. On the other hand, RTA aligns the two text directly at the word level to find corresponding passages without using any structural information in the text.

6.2.3 Experiments with the EUROPARL dataset

The effectiveness of the proposed approach is investigated for the EUROPARL parallel corpus on the sentence alignment task. The English-German 62K dictionary is used for experiments. Each pair of text is aligned using RTA and the sentence

¹<http://cogcomp.cs.illinois.edu/page/tools.view/2>

Table 6.1. Precision, recall and F-measure scores for the sentence alignment task on the EUROPARL dataset.

Approach	Precision	Recall	F-measure
SLA	0.999	0.981	0.990
RTA	0.984	0.952	0.968

correspondences are generated as described in Section 6.1.2. If there is no matching word for a sentence in the alignment, then it is assumed that there is no match. Precision, recall and F-measure scores are generated for SLA and RTA. It is clear that both approaches provide very high matching scores with SLA being slightly better. This is expected since sentence aligners are precisely designed for aligning sentences in parallel corpora such as the EUROPARL collection. We will see that when the texts are not identical and OCR errors are introduced, sentence aligners do not work as well.

6.2.4 Experiments with Gutenberg books

The proposed approach is first compared to the CLIR baseline on the passage retrieval task for three different language pairs. Each speech in the source document is regarded as a passage and the task is to map the corresponding speech in the target document. P@1 score is calculated for all passages in the source document and the average score is reported. If RTA can not match any word in the target text for a given query passage, it does not return any answer. The effects of stop word removal and term weighting are also investigated for the proposed approach. Results are shown in Table 6.2. RTA does substantially better than the baseline algorithm in all cases, although both approaches use exactly the same dictionary. For the English-Finnish pair, both approaches perform much worse, particularly the CLIR baseline. It should be noted that the English-Finnish dictionary contains less than three thousand words whereas English-German and English-French dictionaries contain 62K and 17K words respectively. The retrieval scores for the book *Avare* are

Table 6.2. Average P@1 scores of RTA and the CLIR baseline for the passage retrieval task on the Gutenberg dataset. PL refers to the average word count of the passages in the English version. TW means term weighting, nTW means no term weights are used, SW means stopwords are used while nSW means without stop words.

Book	Lang. pair	PL	RTA nTW SW	RTA TW SW	RTA nTW nSW	RTA TW nSW	CLIR
Oth.	En-Ge	24.1	0.758	0.764	0.622	0.629	0.439
Oth.	En-Fr	24.1	0.697	0.695	0.583	0.586	0.453
Oth.	En-Fi	24.1	0.446	0.449	0.275	0.279	0.075
Avare	En-Fr	46.3	0.828	0.831	0.689	0.697	0.409

Table 6.3. The effect of dictionary size for the book Othello (English-German). RTA compared with the CLIR baseline using P@1 score.

Dictionary	RTA	CLIR baseline
Eng-Ger 5K	0.669	0.308
Eng-Ger 62K	0.758	0.439

significantly better than Othello for the same language pair. Notice that the book Avare is written in prose form and its passages are on average much longer than the passages in Othello.

Stop words are generally known to have a negative impact on retrieval accuracy and therefore they are eliminated. However, stop words help improve the retrieval scores of RTA significantly as shown in Table 6.2. It should be noted that RTA matches a stop word not only because of its occurrence but also its position in the text relative to other matching words. Stop words are especially useful if the dictionary is small since without stop words, there are a lot fewer words for mapping corresponding passages.

Term weighting is another factor which is known to improve the retrieval effectiveness and therefore widely practiced for various information retrieval tasks. In the case of RTA, term weighting provides a limited improvement over the retrieval scores as seen in Table 6.2. This suggests that it is less critical which words match in particular for RTA. The words which are common to both texts and following the

Table 6.4. Average precision (P), recall (R) and F-measure (F) scores of RTA and the SLA baseline for the sentence alignment task on the Gutenberg dataset.

Book	Language pair	RTA			SLA		
		P	R	F	P	R	F
Othello	Eng-Ger	0.783	0.788	0.785	0.895	0.573	0.699
Othello.	Eng-Fre	0.749	0.754	0.751	0.965	0.651	0.777
Othello	Eng-Fin	0.611	0.615	0.613	0.938	0.724	0.817
Avare	Eng-Fre	0.850	0.861	0.855	0.989	0.747	0.851

same order are particular interest to RTA. Given the results, the term weights are not used in the rest of the paper.

The correlation between the dictionary size and the retrieval effectiveness is evaluated for the same book and language pair as shown in Table 6.3. The larger dictionary helps find more word correspondences and hence provides significantly better results for both RTA and the baseline.

Next, the proposed approach is compared to the sentence level alignment (SLA) baseline for the sentence alignment task. The Gutenberg texts have no OCR errors, therefore the sentences were split reliably by the sentence splitter. Sentence level alignments are generated for RTA using the word correspondences as described in Section 6.1.2. It is assumed that the two sentences match correctly if the corresponding sentences are in the same passage in the play. The reason is that sentence boundaries may not be preserved across translations especially for literature in verse form. Overall precision, recall and F-measure ($\frac{2 \times P \times R}{P + R}$) scores are calculated for all sentences in the source document as shown in Table 6.4. The results show that RTA has the highest recall in all cases except the English-Finnish pair which suffers from a small 3K size dictionary. RTA has significantly higher F-measure score for the Othello English-German and Avare English-French book pairs. For Othello English-French, the F-measure scores are comparable. On the other hand, SLA provides the highest precision scores in all cases. This is not surprising since the primary design principle of the sentence alignment tools is to find corresponding passages accurately. Pre-

cise matches are more useful for various natural language processing tasks. However, higher recall values are desirable in the case of searching document translations.

6.2.5 Synthetic experiments

There are two main factors which determine the effectiveness of translation alignment: the OCR accuracy and the content overlap. OCR errors quite often corrupt characters in the text and create words which may not even be in the vocabulary of the language. Thus, they have a negative impact on the retrieval accuracy. The content overlap is defined by the total amount of corresponding text between the original and the translation of the document. More specifically, the content overlap is the number of translated sentences divided by the total number of sentences in the translation. The translated book may include additional material in the form of a commentary, preface, appendix etc. However, the content overlap is generally high for book translations.

The effects of OCR errors and the content overlap are investigated by generating synthetic texts and evaluating the passage retrieval accuracy. The process is as follows: a book translation pair is downloaded from the Project Gutenberg website [2]. In our case, these books are Shakespeare's *Othello* in English and German. The German version is used to generate a synthetic document which is to be aligned later with the English version. Extra material (in German) is inserted in to the German version of *Othello* after randomly selected passages to simulate the additional content. Chunks of text are selected from random locations of the German version of Goethe's *Faust* (from Gutenberg) according to a Gaussian distribution with a mean of 10 sentences and a standard deviation of 5 sentences. Each chunk is inserted into the German version of *Othello* at a random position using a uniform distribution. This process is repeated until a certain amount of content overlap is reached. The next stage is to introduce a certain amount of character level noise to the synthetic

Table 6.5. P@1 scores for varying amounts of document noise and the content overlap.

Overlap (%)	RTA			CLIR Baseline		
	0%	24%	43%	0%	24%	43%
40	0.590	0.450	0.331	0.441	0.319	0.217
50	0.632	0.502	0.372	0.441	0.318	0.216
60	0.664	0.543	0.417	0.441	0.318	0.218
70	0.703	0.577	0.454	0.441	0.318	0.218
80	0.725	0.609	0.498	0.441	0.320	0.216
90	0.746	0.644	0.513	0.441	0.321	0.217
100	0.764	0.669	0.553	0.440	0.320	0.216

text. The character level noise model in [35] is used to simulate OCR errors in the synthetic documents. Several string operations (insertion, deletion and replacement) are done over the entire text until the desired amount of noise is introduced. The distribution for each edit operation is assumed to be uniform, i.e., $[1/3, 1/3, 1/3]$ respectively. The large English-German dictionary (containing 62K word translations) is used in the evaluations for both the CLIR baseline and the recursive alignment scheme. Characters are case folded, the punctuations and numerals are ignored at all stages. The experiments are repeated one hundred times and the results are averaged for varying levels of content overlap (from 40% to 100% with 10% increments) and the character level OCR error rates (0%, 5% and 10%). It should be noted that a German word contains about 5.4 characters on average. Therefore 5% and 10% OCR error rate is equivalent to 43% and 24% word error rate respectively.

P@1 scores are shown in Table 6.5 for different levels of OCR word error rates and the content overlap. It is clear that our algorithm outperforms the CLIR baseline in all cases. P@1 score of our algorithm increases as the content overlap increases. This is also true for all the levels of OCR error rates. The baseline performance is not sensitive to the amount of content overlap. In other words, it is able to retrieve the corresponding passage again despite the extra material. OCR error rate is a key determinant for the success of both frameworks. There is a drastic fall in the

Table 6.6. Average P@1 scores for the sentence retrieval task on the scanned book collection.

Method	k					Time (sec)
	0	1	3	5	20	
RTA	0.773	0.840	0.847	0.860	0.940	0.7
CLIR	0.167	0.180	0.213	0.213	0.220	0.2
SLA	0.127	0.133	0.133	0.133	0.147	815.7

precision scores if a large amount of words are misrecognized. It should be noted that the average word error rate for the scanned book collections is estimated to be between 5% to 15%, as reported in Section 3.5.

6.2.6 Experiments with real scanned books

The effectiveness of the proposed approach is evaluated for the sentence retrieval task on the scanned book dataset. Sentences boundaries are automatically determined using a sentence splitter. This stage is noisy since the sentence splitting depends on several features including capitalization and punctuation. OCR errors quite often corrupt not only letters but also punctuation marks. The negative effects of OCR errors in sentence splitting become more severe if both texts contain significant amount of OCR errors. The English-German 62K dictionary is used for aligning the book pairs using RTA and the matching sentences are generated as described in Section 6.1.2. There are 150 queries in total. Average precision, recall and F-measure scores are generated for RTA, CLIR and SLA. A sentence proximity threshold k is introduced for further analysis of the results. The retrieved sentence is assumed to be correct if it is within k sentences of the actual matching sentence in the target document. The notion is that the reader can still find the correct passage if the position of the retrieved text is slightly wrong.

The main advantages of RTA become clearer in the case of searching translations in scanned book collections. The retrieval results are shown for different values of k in Table 6.6. RTA clearly outperforms both CLIR and SLA with a large margin for all

values of k . The primary advantage of RTA is that, it does not use any text structure information in the text. RTA treats the entire text as a single sequence of words and operate the alignment at the word level. However, both CLIR and SLA depend on the sentence boundaries which are not reliable for scanned book collections. The second advantage is that, RTA exploits the long range order of words in both texts to find correspondences. On the other hand, the CLIR baseline represents each passage or sentence as a bag of words and ignores any local or global word sequence information. The SLA baseline only uses the order of sentences in the text but not the order of individual words across sentences.

The sentence aligner baseline performs the worst in all cases. Recall that Moore's aligner first uses sentence lengths to find a set of initial matches and the word translation model is trained accordingly. The effectiveness of Moore's aligner is therefore heavily depend on the accuracy of the initial pass. This paradigm is therefore not applicable to search translations in scanned book collections.

Yet another advantage of RTA is the speed. The alignment of translations is carried out in a fraction of a second for a translation pair of scanned books. In Table 6.6, the average query resolution time is also given per book pair including any I/O time. The sentence aligner is the slowest approach among others (three orders of magnitude). In [16], it has been reported that Moore's aligner slows down drastically if there is a significant amount of additional or missing text between the translations. Although Moore's sentence alignment toolkit can be optimized for better performance, this approach is very unlikely to outperform CLIR or RTA in terms of processing time. Note that both RTA and SLA depend on offline computation of correspondences. After the offline phase, the queries are resolved using a look-up table. On the other hand, the CLIR approach resolves the queries online for each query.

Aber gleich in dem ersten Satze »einer Apologie stellt er sich auf den Boden der Frömmigkeit und Philosophie, an welchem nur dem Urtheil der Zeit und nach eigener Abriecht die frommen und philosophischen Kaiser selbst stehen wollten. Auf den Christus eifrig beruft er rief ihnen gegentüber in ganz stoischer Weise. Die Wahrheit tadelt er — eben- fällt todt — den tödtlichen entgegen .

(a)

But in the very first sentence of his Apology he takes up the ground of piety and philosophy, the very ground taken up by the pious and philosophical emperors themselves, according to the judgment of the time and their own intention. In addressing them he appeals to the 'Christ' in a purely Stoic fashion. He opposes the truth — also in the Stoic manner — to the mortal enemy.

(b)

Figure 6.4. Sample corresponding passages for the English and German versions of the History of Dogma. Some OCR errors. RTA finds this passage while both other techniques fail.

Figure 6.4 illustrates a piece of text from a scanned book titled “The history of Dogma” - a book on religion and its translation taken from another scanned book. In this case both CLIR and SLA failed to retrieve the corresponding passage whereas RTA found the correct match. Both the German and English versions contain several OCR errors.

For another book pair, RTA failed to retrieve the corresponding passages in four out of five queries. Further investigations revealed that this is because of high rates of OCR errors in the German version of book. The OCR word accuracies for the English-German book pair are approximated as 86.2% and 28.1% respectively, using the automatic OCR evaluation framework described in Section 3.5. This is an extreme example since scanned books have higher OCR accuracy in general. It should be noted that both baselines failed to find the correct passages for all queries in this particular translation pair.

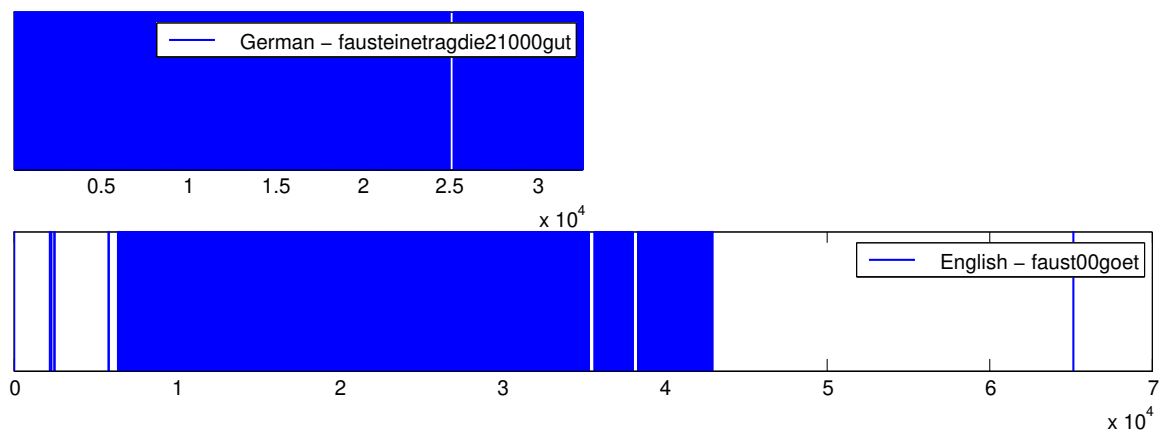


Figure 6.5. The figure shows the approximate overlap between a German original (upper bar) and the English translation (lower bar) of Goethe’s Faust using the proposed visualization scheme for translations.

6.3 Mapping translated portions of texts

The aim is to map and visualize the translated portions of the books which are known to be translations of each other. Our approach is to align the books using the recursive translation alignment (RTA) scheme introduced in Chapter 6.1.1. The translated portions of the texts are expected to have a higher number of matching words in the alignment. This is true even for high OCR error rates and significant changes in the language. It may be hard to define exact boundaries of the translated text. Therefore the same visualization approach used for partial duplicates is adopted as described in Chapter 4.3. Basically each book is divided into a number of bins. Each bin includes one hundred words. If there are more than a specified number of matching words (i.e., ten matches), then the corresponding bin is colored blue, otherwise white. Figure 6.5 shows an example visualization for a translation pair of books. These books are downloaded from the Internet Archive’s database [1]. The lengths of the bars reflect the relative sizes of the two books. Blue (black) denotes aligned portions. The German version contains just the text of Faust while the English version contains an additional preface and introduction and an appendix and notes at the end. The additional content is reflected by the white spaces in the English bar.

A cross-lingual text alignment approach is proposed to efficiently map overlapping content across translations. The proposed approach borrows several insights presented in the preceding chapters. So far, the idea is to exploit the OCR output to perform several tasks defined for scanned book collections. However, the OCR output may not be available in certain cases. In the next chapter, we propose several approaches to enable text search over the document images using image search mechanisms.

CHAPTER 7

SEARCHING DOCUMENTS USING IMAGE FEATURES

One way to search printed document images is to recognize characters and perform text queries. This approach is feasible as long as the document is not noisy and recognition accuracy is high enough. For noisy documents, Optical Character Recognition (OCR) error rates can get very high and recognized text becomes unusable. One remedy to this problem is to correct errors in the text. Error correction schemes are shown to decrease the amount of OCR errors [54, 12, 104]. Another option is to compensate for OCR errors in the query resolution stage. For example searching for n-grams of letters is shown to improve retrieval accuracy [41, 79]. One can also use both of these approaches to improve text search. However, these methods have limited capability in the sense that they can not retrieve information which is not captured by the OCR engine. In such cases, information is permanently lost and there is no way to recover it.

An alternative approach is to use image search mechanisms to help alleviate the negative effects of document noise. These approaches make use of image features directly in the search process. However, typical applications of these methods have certain limitations. One important limitation is that it is not possible to perform arbitrary queries. Users have to find an example word image from the same document collection and use it for querying (query by example, QBE) [84, 90, 6]. Query words which are out of vocabulary are therefore problematic. Scalability is yet another issue since computationally intensive operations are performed over high dimensional feature vectors for each query.

Here a number of approaches are presented to help alleviate these limitations for searching text in document images. The observation is that scanned books are typically printed in a single global font. Different instances of the query word are therefore visually similar to each other and we can use this information to facilitate or improve text search. In Section 7.2, we first propose a fast image search framework for searching text in the page images of a scanned book. It is shown that the proposed framework approach is as effective as searching text using high quality OCR output. Query resolution time is under 10 milliseconds over the entire book of length 363 pages. This approach is also shown to be effective for noisy document images written in Telugu for which there is no OCR engine available [121].

In Section 7.3, the image features used in this framework are later used to build a dependence model [70] which enables the resolution of *arbitrary* text queries in the document images. The dependence model approach requires the precomputation of dependencies between image features and letter bigrams. Those dependencies are used to resolve arbitrary text queries with an instant response time without the need for any example query word image. The proposed dependence model is shown to be effective for searching text in books printed in Latin, Telugu and Ottoman scripts. In the case of searching books printed in Latin script, it is demonstrated that the dependences between the visual terms and letter bigrams can be automatically learned using noisy OCR output. It is also shown that OCR text search accuracy can be significantly improved if the OCR text search is combined with the proposed approach. It should be noted that there is no commercial OCR engine available for Telugu and Ottoman script. In these cases, the dependences are trained using manually annotated document images. It is demonstrated that the trained model can be directly used to resolve arbitrary text queries in other scanned books despite font type and size differences. The details are elaborated in the following subsections. The visual features are described first.

7.1 Visual features

OCR engines rely on features extracted from connected components and they tend to make errors in recognizing underlined or crossed word images which are very common noise types in real document images. Connected component analysis is therefore not desirable for noisy document images. On the other hand, image search mechanisms are capable of accounting for partial matches between the word images for both search and retrieval tasks. This is basically achieved by using features which are robust to document noise and similarity functions which account for partial matches.

The offline processing starts with defining a number of salient points (also referred to as “keypoints”) in the document images. Keypoints must be repeatable for matching purposes, i.e., matching keypoints needs to be identified for different instances of the same word image. The Fast-Corner-Detector [91] is used to locate keypoints in the page images. SIFT (Scale Invariant Feature Transform [60]) features are extracted from the patches placed over the corner points. A feature vector of 128 dimensions is extracted to represent each keypoint. The scanned page images do not have any significant page skew, therefore the patch orientation is fixed to be zero degrees. If the word bounding box is known, then the patch size is set to be equal to the height of the box. Otherwise it is set to the line height of the text which is typically constant across the pages for scanned books.

Using high dimensional features for matching word images is computationally expensive. One well-known practice is to map feature vectors to discrete values using clustering techniques [75]. The Hierarchical K-Means (HIKMEANS) clustering algorithm is used for quantizing SIFT descriptors [110]. Each feature vector is given a discrete label according to the cluster it belongs to. This label is referred to as a “vis-term ID” and represented with the letter v . The visterm vocabulary is determined by the number of clusters defined in the clustering processing. Vocabulary size is fixed

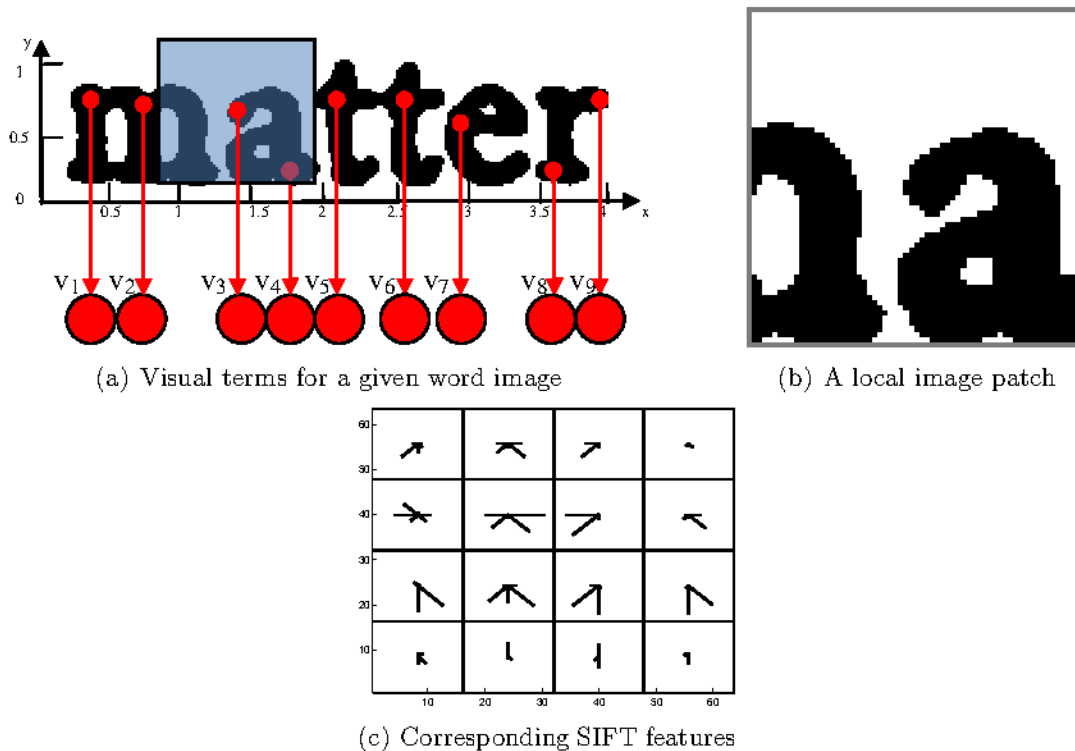


Figure 7.1. Visual features are shown for an example word image. In a) small dots correspond to the local interest points. Local patches extracted from interest points are quantized into visual terms (v_1, v_2, \dots, v_9) which are represented with large big circles at the bottom. b) and c) shows an example image patch from the word image and the corresponding SIFT features respectively. There are typically around 100 visual terms per word image.

at 4K visterms since smaller vocabularies are observed to provide higher matching performance.

At the end of the offline processing step, a word image I is represented with a set of visterms $\{v_1, v_2, \dots, v_m\}$. Each visterm v_i is a discrete number which corresponds to a local image patch placed over the keypoint i . Each visterm also has a (x, y) coordinate over the coordinate frame of the word image I . These coordinates are normalized by the height of the word image I if word bounding boxes exist. The visterms are stored according to their natural order on the X axis of the image plane as shown in Figure 7.1. An optimized version of an inverted index is also created offline for keeping all $\langle \text{word ID}, \text{visterm ID} \rangle$ pairs. The inverted index is later

used to efficiently find the common visterms between the query word and the test image.

In our experiments, extracting visual features from a single document image (12 megapixels) takes about 30 seconds using an unoptimized MATLAB implementation. Locating the corner points takes less than 1% of the total processing time. Placing the image patches on interest points and extracting SIFT features takes 96% of the processing time using the implementation provided by the VLFeat computer vision library [110]. The remaining 3% of the total processing time is spent on the discretization of feature vectors.

Using a GPU implementation of SIFT [113], the offline processing would take less than 5 minutes for a book with 200 pages and 100MB of main memory is sufficient for online queries. Efficient indexing of visterms ensures that resolving a single query takes about 0.01 second.

7.2 An efficient word spotting approach for noisy document images

Given a query word image, the aim is to identify and rank similar word images in the context of the book. The existence of common visual features is necessary but not sufficient to qualify a word image for being a match. Their spatial configuration also has to be consistent with the ones in the query word. A two stage similarity search framework is therefore devised for ranking word images given a query word image. The details of the proposed framework are elaborated in the following subsections followed by evaluations.

7.2.1 The proposed framework

Each word image is represented with the visual features described in Section 7.1. The first stage of the framework is to identify and weight the common visterms for

each test image. This stage is referred to as the coverage test and it helps eliminate false matches at early stages. In the next stage, a configuration score is calculated for the spatial arrangement of common visterms between the query and each test image. Finally all word images in the book are ranked based on a final similarity score which is a linear combination of the coverage and configuration scores:

$$Sim(I, Q) = \lambda Cover(I, Q) + (1 - \lambda) Config(I, Q) \quad (7.1)$$

where λ is a weighting parameter, I and Q are the sequence of visterms (sorted based on their X coordinates) of the test word and the query image respectively. The details of these score functions are described in the following subsections.

7.2.1.1 Coverage analysis

The coverage score simply accounts for the ratio of common visterms to the ones in the test image. There are certain visterms which are rare in the sense that they occur less frequently in the whole book but give strong evidence for the existence of certain letters. In order to incorporate this information, each visterm is given a weight which is inversely related to its collection frequency. More specifically,

$$Cover(I, Q) = \frac{\sum_{i \in I \cap Q} w_i}{\sum_{j \in I} w_j} \quad (7.2)$$

The weight w_i for the visterm i is defined as

$$w_i = \frac{1}{\log(f_i + 1)} \quad (7.3)$$

where f_i is the frequency of the visterm i in the whole book.

It should be noted that the coverage test does not account for multiple occurrences of visual terms in the word image. The reason is that visual terms which are positioned next to each other in the word image tend to obtain the same visterm ID as shown

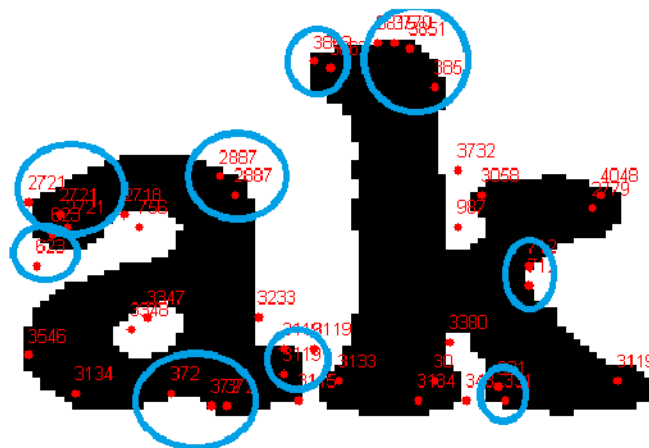


Figure 7.2. Corner points and corresponding visterm IDs for a letter bigram image. Visterms having the same ID are shown in circles. Notice that some visterms are spatially very close and therefore image features extracted from these regions are almost identical.

in Figure 7.2. Indeed, these visterms are artifacts of keypoint detectors and they do not provide any further evidence for resolving queries. It is not desirable to account for such visterms more than once for scoring.

After ranking word images based on the coverage score, the word images which are not in the top 10% of the list are filtered out. The rationale behind this approach is that the total number of true matches is not expected to be larger than the frequency of the most frequent word in the language of the book. For example, “the” is the most frequent word in English and constitutes approximately 6% of an English text. The result set for the query “the” should not therefore include more than 10% of the book despite the existence of a large number of false matches. Given that typical user queries consists of infrequent words, such as names and places, it is quite unlikely to miss true matches in the filtering stage.

7.2.1.2 Configuration analysis

One way to verify the configuration of visterms on the image plane is to search for a transformation matrix for the visterms in the query to the test image. A well-known

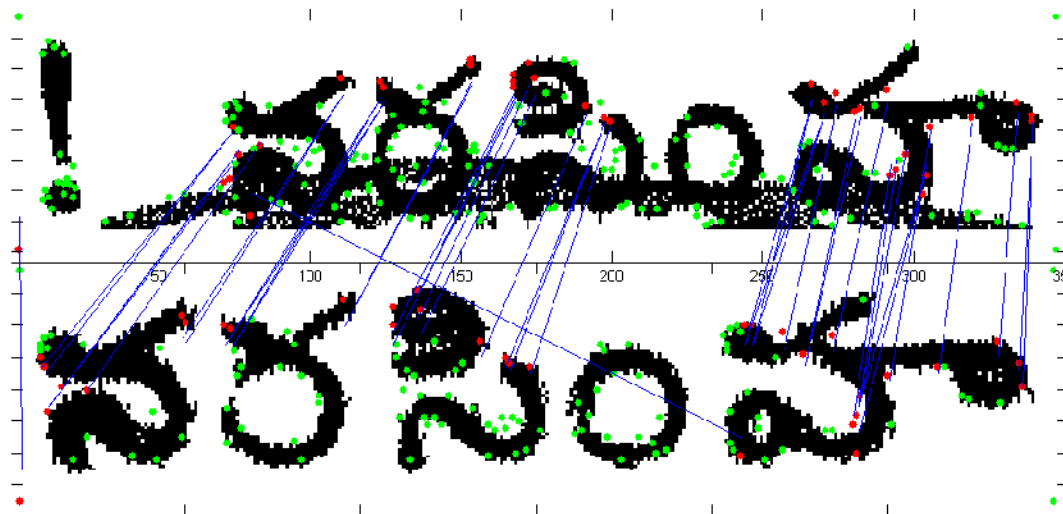


Figure 7.3. Matching visterms between two instances of a Telugu word from Telugu-1718 are shown. There are a large number of matching visterms following the same order even though the top image is underlined.

approach is the RANSAC algorithm [37]. In a nut-shell, RANSAC randomly selects a number of visterms in the query image and calculates a transformation matrix that maps them to the other image plane. This process is applied iteratively N times and the best transformation matrix is returned as the result. On every iteration, RANSAC fits a transformation matrix and calculates the quality of the fit by iterating over all visterms which makes it computationally expensive.

Here we devise an efficient method for testing the configuration of visterms between two word images using the notion that the respective order of letters is not supposed to change along the X axis. This is true even for text written in different fonts, faces and sizes (see Figure 7.3). Therefore it is sufficient to project the visterms on the X axis and compare the resulting sequence of visterms. There have to be a large number of visterms having the same order in both sequences if the two word images match. The problem turns out to be a search for the longest common subsequence (LCS) which can be solved quite efficiently for short sequences using dynamic programming [31]. Here we use the length of LCS to calculate the configuration

similarity as follows:

$$Config(I, Q) = \frac{\sum_{i \in LCS(I, Q)} w_i}{\sum_{j \in Q} w_j} \quad (7.4)$$

The numerator is the weighted sum of the visterms in the LCS(I,Q) and the denominator is the weighted sum of all visterms in the query image Q . The configuration score has a range $[0, 1]$ and it is 1 if the two sequences are identical and 0 if they do not have any common visterm.

7.2.2 Experiments

7.2.2.1 Datasets

Three books are used for the experiments. Two of them are printed in Telugu script and they are referred as “Telugu-1716” and “Telugu-1718”. These books contain word bounding box information along with the ground truth text. It should be noted that there is no publicly available OCR engine for Telugu script.

The other book is “Adventures of Sherlock Holmes” written by Arthur Conan Doyle in English. Document images and the OCR (ABBYY FineReader 8.0) output are downloaded from the Internet Archive’s website [1]. In total there are 363 document images including 113K English words. The OCR output also contains bounding box information for each recognized word. For evaluation purposes, a noise-free version of the same text is downloaded from the Project Gutenberg’s website [2]. For labeling word bounding boxes, the OCR output and the ground truth text are aligned using the recursive text alignment approach presented in Chapter 3. The estimated character accuracy for the whole book is 98.4%. Punctuations are ignored at all stages. A query test set is generated for each book. These word images are randomly selected from the vocabulary of the book itself with the restriction that the query words appear at least three times in the ground truth text. It should be noted that majority of the words in the vocabulary of the text appear only once in the case of Telugu language. There are only 50 words which appear at least three times in the

Table 7.1. MAP scores comparing the document image search and OCR text search for the English Book

Book	Search Method	MAP
English Book	OCR text search	0.923
English Book	image search	0.930

Table 7.2. MAP scores of the proposed image search framework for the Telugu books

Book	#words	MAP
Telugu-1716	21142	0.93
Telugu-1718	4284	0.94

entire text of the test book TELUGU-1718. For simplification purposes, the query set size is therefore set to 50 for all test books. For the English book, the estimated OCR accuracy is 92.3% for the words in the query test set.

7.2.2.2 Evaluation

The aim is to investigate the effectiveness of the proposed image search framework given a particular query. For this purpose two types of experiments are performed. The first one is to compare regular text search over the OCR output to the image search. Notice that image search is case-sensitive whereas text search is not, because the image features extracted from upper and lower case letter are different because of the appearance. In order to make the evaluation fair, we only focus on single word search where text search is also case-sensitive. We do not employ any advanced query evaluation techniques for both text and image search, such as query expansion, stemming etc.

Table 7.1 compares OCR text search to our image search framework. The Mean Average Precision (MAP) measure is used for evaluating ranked lists. OCR text search was not successful in retrieving 8% of the true positives. Therefore its MAP is estimated to be 92.3% using an exact-match evaluation strategy. MAP score for

Query image:

వరసింహ

నేలపై నెచ్చోట నెలపు చాలనియట్లు
 కొండపై నిలుకట్టు కొంటివేల
 జనపాలి వసియించు జనపదంబులువీడి
 దుర్గమారణ్యంబు దూరితేల
 మనుజసమాజ సంసర్గ సౌఖ్యముమాని
 అలఘు సత్వచయంపుఁ జెలిమియేల
 కడుతిన్ననై నమార్గము రచించుటఁజాసి
 వక్రశ్లోసోపావపథములేల
 చటులకంఠీరవాకార సహజగుణము
 మానవచ్చునె నీవంటివానికై న
 దనుజసంహార ! మాల్వభూధరవిహార !
 ధర్మసంచార ! **వరసింహ** ! ధైర్యసార !

17

(624)

ఓ **వరసింహ** ముమ్ముద్ధరింపుమటన్న
 ఆరారవములు మిచ్చందిమ్రోయ
 ఆండగా నీవుంటి వ నను విశ్వాసంబు
 మోము దామరలఁ బ్రస్తుటముగాఁగ
 హృదయాంతర విశేష్ట మధురభక్తి ప్రభా
 వమ్ముకై మోక్షులఁ బ్రకటితముగ
 దాసానుదాసుండఁ దమకనుభావంబు
 కచఖండనక్రియాకలనఁ జెలియఁ
 గవ్వవప్తముల్ సైచి సంకటములోర్చి
 వచ్చు భక్తులఁగాపాడి వరములిమ్ము
 దనుజసంహార ! మాల్వభూధరవిహార !
 ధర్మసంచార **వరసింహ** ! ధైర్యసార !

21

వరసింహ

! వరసింహ

Figure 7.4. Example Telugu word images which are correctly retrieved using our methodology.

the image search is better than the regular text search for this particular book even though the OCR accuracy is very high.

Table 7.2 shows the MAP scores for the Telugu books. Since there is no OCR engine for Telugu, we can not compare the image search with OCR text search for these books. However, it is clear from the MAP scores that image search is quite effective in searching Telugu books.

Figure 7.4 shows the returned word images for the query word at the top. Notice that connected component analysis or contour based approaches would fail when word images are underlined or connected by ink. We make use of the sections of letters which are not corrupted by the noise. This information provides strong evidence for

QUERY WORD		
<u>మాల్వభూధరవిహార</u>		
Rank	Retrieved Word Images	Ground Truth
25	<u>మాల్వభూధరవిహార</u>	MATCH
50	<u>మాల్వభూధరవిహార</u>	MATCH
75	<u>మాల్వభూధరవిహార</u>	MATCH
100	<u>మాల్వభూధరవిహార</u>	MATCH
110	<u>మాల్వభూధర</u>	SUBSET MATCH
111	<u>మాల్వాద్రి-కొండయ్య-మాలకొండయ్య</u>	PARTIAL MATCH

Figure 7.5. A ranked list for the long query word shown at the top. There are 109 relevant word images in the book. AP score for this query is 1.0.

QUERY WORD		
<u>నీకు</u>		
Rank	Retrieved Word Images	Ground Truth
5	<u>చీకు</u>	PARTIAL MATCH
10	<u>నీకుం</u>	SUBSET MATCH
18	<u>నీళ్ళకు</u>	SUBSET MATCH
20	<u>నీడద్వారంబునకుం</u>	SUBSET MATCH
21	<u>నీళ్ళు</u>	PARTIAL MATCH
22	<u>నీరుకాకు</u>	SUBSET MATCH

Figure 7.6. A ranked list for the short query word shown at the top. Incorrect matches are shown along with their rank and the matching characters of the image are underlined. AP score for this query is 0.85.

being a match. Figures 7.5 and 7.6 show two other examples for short and long query words.

An efficient word spotting framework is presented for searching noisy document images. The drawback of word spotting systems is that a query word image is necessary for searching the document images. One needs to find an example word image to run the query. In the next section, we devise a dependence model which enables the searching for an arbitrary text query in the document images in real time without the need for an example word image.

7.3 A Discrete MRF Model for Searching Arbitrary Text in Document Images

Our proposed approach adapts the general MRF framework proposed by Metzler and Croft for text retrieval [70]. This general framework has been used for image retrieval as well by Feng and Manmatha [36]. Searching text in document images task is slightly different from the text retrieval problem. An arbitrary text query Q (such as “Sherlock”) and visual features for each word image I in the book are given. The task is to rank all the word images according to their relevance to the query word $P(I|Q)$. In our framework, the query word is decomposed into its letter bigrams q_i and the posterior probabilities $P(I|q_i)$ are estimated efficiently for each word image. These probabilities $P(I|q_i)$ are later combined to estimate $P(I|Q)$. The details of the proposed framework are discussed in the subsections below after a brief overview of the general MRF framework proposed by Metzler and Croft [70].

7.3.1 The general MRF framework

Markov random fields (MRFs) are useful for modeling the joint distribution of a set of random variables. In a nut-shell, a Markov random field is an undirected graph, where nodes represent random variables. Edges between nodes represents

conditional dependencies between random variables. Based on the Markov property, it is assumed that certain random variables are independent of all others. Dependencies are therefore defined between certain groups of random variables. These groups are called “cliques”. For each type of clique c in the graph, a non-negative potential function $\phi_{c;\Lambda}$ is defined. These functions are parameterized by Λ and they are used for estimating joint probabilities.

The ultimate aim is to calculate the posterior probability $P_\Lambda(I|Q)$ for all word images in the collection and then rank them based on their relevance to the query. We follow the derivation defined in [70] for the estimation of the joint probability $P(Q, I)$:

$$P(Q, I) = \frac{1}{Z_\Lambda} \prod_{c \in C(G)} \phi(c; \Lambda) \quad (7.5)$$

where Z_Λ is:

$$Z_\Lambda = \sum_{Q, I} \prod_{c \in C(G)} \phi(c; \Lambda) \quad (7.6)$$

It is computationally expensive to compute Z_Λ since there are a large number of summations. In our case, the problem is to rank word images based on their posterior probabilities $P_\Lambda(I|Q)$, therefore we can ignore constant Z_Λ .

Once the normalizing constant is ignored, estimating posterior probabilities becomes much easier. According to [70, 36], posterior probabilities can be estimated as follows:

$$\begin{aligned} P(I|Q) &= \frac{P_\Lambda(Q, I)}{P_\Lambda(Q)} & (7.7) \\ &\stackrel{rank}{=} \log P_\Lambda(Q, I) - \log P_\Lambda(Q) \\ &\stackrel{rank}{=} \sum_{c \in C(G)} \log \phi(c; \Lambda) \end{aligned}$$

where $\stackrel{rank}{=}$ indicates rank equivalence. It should be noted that the resulting formula turns out to be a sum of logarithm of potential functions over all cliques. The potential

function is often assumed to have the following form:

$$\phi(c; \Lambda) = \exp[\lambda_c f(c)] \tag{7.8}$$

where $f(c)$ is some feature function over clique c , and λ_c is the weight of this particular feature function. Then the ranking function simplifies to:

$$P_\Lambda(I|Q) \stackrel{rank}{=} \sum_{c \in \mathcal{C}(G)} \lambda_c f(c) \tag{7.9}$$

which is a linear function over feature functions and can be computed efficiently. λ_c is a weight factor and it is defined for each clique in the MRF model.

7.3.2 The proposed approach

Our aim is to locate query letter bigrams q_j in all word images and then sort the word images in the book based on their relevance to the query word Q . The existence of letter bigrams is necessary but not sufficient to qualify a word image for being a match. Their order must also be the same as for the query word. Here we devise an MRF model so that both conditions are satisfied.

We assume that all visual terms v_i are independent of each other given the query word Q . One could also define higher order dependencies between a large number of random variables in the MRF model. However training higher order dependencies becomes impractical when the dimensionality of all bigram letter classes (4K for English) and the visterm vocabulary size (4K in our experiments) is considered. Two types of cliques are defined in our model. The first type consists of all pairs between visterms v_i of the word image I and letter bigrams q_j of Q . The second set of cliques include all letter bigram pairs in Q . These cliques are referred to as type vq and type qq cliques respectively.

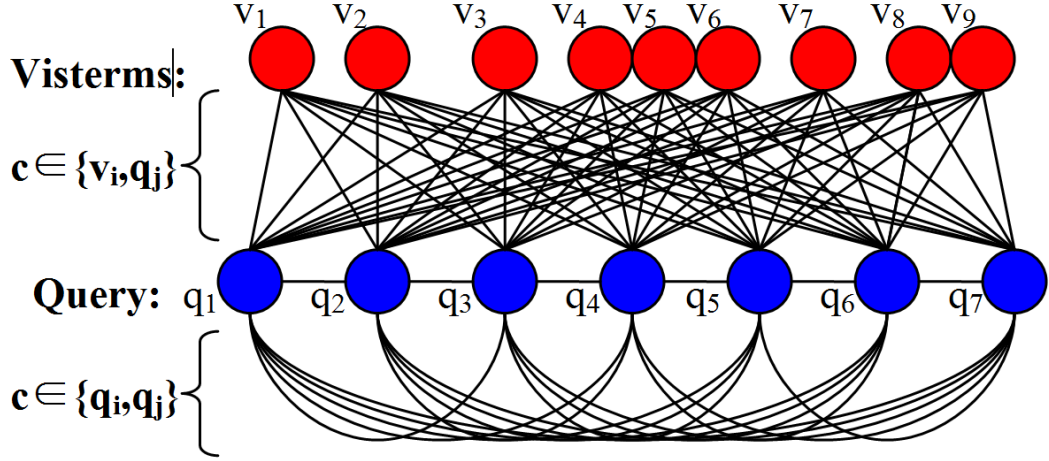


Figure 7.7. The configuration of our MRF model for searching text in document images.

Estimated clique potentials for different types of cliques are later combined into a final MRF score as follows:

$$P_{\Lambda}(I|Q) = \lambda_M NMRF_{vq} + (1 - \lambda_M) MRF_{qq} \quad (7.10)$$

where λ_M is a parameter whose range of values is defined to be $[0,1]$. $NMRF_{vq}$ stands for normalized MRF score for the sum of clique potentials of type vq . Similarly MRF_{qq} stands for the sum of clique potentials of type qq . The estimation procedure for these scores is explained in the following subsections. The configuration of our discrete MRF model is depicted in Fig.7.7.

7.3.2.1 Modeling visterm-letter bigram dependencies

The posterior probability of a word image I given a query word is formulated as follows:

$$MRF_{vq} = P_{\Lambda}(I|Q) = P_{\Lambda}(v_1, v_2, \dots, v_m | q_1, q_2, \dots, q_n) \quad (7.11)$$

where v_i corresponds to visterm i and q_j corresponds to letter bigram j in the query word. According to Eq.7.9, we can rewrite Eq.7.11 as:

$$MRF_{vq} = \sum_{c \in \{v_i, q_j\}} \lambda_c f_{vq}(c) \quad (7.12)$$

where c stands for a clique formed by a visterm v_i in I and a letter bigram q_j in Q . The feature function $f(c)$ is defined to be the posterior probability of q_j given v_i :

$$\begin{aligned} f_{vq}(c) &= \Pr(q_j | v_i) \\ &= \frac{\Pr(v_i | q_j) \Pr(q_j)}{\Pr(v_i)} \end{aligned} \quad (7.13)$$

where \Pr denotes probability distributions.

We are interested in not only the existence but also the location of each letter bigram in the word image. Therefore we need to find the location of each bigram in the word image if it exists. One option is to slide a window over the word image. However, determining window width is problematic since letter bigrams may have different widths for different text fonts. A better option is to slide a Gaussian window. It is possible to incorporate a Gaussian window into our MRF model by varying the values of λ_c as follows:

$$\lambda_c = G_{\mu, \sigma}(x_i) \quad (7.14)$$

where x_i is the height normalized coordinate of visterm i on the X axis of the word image. The Gaussian window is parameterized by μ and σ . The aim is to find a value μ for each q_j so that Eq.7.11 is maximized:

$$MRF_{vq} = \sum_{c \in \{v_i, q_j\}} G_{\mu_{q_j}, \sigma}(x_i) f_{vq}(c) \quad (7.15)$$

and the estimated location of letter bigram q_j in the word image is given by:

$$\mu_{q_j} = \arg \max_{\mu} \sum_{c \in \{v_i, q_j\}} G_{\mu, \sigma}(x_i) f_{vq}(c) \quad (7.16)$$

One problem is that there are some visual terms positioned next to each other in the word image and they have exactly the same visterm ID as shown in Fig. 7.2. Indeed, these visterms are artifacts of keypoint detectors and they do not provide any further evidence for resolving queries. It is not desirable to account for such visterms more than once for scoring. A remedy for this problem is to account for the existence but not the frequency of visterms in word images using a Bernoulli Model. We only account for the visterm whose λ_c weight is the highest for a given μ and a Bernoulli Model is adopted for estimating probabilities as described in Section 7.3.2.3.

Since each visterm class can contribute to the sum at most once and $Pr(q_j|v_i)$ is a distribution over query bigrams, the upper bound for $\sum_{c \in \{v_i, q_j\}} \lambda_c f_{vq}(c)$ becomes $G_{\mu, \sigma}(\mu)$. As a result, the range of values for MRF_{vq} becomes $[0, |Q|G_{\mu, \sigma}(\mu)]$, where σ is a parameter. It is not desirable to have different ranges of values for queries varying in length. Therefore the MRF score is normalized by the query length $|Q|$ as follows:

$$NMRF_{vq} = \frac{1}{|Q|} \sum_{c \in \{v_i, q_j\}} G_{\mu_{q_j}, \sigma}(x_i) f_{vq}(c) \quad (7.17)$$

where $NMRF_{vq}$ corresponds to the normalized MRF_{vq} score.

7.3.2.2 Modeling the order of letter bigrams

The second part of the MRF model accounts for the order of letter bigrams in the word image. The estimated locations of letter bigrams are used μ_{q_j} for determining whether they have the correct order compared to the original query word. More specifically, clique potentials for letter bigram pairs are defined according to Eq.7.9 as follows:

$$MRF_{qq} = \sum_{c \in \{q_i, q_j\}} \lambda_c f_{qq}(c) \quad (7.18)$$

where

$$f_{qq}(c) = \begin{bmatrix} 1 \text{ if the order of } q_i \text{ and } q_j \text{ is correct} \\ 0 \text{ otherwise} \end{bmatrix}$$

$$\lambda_c = \frac{1}{n(n-1)/2} \quad (7.19)$$

and n is the number of letter bigrams in the query word. It is assumed that each query letter bigram and their respective order is equally important. Therefore each clique is equally weighted in a way that MRF_{qq} score's range is set to $[0, 1]$.

7.3.2.3 Probability estimation

The training set C is composed of word images represented by a set of visterms $\{v_1, v_2, \dots, v_m\}$ and a set of letter bigrams $\{q_1, q_2, \dots, q_n\}$ associated with it. Each word image is assumed to contain at least one character and one visterm. For a word with $n - 1$ characters, there are n letter bigrams. One can create a training set synthetically by rendering a large set of word images or individual letter bigrams in various fonts and sizes. Another option is to run an OCR engine on sample document images and use the recognized word image boxes and their text content for training.

Given a training set, the aim is to learn $\Pr(v_i|q_j)$, $\Pr(v_i)$ and $\Pr(q_j)$ in Eq.7.13 for all visterm and letter bigram classes. In this work, it is assumed that $\Pr(q_j)$ is uniform, meaning that each letter bigram class is equiprobable. Similarly, $\Pr(C_k)$ is also assumed to be uniform. In other words, each word image in the collection is equiprobable.

A multiple Bernoulli model is adopted for learning $\Pr(v_i|q_j)$ and $\Pr(v_i)$. According to the model, the existence of visterms in a word image is important, not their respective frequencies. In other words, the probability of $P(v_i|I)$ is estimated using a discrete Kronecker delta function:

$$P(v_i|C_k) = \delta_{v_i, C_k} \quad (7.20)$$

where $\delta_{v_i, C_k} = 1$ if a visterm v_i occurs in the representation of the word image C_k .

Given a training collection C , $P(v_i)$ is calculated by marginalizing v_i over the entire collection C :

$$P(v_i) = \sum_k P(v_i|C_k)P(C_k) \quad (7.21)$$

where C_k is a word image in C . Similarly $P(v_i|q_j)$ is calculated by marginalizing v_i over the set of all word images in C which contain letter bigram q_j :

$$P(v_i|q_j) = \sum_k P(v_i|C_k, q_j)P(C_k|q_j) \quad (7.22)$$

This method is referred to as the **Union Model** in the rest of the paper.

One problem with the union model is that it simply blends all the visterms of training images which contain the letter bigram q_j for learning $P(v_i|q_j)$. However, some visterms in the training image C_k are not associated with q_j in particular. It is desirable to differentiate the visterms which are particular to letter bigram class q_j and use only them for estimating posterior probabilities.

Here we devise another method, which is referred to as the **Intersection Model**, for estimating $P(v_i|q_j)$ which discards visterms which belong to letter bigram classes other than q_j . The idea is to intersect visterms of word image pairs which are known to contain letter bigram class q_j . Visterms in the intersection are meant to be specific to q_j and therefore it is safe to use them for probability estimations. This process is performed for all pairs of word images in J . Formally speaking, $P(v_i|q_j)$ is estimated by marginalizing v_i over all pairs of word images containing q_j :

$$P(v_i|q_j) = \sum_k \sum_l P(v_i|C_k, C_l, q_j)P(C_k, C_l|q_j) \quad (7.23)$$

Assuming that training images C_k and C_l in C are independent of each other, Eq.7.23 becomes:

$$P(v_i|q_j) = \sum_k \sum_l P(v_i|C_k, q_j)P(v_i|C_l, q_j)P(C_k|q_j)P(C_l|q_j) \quad (7.24)$$

It should be noted that the term in the sum is non-zero if and only if both images contain the visterm v_i . It follows from the fact that the term $P(C_k|q_j)P(C_l|q_j)$ is equal to one if q_j occurs in both images C_k and C_l , zero otherwise.

Another advantage of the intersection method is that it discards visterms which occur only once among all instances of word images containing q_j . This is desirable since such visterms are very likely to be products of document noise and/or discretization errors.

Figure 7.8 illustrates the proposed learning models for training the visterm distribution of the query letter bigram $q_j = \text{“th”}$. For simplification purposes, there are only three training instances of word images C_1 , C_2 and C_3 containing the letter bigram “th”. These training images correspond to the words “their”, “another” and “without” respectively. Each training image is also associated with a number of visual terms denoted with v_i . It should be noted that the training images contain visual features for not only the letter bigram class “th” but also others such as “he” and “an”. In this example, each letter bigram is assumed to be directly related to only one visual term and the size of visual vocabulary size is set to 25 for illustration purposes. If a visual term appears in the training image, then the corresponding value $P(v_i|C_k, q_j)$ in the visterm distribution is set to one in the bar graphs shown in Figure 7.8 a), b) and c). Otherwise the value is set to zero. The first training image C_1 contains six visterms whereas the other two images contains eight visterms each.

Figure 7.8 d) shows the distribution of visterms estimated by the Union model. Assuming that each training instance is equally likely, the Union model simply averages the corresponding probabilities for each visterm to learn the visterm distribution for the letter bigram class “th”. Visual term v_{13} appears in all training images and therefore the estimated value for $P(v_{13}|q_j)$ is equal to one. Some visual terms appear

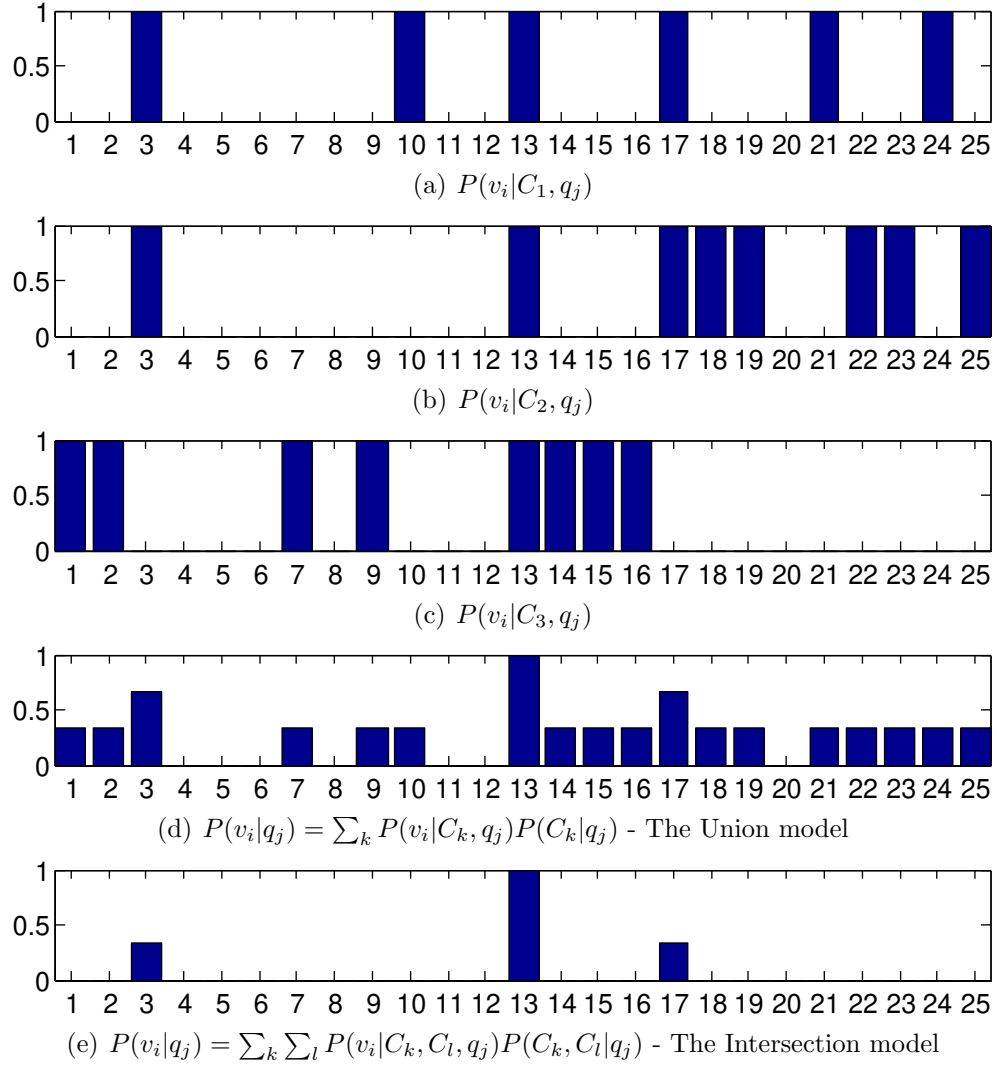


Figure 7.8. The learning models illustrated for learning the probability distributions of visterms for the letter bigram class q_j from three training word images C_1 , C_2 and C_3 . The visterm distribution of visterms for each training sample are shown in a), b) and c). The horizontal and vertical axes represent the visterm IDs v_i and the corresponding probability respectively. Estimated visterm distributions for the letter bigram class q_j are shown using d) the Union and e) the Intersection learning models.

only in a subset of the training samples and their probability values are directly proportional to the number of training instances that contain the corresponding visterms. Figure 7.8 e) shows the visterm distribution estimated by the Intersection model. The intersection model simply iterates over all distinct pairs of training examples and intersects the visterm distributions to eliminate visual features which are not peculiar to the letter bigram class q_j . Once the training instances are assumed to be equally likely, the resulting distribution is simply the average of the visterm distributions of each intersection. As seen in Figure 7.8 e), the intersection model successfully eliminates the visual terms which are not related to the letter bigram class “th” in most cases. In this particular example, the only visual term which is related to the letter bigram class “th” is the visterm v_{13} . Visual terms v_3 and v_{17} obtained non-zero values since those visual terms appear in more than one example in the training images. More precisely, visual terms v_3 and v_{17} correspond to visual features which represent letter bigrams “he” and “r-” which are both common in the training samples “their” and “another”. Therefore visual terms v_3 and v_{17} appear in the visterm distribution after the intersection.

From the example above, it is clear that the selection of training instances plays an important role for estimating the probabilities in the proposed dependence model. In an ideal case, the training instances contain word images which are distinct from each other. Indeed, the training samples should not also have any common letter bigram other than the letter bigram being trained. For example, one should not include the training example “there” in the training set given that there exists a training image containing the word “the”. Notice that these two words have four letter bigrams in common including the space character. This is not desirable since these training instances alone do not help distinguish the visual terms specific to the letter bigram class “th”.

Assuming that the training instances are equally likely and independent from each other, the probability estimation for the Intersection model can be simplified as:

$$P(v_i|q_j) = \frac{\binom{f_i}{2}}{\binom{n_j}{2}} \quad (7.25)$$

where f_i is the total number of images containing the visterm v_i and letter bigram q_j , and, n_j corresponds to the total number training of images containing letter bigram q_j . This implies that there is no need to intersect the visterm distributions for all pairs of training images explicitly if the training images are assumed to be independent and identically distributed. The simplified Intersection model has a linear time complexity, the same as the Union model, since the frequency values f_i can be computed by iterating over the set of training images at once.

One last problem is that there may not be enough training instances to train visterm distributions for some letter bigram classes. It is desirable to estimate those probabilities using a smoothing technique. More specifically, the estimated probabilities are first normalized,

$$\Pr(v_i|q_j) = \frac{P(v_i|q_j)}{\sum_i P(v_i|q_j)} \quad (7.26)$$

$$\Pr(v_i) = \sum_j P(v_i|q_j)P(q_j) \quad (7.27)$$

and smoothed,

$$\tilde{\Pr}(v_i|q_j) = \lambda_S \Pr(v_i|q_j) + (1 - \lambda_S) \Pr(v_i) \quad (7.28)$$

where λ_M is a parameter whose range is $[0,1]$ and $\tilde{\Pr}(v_i|q_j)$ denotes smoothed probabilities. It should be noted that query terms typically correspond to words which appear rarely in the context such as names and places. It is quite likely that the query terms includes letter bigrams which are also rare in the text. It is not desirable to give a lower weight to the features belonging to rare letter bigrams in probability estimations because of their lower prior probabilities. As a solution to this, $P(q_j)$ is assumed to be uniformly distributed in Equation 7.27.

7.3.2.4 Indexing letter bigrams

The final MRF score $P_{\Lambda}(I|Q)$ uses the likelihood of the query letter bigrams and their respective positions in the corresponding word image. It is computationally expensive if those likelihood values are computed during query time. Our approach is to calculate those values only once for all letter bigrams along with their positions in all test images and use these values for resolving the queries instantly. The likelihood values for all letter bigrams and their respective positions are stored in data matrices of size $n \times m$, where n and m refer to the number of test images and letter bigrams respectively. Given a text query, the letter bigram likelihood values are simply looked up for calculating the final MRF score for each test image. One problem with this approach is that the size of these matrices can get very large for large document collections. For a book written in English, the memory overhead is expected to be around 3GB since a typical book contains about 100K words and there are four thousand letter bigrams in total including numeric characters. If memory is at a premium, then it is possible to compress these matrices using an inverted index. This is achieved by applying a threshold on the probabilities to disregard letter bigrams which are unlikely to exist in the word images. The list of boxes relevant to each letter bigram are then stored in the posting lists of the inverted index along with the letter bigram positions in the word images. For simplification purposes, no compression scheme is applied to the data matrices in the experiments. Experiments show that precomputation of likelihood values for letter bigrams provides real time query resolution performance.

7.3.3 Query Resolution

Given a single word text query, the proposed framework is capable of ranking word images in the document images according to the visual similarity. Notice that image features extracted from upper and lower case letters are different because of

their appearance. Therefore image search is case-sensitive whereas text search is not. For compensation, one can run a number of queries in parallel for each possible capitalization of the query word and fuse those rankings. One can also use language specific tools to improve the query performance as discussed in the subsections.

7.3.3.1 Morphological expansion

One approach for improving the search effectiveness is to use morphological variations of the query word. These variations include different forms of the word, such as nouns, verbs, adverbs, adjectives, plural/singular etc. For example; “walks”, “walking”, “walker” and “walked” are among the morphological variations of the query word “walk”. Two scenarios are investigated. The first scenario uses only the given query word for ranking word images. The second scenario uses all the morphological forms of the query word and the matching scores for each word image are averaged. In both scenarios, all the morphological forms of the word are considered to be positive in the ranked list. Mean Average Precision (MAP) is used for the evaluation of search results. It is seen that morphological expansion of the query word provided a lower MAP score (0.91 versus 0.982 for the query “walk”) compared to the case where only the query word is used for querying. For morphological expansion, false positives not only include the semantically related words but also includes words which are visually similar but meaning wise different such as “talk”, “talker”, “talked”, “weaver” and “wall” for the query word “walk”. If the word “walk” is used as the only query word, then many fewer words are false positives such as “wall” and “talk”. Semantic expansion of the query word is not of help in determining visual similarity, or vice versa.

For searching irregular words, use of morphological expansion gets even more complicated. For example, a large number of words including “pound”, “bound”, “bounded”, “abound”, “founder”, “profound”, “profoundly”, “sound”, “wound”, “round”

Table 7.3. Comparison of the methods for the query result evaluation. It is seen that morphological expansion underestimates the MAP score.

Evaluation Method	MAP
Exact match	0.91
Morphological match	0.71

obtain high rank for the query word “find”. The reason is that its past participle form, “found”, is visually similar to those words. For the particular query word “find”, morphological expansion provides 0.09 MAP score whereas the original query word has a score of 0.91. Morphological expansion is not of help with foreign words either, such as names and places.

Although morphological expansion does not seem to be promising for improving the search effectiveness, it retains its potential for automatic evaluation of the query results. Given a single query word and the corresponding ranked list of word images, the problem is to calculate a score for the search effectiveness (in our case, MAP). If the retrieved word image includes a morphological variation of the query word (having the same meaning), then it is determined to be a positive match. This approach may work reasonably well for querying regular words (such as “murder” etc). However, it does not work for querying irregular words such as “find”, “see” etc. The reason is that irregular forms of the query word (“found” and “saw”) do not necessarily have enough visual similarity to the query word in order to have them ranked higher in the list. They obtain relatively lower rank in the ranked list but are still considered to be a true positive. This is not desirable since lower MAP scores are reported although all the positive examples of the actual query “find” may be matched correctly. The disparity between the reported results is seen in Table 7.3 for a query set of size ten.

One solution to the disparity problem is to use only the morphological variations of the query word which also exhibit visual resemblance. For example, using “finder” and “finding” in the expansion but not “found”. However, this would not solve the problem either. The reason is that failure in finding the morphological variations

should not penalize the search effectiveness of the actual query word. For example, word images containing the word “wall” obtain higher scores compared to “walked”, “walker” or “walking” for the query word “walk”. This again causes the MAP score to seem lower than it actually is.

7.3.3.2 Stemming

Stemming is another way to retrieve morphological variations of the query word. The basic idea is to query the stem of the query word. For example, “experi” is the stem of the word “experiment” and all the words which includes the bigrams of the stem are retrieved. There are two main problems with this approach. First, there may be multiple words having the same stem but different meaning. In the case of querying “experiment”, the word “experience” is also retrieved at the top of the ranked list although it is both semantically and lexically irrelevant to the query. The second problem is that, stems usually have a small number of letters compared to the actual form of the query word. It is therefore very likely to confuse irrelevant words which have the same bigrams in the same order as the stem word. For example, “act” is the stem word for query “actor”. The stem is included in some other words such as “attractive” and “reaction” which have no semantic and relationship to the actual query.

For simplification purposes, we do not use either morphological expansion or stemming in this context. These approaches are language specific and they have their own complications. Most importantly, semantic or visual similarities between the words do not necessarily imply the one or the other.

7.3.4 Experiments

The aim is to investigate the effectiveness of our image search engine given a particular text query. In order to make the evaluation more fair, we only focus on single-word search. The OCR text search baseline is also case-sensitive. Punctuation

is ignored at all stages. For simplification purposes, we do not employ any advanced query evaluation techniques for both text and image search, such as query expansion, stemming etc. In this way the evaluation becomes independent of the language of the book.

The effectiveness of the proposed approach is evaluated using printed books written in three different scripts: Latin, Telugu and Ottoman. The books printed in Latin and Telugu script are written in English and Telugu (an Indian language) respectively. The Ottoman books are written in a language called Ottoman which is a language mix of Arabic, Persian and Turkish. The Latin alphabet includes a fixed set of characters which do not change their shape based on their context in the text. Therefore texts printed in Latin script are relatively easy to recognize and there are several high accuracy commercial OCR engines available for this purpose. In the case of Telugu and Ottoman, there is no commercial OCR engine available due to their complexities as discussed in the following subsection.

The effectiveness of the proposed approach is first shown for searching text in document images printed in Latin script. The dependences between the visual terms and letter bigrams are automatically trained using noisy OCR output. It is demonstrated that OCR text search accuracy can be significantly improved if it is combined with the proposed word image search based approach. Telugu and Ottoman experiments further demonstrate the effectiveness of the proposed approach for searching text in noisy document images printed in more complex scripts for which there is no commercial OCR engine available. Detailed information about the datasets, training the proposed model and evaluations are given in the following subsections.

7.3.4.1 Datasets

7.3.4.1.1 The Latin Dataset : The Latin experiments consists of two publicly available books printed in Latin script. The book “Adventures of Sherlock Holmes” by

Table 7.4. Frequency distribution of the words in the Latin, Telugu and Ottoman books after ignoring punctuation.

Dataset	Total #words	Vocab. Size	#words per frequency			
			1	2	3	≥ 4
LATIN-S.H.	103375	9080	4552	1408	713	2407
LATIN-W.H.	119275	10530	5025	1701	904	2900
TELUGU-1716	21142	12752	10556	1248	356	592
TELUGU-1718	4294	2951	2812	89	14	36
OTTO-1	9879	4997	3785	653	205	354
OTTO-2	3548	2498	2064	275	88	71

Arthur Conan Doyle is used for training the parameters of the proposed model. This is the same book used for the experiments in the previous section. The test book is titled “Wuthering Heights” by Emily Brontë and it contains 299 pages in total. The ground truths are automatically generated by aligning the main text (downloaded from the Project Gutenberg website) with the corresponding OCR text output using the Recursive Text Alignment Scheme. In the case of the test book, the text alignment automatically annotated 286 scanned page images which correspond to the main text. The estimated OCR word and character accuracy values for the main text are 88.67% and 97.01% respectively. The first 236 pages and the corresponding OCR text output are used for training the visual vocabulary and the visterms distributions. The last 50 pages of the main text are used for evaluation purposes. Estimated OCR word accuracy is 89.35% for this portion of the text. All the words in the vocabulary of the last portion of the book are used for querying. The query test set contains 3898 words in total. Detailed word frequency statistics for the Latin books are given in Table 7.4 after removing the pages which could not be automatically annotated because of scanning errors such as duplicated and missing pages. Notice that approximately half of the words in the vocabulary of each book appears only once in the context.

In the case of Latin script, letters are typically composed of straight ink pieces and/or round curves. As a result, the corner detector locates relatively fewer number of corner points. A dense sampling approach is therefore adopted to address the

sparse keypoint problem. More specifically, the page images (12 megapixels) are first downsampled by a scaling factor (0.27). All the ink (foreground) pixels in the downsampled image are regarded as keypoints. The image patches are placed at the corresponding positions in the original page image and the features are extracted as described in Section 7.1.

7.3.4.1.2 The Telugu Dataset : Telugu is a widely spoken language in India (>80 million people) and it has its own script. The Telugu script is similar to other Indian scripts in various ways. As in most Indian scripts, most characters are composed of more than one connected component. The primary complexity of the Telugu script is the spatial distribution of the connected components that make up the characters. Although the individual characters are lined up from left to right, the connected components of a particular character might be positioned not only in the horizontal order, but also they might be above, below or even inside other connected components. Another complexity is that a word in Telugu might have slightly different pronunciation and appearance in different contexts although the semantics of the word is the same. Due to the complexities of this script, the character recognition accuracies are typically quite low [78, 39]. There is no commercial OCR engine available for recognizing characters in Telugu script.

The Telugu experiments consist of two publicly available books printed in Telugu script [121]. These books were annotated manually using an ASCII coding scheme. Each character is encoded by at least one but typically multiple ASCII characters. Therefore the mapping between each character glyph and the ASCII characters are not one to one. This type of annotation is actually not desirable for training the proposed model. It violates the assumption of one-to-one mapping between each character class and their corresponding visuals in the word images. However, the experiments demonstrate that the proposed model tolerates one-to-many and many-to-one character mappings as well. Figure 7.14 shows some example word images

written in Telugu along with their ASCII encodings. The word frequency statistics of the Telugu books are given in Table 7.4. The books are named Telugu-1716 and Telugu 1718 and they are used for training and testing the proposed model respectively. These books contain a total of 21142 and 4294 words respectively. Notice that the majority of the words in these books (82.7% and 95.3% of the vocabulary words, respectively) appear only once in their respective context. This makes the conventional word spotting approaches not applicable for searching text in these books. Word spotting approaches need at least one word image example to search for other instances of the query word using visual similarities. All the words in the vocabulary of the test book are used for evaluation purposes.

7.3.4.1.3 The Ottoman Dataset :

The Ottoman script is quite similar to the Arabic script with some additional characters and missing diacritics. A publicly available Ottoman dataset is used for evaluation purposes. It consists of 100 document images (300 dpi - binary images) scanned from two different books teaching Ottoman script. [117]. Each book contains a number of short readings written in Ottoman language (a mix of Arabic, Farsi and Turkish). Each reading published in these books is originally scanned from different sources and the font type and/or the size of text therefore varies for each article. The articles scanned from the first book contain 60 document images and they are used for training the proposed dependence model. The rest of the document images are used for testing purposes. Table 7.4 shows word frequency statistics for both sets.

The training and test sets (OTTO-1 and OTTO-2) consist of 9879 and 3548 word images respectively. As in the case of Telugu, most words appear only once in the respective context for both sets. More specifically, 75.7% and 82.6% of the words in the vocabulary appears only once in the training and test sets respectively. Word spotting techniques are therefore not applicable for searching text in these collections as well.

صلح صورت قطعيه ده تکرار
اوزره آنطالیاده ایتالیان مثلکنه
کتبخانه لر مفتش
فاتحک استانبولی فتح ایتدیکی
بجر سفید وچناق قلعه ده کی

Figure 7.9. Example text lines from the Ottoman dataset. The Ottoman script is quite similar to the Arabic script with some additional letters and missing diacritics.

The ground truth contains locations of each connected component in the document images along with the associated letter symbols. Specifically, there are 48 integer coded ink shapes which are used for annotating the connected components. The entire dataset contains a total of 70K shape-coded characters. Line and word boundaries are used for word annotations and they are determined using projection profiles. The best recognition accuracy reported for this Ottoman dataset is 93% [117] and there is no commercial OCR engine available for Ottoman script. This dataset is the most challenging one among the others because the font type, the font size and the document noise vary across different articles. Figure 7.9 shows example lines from the Ottoman dataset.

7.3.4.2 Training

The first step is to train a visual vocabulary from the scanned page images and it does not require any labeled data. The visual vocabulary is trained simply by selecting a number of document images at random and clustering the extracted visual features using the Hierarchical K-means algorithm, as described in Section 7.1. The

vocabulary tree is configured so that it is two levels deep with a branching factor of 64 (results in 4096 visual terms in total). It is used for quantizing the feature vectors in the document images.

The second step is to estimate the visual term distributions of all letter bigram classes. In the case of English, there are $N = 63 \times 63 = 3969$ letter bigram classes including numbers, upper and lowercase letters, and the space character. The Ottoman dataset includes 48 shape codes primitives and there are $49 \times 49 = 2401$ letter bigrams including the space character. The Telugu dataset is annotated using the lower case letters of the English alphabet which yields $27 \times 27 = 729$ letter bigram classes in total. One way to estimate the prior $P(v_i)$ and posterior $P(v_i|q_j)$ probabilities is to use scanned page images with annotated word bounding boxes. However, it is not possible to estimate visterm distributions for all letter bigram classes. It is observed that only about 300 of the 4K letter bigram classes have more than 20 occurrences in a single book. Most of those classes correspond to the most frequent lowercase letter bigrams in English. These letter bigrams are sufficient to generate the majority of words in English language. Another option is to estimate visual distributions using synthetic word images. However, learning image features from synthetic images has its own challenges. It is not easy to model document noise and different fonts [51]. Experiments with synthetic word images performed much worse and therefore it is not discussed further.

Figure 7.10 shows the number of distinct letter bigrams whose frequency is greater than 20 as a function of length of the text. 200 noisy-free English books from the Project Gutenberg website are combined into a single text with a total of 21 million words. It is clearly seen that the total number of distinct bigrams increases with a falling rate as the length of the text increases. This actually follows from Zipf's law that the rank of letter bigrams is inversely proportional to their frequencies. Notice also that only 1778 out of 3969 letter bigrams are learned from a text with 21 million

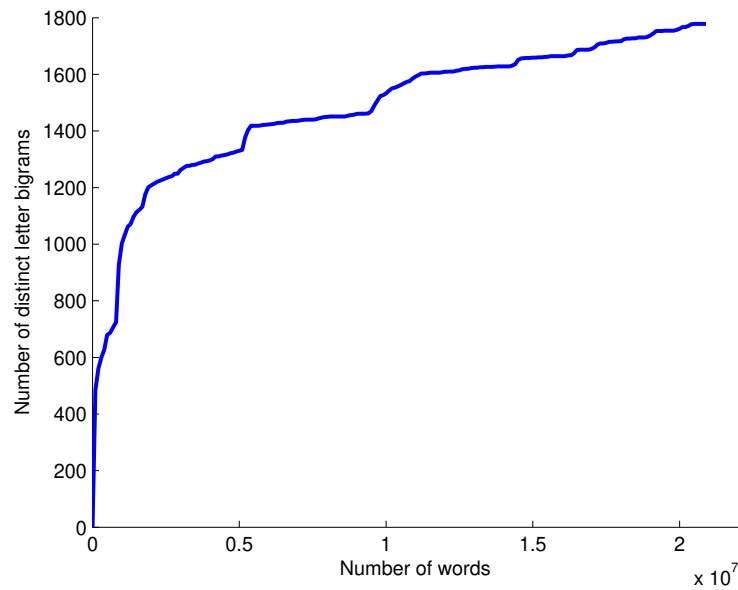


Figure 7.10. The number of distinct letter bigrams as a function of the length of the text.

words whereas 1013 letter bigrams are never observed. Letter bigrams which were not observed in the text were the letter pairs which rarely occur such as “zX”, or, the ones which contain one letter and one numeric character such as “9w”.

For a given query word, it is sometimes not necessary to have visual features for all of its letter bigrams. In most cases it is sufficient to have a number of bigrams for which the visual features are known. Typically visual features for the first and the last letter bigrams of the query word are known. For example, letter bigrams (space,H) and (s,space) are known for the query word “Holmes”. Using only those two bigrams, the majority of words in the vocabulary of the text can be reliably filtered out for being a match. The reason is that there are relatively a small number of words which start with the letter ‘H’ and end with the letter ‘s’ in the vocabulary of the whole book. The more letter bigrams are involved in the search process, the more precise the retrieval becomes.

Table 7.5. The training and test sets used for evaluation purposes (Latin, Telugu and Ottoman scripts).

Script	Training visual vocabulary	Training visterm distributions	Training model parameters	Test set
Latin	W.H. (1st portion)	W.H. (1st portion)	S.H.	W.H. (2nd portion)
Telugu	TELUGU-1716	TELUGU-1716	TELUGU-1716	TELUGU-1718
Ottoman	OTTO-1	OTTO-1	OTTO-1	OTTO-2

Table 7.5 summarizes the training and test sets used in the experiments. In the case of Latin script, the MRF model parameters are learned from another book (Sherlock Holmes, S.H.) and applied on Wuthering Heights (W. H.). The OCR output of the test collection itself is used for training visual vocabulary and visterm distributions. Notice that the training process for Latin is fully automatic and there is no need for annotated data for searching texts printed in Latin script. In the case of Telugu and Ottoman there is no OCR output available. In these cases, an annotated book is therefore necessary for training the model parameters, visual vocabulary and visterm distributions. It should be noted that, the proposed approach can search for arbitrary text in document images unlike the word spotting approaches. Therefore all the words in the vocabulary of the test set is used for evaluation purposes.

In the last step, the main parameters of the proposed MRF model λ_M and λ_S are estimated using the labeled set of word images. The overall effectiveness of the proposed approach is not quite sensitive to the Gaussian parameter σ and therefore it is set to 0.5 in all experiments for simplification purposes. Yet another parameter is the sliding interval for the Gaussian window. Smaller intervals yields better results however processing time may get very large. The point is to ensure that we do not skip over any letter while sliding the window. Therefore this value is set to 0.5 times the height of the word bounding box. In the ideal case, it corresponds to $2n - 1$ intervals for $n - 1$ letters in a word image.

7.3.4.3 Latin script experiments

The OCR text output and corresponding word bounding boxes of the training book “Sherlock Holmes” are used for training the model parameters λ_M and λ_S . Two hundred query words are randomly selected from the vocabulary of the training book to determine the parameters which maximizes the Mean Average Precision (MAP) score. Estimated model parameters are later used for the test book.

The first 236 pages of the test book “Wuthering heights” are used for learning the visual vocabulary and visterm distributions for each letter bigram. The OCR output of the test book is used as the ground truth for this purpose. It should be noted that the OCR output is noisy. Estimated OCR letter bigram accuracy is 94.09% for the test book. All the words that appear at least once in the last 50 pages of the book are used for querying. There are 3898 query words in total.

Ukkonen’s q-gram distance measure [106] is adopted as the OCR text search baseline. This approach has been previously shown to be effective for searching OCR degraded texts [42]. In a nut-shell, the q-gram distance approach uses the letter bigrams to represent the input strings in the vector space. The distance between the two words are defined by the Manhattan distance between the q-gram vectors. The resulting score is a discrete number with a range $[0, n + m]$, where n and m are the number of letter bigrams in the input words respectively. If the input words are similar, then the q-gram distance measure is expected to be smaller. In our case, each word bounding box is associated with its OCR text output. The OCR output is used to rank all the word images according to the q-gram distance score to the query word.

In this work, the “normalized q-gram similarity score” is introduced for combining the q-gram distance score with the image search score:

$$S_q(x, y) = 1 - \frac{D_q(x, y)}{|x| + |y|} \quad (7.29)$$

Table 7.6. MAP scores results for resolving arbitrary query in the test book titled “Wuthering Heights”.

Search method	Learning Model	λ_M	λ_S	λ_C	MAP
OCR text search	-	-	-	-	0.930
image search	union	1.0	0.01	-	0.497
image search	union	0.53	0.01	-	0.817
image search	intersection	1.0	0.01	-	0.792
image search	intersection	0.19	0.01	-	0.854
combined	union	0.53	0.01	0.47	0.957
combined	intersection	0.19	0.01	0.50	0.958

where $|x|$ and $|y|$ corresponds to the total number of letter bigrams in the two input words respectively. The normalized q-gram similarity score has a range $[0, 1]$ and it produces a higher score if the two words are similar. It is equal to one if the input words are identical. The normalized q-gram similarity score is linearly combined with the image search score:

$$C(I, Q) = \lambda_C \times P_\Lambda(I|Q) + (1 - \lambda_C) \times S_q(I_{OCR}, Q) \quad (7.30)$$

where Q is the text query, I_{OCR} corresponds to the OCR output for the word image I and λ_C is the parameter used for combining the normalized q-gram similarity and the word image relevance scores. The word images are finally sorted in descending order of their combined scores $C(I, Q)$. λ_C parameter has a range of $[0, 1]$ and is determined using the training book.

Table 7.6 shows the retrieval scores of the proposed and baseline approaches for the test book “Wuthering Heights”. The OCR text search baseline provides a MAP score of 0.930. The proposed dependence model is trained using the noisy OCR output and it produces relatively lower retrieval scores compared to the OCR text search baseline. The corresponding MAP scores are 0.854 and 0.817 using the Intersection and Union dependence learning models, respectively. The retrieval scores without the letter bigram positional dependences ($\lambda_M = 1.0$) are relatively lower for both Intersection

and Union models. This indicates that the letter bigram sequence information is useful for retrieving relevant word images. Combining the OCR text search baseline with the proposed dependence model provides the highest retrieval scores - 0.958 and 0.957 respectively. This indicates that, even if the OCR (and consequently the OCR text search baseline) fails in certain cases, the visual features trained from the correctly recognized words can be effectively used to retrieve the misrecognized word images. It is demonstrated that the global font feature is useful for effectively training the visual appearances of letter bigrams and improving the text search in noisy document images.

Additional experiments are carried out to investigate the effects of font differences in the training and test phases. For this purpose, the vocabulary tree, visterm distributions and model parameters are entirely learned from the training book Sherlock Holmes and applied to the test book Wuterhing heights. Even though the font types of the two books are relatively similar, the document noise inherent in both in the scanned pages and the OCR output are different. The test book “Wuthering Heights” has much lower OCR accuracy and it contains a large number of letters connected due to ink deformations and binarization artifacts which causes OCR errors. The retrieval scores were therefore significantly lower than the scenario where the book itself is used for training purposes and it is not discussed further. It should be noted that different books are printed at different times and places with different equipment. The physical conditions, the scanning quality and other pre and post processing steps defines the image noise inherent in the document images. Each book’s document noise is different and unique to itself, even if they are printed in the same font type. Therefore it is desirable to learn the visual models from the book itself for which the text search is performed.

The impact of query word length in the search effectiveness is investigated in Figure 7.11. It is clear that MAP scores increase as the query words get longer.

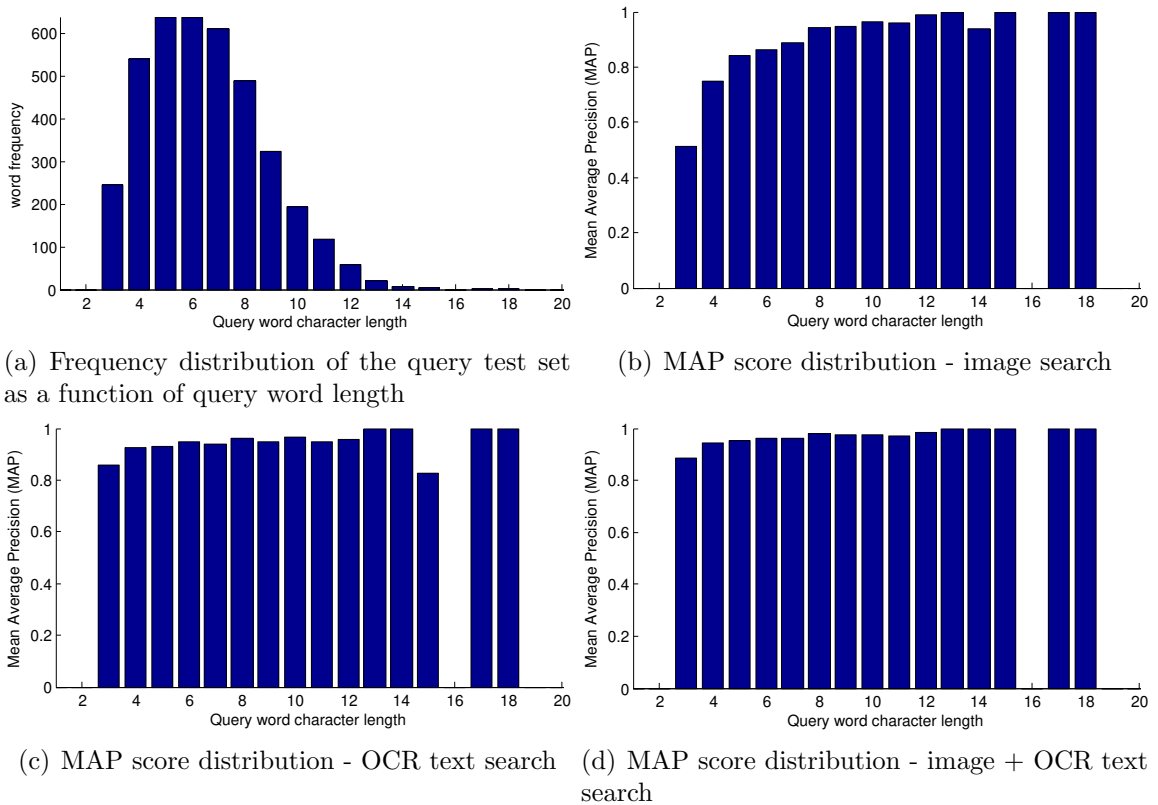


Figure 7.11. Distribution of query words as a function of their length is given in a). MAP score distribution as a function of the query word length is given for b) the proposed approach (Intersection model), c) OCR text search baseline and d) the two approaches combined.

This effect is more evident for the image search approach. The lowest MAP scores are obtained for query words which include only three letters. In the case of image search, the query word “the” is commonly confused with “there” and “therefore” since these words include all the letter bigrams of the query word exactly in the same order. It should be noted that the proposed approach only accounts for the existence and the global order of letter bigrams in the word image. A test image therefore obtains a high matching score if it subsumes the letter bigrams of the query word and the letter bigrams follow the same order with the query word.

Figure 7.12 shows example word images for a number of query words. The word images “scoundrel”, “movements” and “worship” were not retrieved by the OCR text

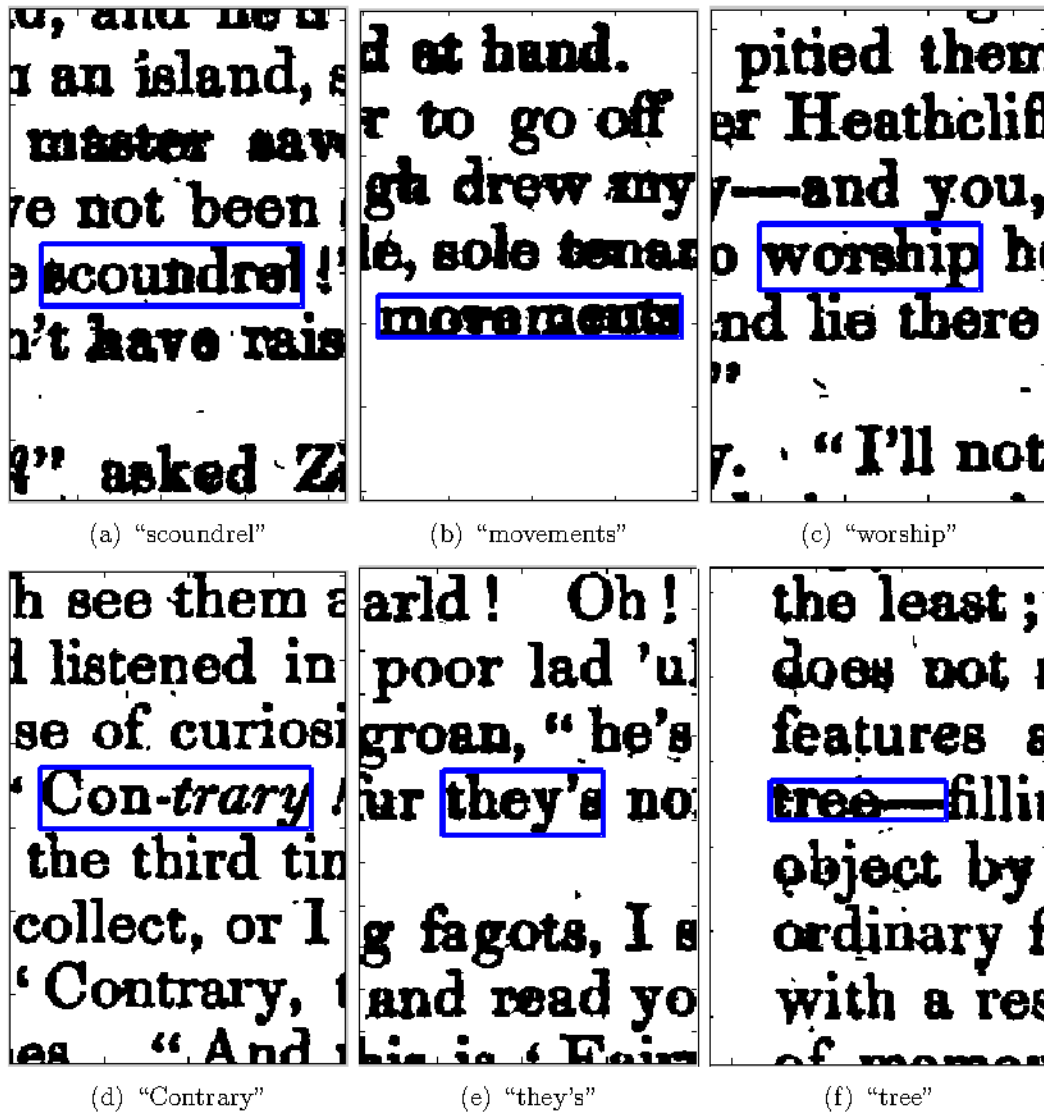


Figure 7.12. a), b) and c) shows example word images which are correctly retrieved by the proposed dependence framework but missed by the OCR text search baseline because of OCR errors. d), e) and f) shows word images which are correctly recognized by the OCR engine and retrieved by the text search baseline. These word image images were missed by the proposed approach. The combined approach (image + OCR text search) correctly retrieved all these word images.

search baseline because of OCR errors. The corresponding OCR text outputs for the word images are “scpimdrei”, “mofemems” and “wcnnip”, respectively. However, these word images were successfully retrieved by the image search approach with an AP score 1.0. The example word images “Contrary”, “they’s” and “tree” were correctly recognized by the OCR engine and therefore retrieved by the OCR text search baseline approach. The image search approach failed in those cases. In the case of the query “Contrary”, it turns out that the italic form of the word is visually dissimilar to the other instances of the query word. Notice that the image search approach accounts only for visual similarities to retrieve matching word images. In the case of the query word “they’s”, the punctuation letter changes the visual appearance of all the image patches in and around the letters “y” and “s”. As a result the word image instances of “the” and “there” were ranked at the top of the ranked list. It should be that the proposed approach removes all punctuation letters at all stages and therefore does not recognize punctuation letters as letter classes. This type of errors could be potentially avoided if the punctuation letters were also regarded as a letter class along with other alphanumeric characters. In the last query word example “tree”, the visual features for the query letter bigram (e,space) are disturbed by the existence of the long dash character. The example word image was therefore ranked much lower in the ranked list. The combined approach (OCR text baseline + image search) were able to retrieve all these six word images correctly along with other true positive instances at the top of the ranked list. Overall, both OCR text search baseline and the combined approach provide the same AP scores for 3220 out of 3898 test query words in total. The combined approach provides better AP results for 573 queries out of the remaining 678 queries. The corresponding MAP scores for the test query set were 0.93 and 0.958 for the OCR text search and the combined approaches, respectively.

Table 7.7. Experimental results for the Telugu dataset.

Training Model	Dataset	λ_S	λ_M	MAP
Intersection	TELUGU-1716	0.002	0.46	0.563
	TELUGU-1718	0.002	0.46	0.562
Union	TELUGU-1716	0.016	0.65	0.488
	TELUGU-1718	0.016	0.65	0.436

7.3.4.4 Telugu script experiments

The model parameters, the visual vocabulary and visterm distributions for each letter bigram are trained on the training set (TELUGU-1716) and directly applied on the test set. The MAP scores produced by the proposed approach using the Union and Intersection training models are shown for the training and test sets in Table 7.7. On the test set (TELUGU-1718), Intersection model provides a MAP score of 0.562 which outperforms the Union model (0.436) on the same set with a very large margin. The training and test set’s MAP scores are quite close to each other in the case of the Intersection model. This makes the Intersection model preferable since it generalizes better on the test set. The smoothing factor λ_S is seen to be quite small for both training models. This implies that the training instances were sufficiently informative for learning the visterm distributions of the letter bigram classes appearing in the query terms. It should be noted that the training set includes only about 21K annotated words. The Intersection model’s MRF parameter λ_M is relatively smaller than the corresponding value for the Union model. This indicates that the Intersection model weighs the letter bigram positional information MRF_{qq} more than the bigram existence score $NMRF_{vq}$.

In the case of TELUGU-1718 test set, a MAP score of 0.562 score actually implies that the relevant word images are typically found at the top two or three positions in the rank list. This follows from the fact that over 95% of the query words appear only once in the test book TELUGU-1718 (Table 7.4). For a given query word with a single relevant word image, if the search term appears in the first position of the ranked list,

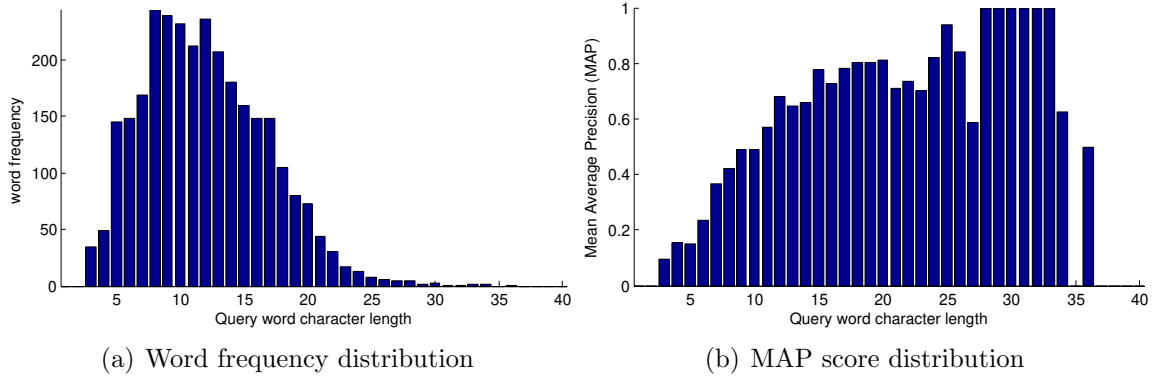


Figure 7.13. Intersection model’s MAP score distribution on the TELUGU-1718 test set as a function of the query word length.

then the AP score is equal to 1.0. Otherwise, if it appears in the second or third rank, then the AP score drastically reduces to 0.50 and 0.33 respectively. Since the MAP score is 0.562 for the Telugu test set, the expected rank for the relevant word image in the list is approximately two. It should be noted that there are also a significant amount of annotation errors in the Telugu books as discussed later. Therefore actual MAP scores are expected to be higher than 0.56.

In order to leverage the overall recognition accuracy, OCR engines typically use predefined dictionaries and/or language models trained for each language. OCR is therefore generally good at recognizing frequently occurring words such as “the”, “and”. However, OCR tend to be error prone for recognizing rare words such as names and places (especially if they are long) which do not appear in the dictionary or the language. In the case of word spotting, the user must be able to spot an instance of the query word image in the documents in order to perform the query. This type of search is therefore more suitable for searching frequent words. The terms which appear only once can not be searched in this way. Unlike OCR and word spotting approaches, the proposed approach is able to find long and rare query terms effectively without requiring any dictionary or explicit language model. As the query word gets longer, the retrieval accuracy increases as well as shown in Figure

7.13. The words in the vocabulary of the test book are grouped by their length (total character count) and a MAP score score is calculated for each group. The frequency of query words and the corresponding MAP values are given respectively for each group. It is clear that longer queries have higher MAP scores. This type of behavior is desirable since the proposed approach better responds to the type of queries which appear frequently in practice.

Figure 7.14 shows three examples of searching text in the Telugu test set using the proposed approach. The query words are shown at the top of each subfigure. The retrieved word images are listed according to their ranks and the corresponding ground truth label is given under each word image. If the retrieved word image is relevant to the query, then it is labeled with green (light circle), otherwise with red (dark circle). The rank of each word image in the ranked list is also indicated by the column on the left.

The first example query word is “maalyabhuudharavihaara” and characterized by a large number of characters (Figure 7.14a)). It should be noted that the query is also formulated using the same way the word image annotations are performed. Notice that there are 22 characters in the text query whereas there exist only nine character glyphs in the corresponding word image. The ranked list of word images include all the 109 examples of the query word in the test set without any false positive. This indicates the effectiveness of the proposed approach in the cases where one-to-one mapping exist between the characters in the annotation and the character glyphs in the word image According to the ground truth, the AP score for this query is 0.756 although manual evaluation of the ranked list yields an AP score of 1.0. It turns out that annotation errors and inconsistencies widely appear in both Telugu books. Over a number of examples, it is seen that approximately 10% of the words have annotation errors. This indicates that, despite the noisy annotations, the proposed

Query = "maalyabhudharavihaara"	
Rank	Retrieved Image
1	● మాల్వభూధరవిహార 'maalyabhudharavihaara'
3	● మాల్వభూధరవిహార 'narsinha'
6	● మాల్వభూధరవిహార 'aalyabhudharavihaara'
12	● మాల్వభూధరవిహార 'maalyadharabhudharavahaara'
103	● మాల్వభూధరవిహార 'maalyadharavihaara'
110	● మాల్వభూధర 'maalyabhudhara'
111	● మాల్వభూమిద్రమవికుంఠ 'maalyabhuumidhramevikuntha'

(a) Long query example

Query = "narsinha"	
Rank	Retrieved Image
1	● నరసింహ 'narsinha'
15	● నరసింహ 'narsinha'
75	● నరసింహలు 'narsinhu'
88	● సధివసింప 'sadhivasinpa'
92	● నరసింహ 'narsinha'
128	● నరసింహ 'narsinha'
182	● సర్వంసహా 'sarvansaha'

(b) Retrieving noisy word images

Query = "niiku"	
Rank	Retrieved Image
1	● నీకు 'niiku'
11	● వీరరసావతారుడనుచు 'viirarasaavataarudanuchu'
13	● నీచతకును 'niichatakunu'
14	● నీకు 'niiki'
15	● నీవధికతముడ 'niivadhikatamuda'
16	● ఆ సికులనుజూడ 'astikulanujuuda'
17	● నీభక్తు 'niibhaktu'

(c) Short query example

Figure 7.14. Telugu word images successfully retrieved by the proposed model.

model can effectively learn the dependences between the visual terms and character letter bigrams.

The second example is medium length query word: “narasinha”, for which there are noisy instances of relevant word images. According to ground truth, there are 113 samples of the query word and the AP score is 0.774. Manual evaluation of the query result indicates that actually there are 120 examples of the query word and the true AP score is 0.85. Notice that the noisy examples of the query word are successfully retrieved by the proposed approach. It should be noted that OCR typically fails to recognize character glyphs which are underlined or connected to other characters in the document image. The false positive examples (word images with rank 75 and 88) have a number of common characters with the original query word. These words obtained a high rank since they are visually quite similar to the query word.

Figure 7.14c) shows the third query example, “niiku”, which is considered to be a short query. According to the ground truth, there are 11 positives examples and the AP score is 0.99. After manual investigation, it is seen that there are actually 12 positives examples which makes the actual AP score 0.98. The word images ranked 11th and 13th are the two false positives which include visually similar characters to the ones queried.

Figure 7.15 demonstrates an unsuccessful short query example “tama” with an AP of 0.0018. There is only one relevant word image to the query in the entire book and it is ranked 565. Notice that the first six false matches have the same or visually similar character glyphs appearing in the same order as the query word. These are actually subset matches which are desired to be ranked after the true positive examples. The failure analysis indicates that there might be potential improvements to the proposed approach especially for resolving short queries.

Overall, the experiments demonstrate the effectiveness of the proposed approach for searching noisy Telugu documents for which OCR and word spotting techniques








Query = "tama"	
Rank	Retrieved Image
1	 తలపూవుగా 'talapuuvugaa'
2	 మిత్రరూపులు 'mitrarupulu'
3	 తనువులు 'tanuvulu'
4	 తరువాత 'taruvaata'
5	 తలఁపఁబడియె 'talapabadiye'
6	 తగున 'taguna'
565	 తమ 'tama'

Figure 7.15. An unsuccessful short query example.

are not applicable. The training set contains a small amount of labeled word images and this was sufficient to search text in another book. The query response time is 4 milliseconds per query.

7.3.4.5 Ottoman script experiments

The document images used for Telugu and Latin experiments are printed entirely in the same font type and size. In these cases, fixing size and the orientation of the image patch across the document images is sufficient to find visual correspondences between the word images. However, this is not the case for the Ottoman dataset which includes several articles scanned from several sources and the font vary across document images. The visual features extracted from the same word images printed

in different font type and size do not match especially if the patch size is fixed. In this section three different approaches are investigated to address this problem. The first approach is to use the scale-invariant SIFT keypoint detector which automatically determines the coordinate, scale and orientation of each keypoint. The scale-invariant nature of SIFT help match visual features across different scales. However, SIFT features are known to be sensitive to certain types of document noise such as text underlining and ink bleeding [121]. The second approach is to use the fast-corner-detector to find the location of the visual terms. Each word image is assigned a uniform image patch scale relative to the height of the line it belongs to. In other words, the patch scale of a word image is its line height multiplied with a constant called “patch scale factor”. The third approach is similar to the previous one except that the height of the word bounding box is used for determining the patch scale. Unlike the SIFT approach, the latter approaches assume that the page skew is corrected and the image patch orientation is set to zero.

The effectiveness of line and box height based patch scale estimation approaches are tested on the OTTO-1 training set by varying the value of the patch scale factor. The model parameters λ_S and λ_M are trained using the visual features extracted by the scale-invariant SIFT keypoint detector. The same model parameters are used for the other settings as well. The experiments are repeated both Union and Intersection learning models and the results are given in Figure 7.16. On the training set, patch size estimation using the box height gives the best results for both learning models. The Union model provides slightly higher MAP score on the training set with a patch scale factor 0.75.

The best patch scale factor is determined for each configuration on the training set using the MAP scores plotted in Figure 7.16. Estimated patch scale factors are then applied to the test set for the corresponding configuration and the obtained MAP scores are given in Table 7.8. Notice that the MRF model parameters are estimated

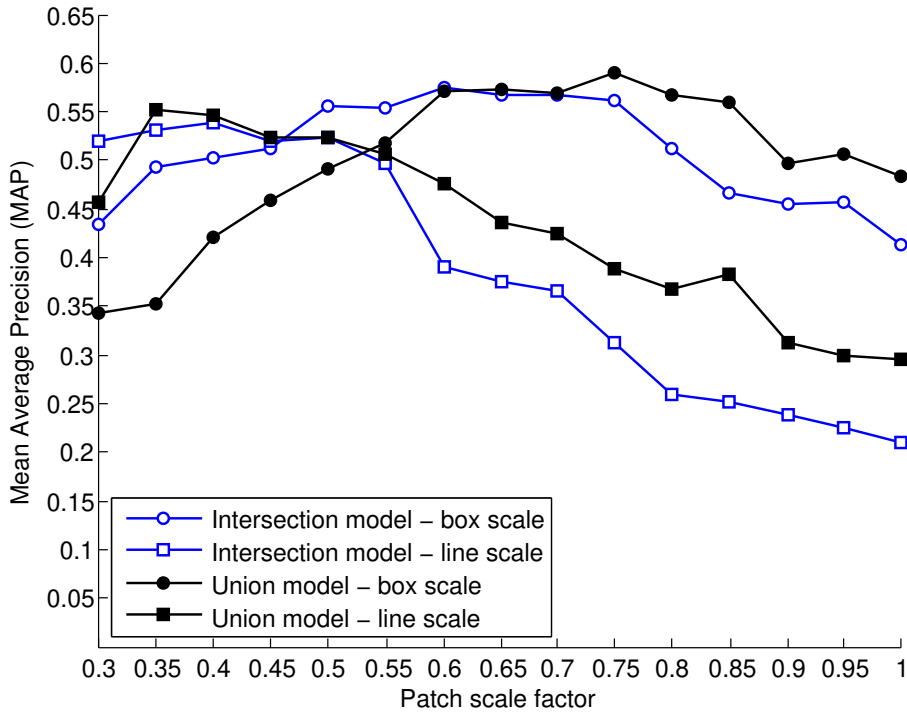


Figure 7.16. MAP scores as a function of the patch scale factor for different configurations of the proposed model computed for the Ottoman training set.

using the scale-invariant SIFT features and the same parameters are used for testing the configurations as well. On the training set, the best model uses the SIFT keypoints coupled with the Union model. On the other hand, the same configuration provides poorer MAP scores on the test set compared to the line and box height based patch scale estimation approaches. Although the Intersection model has a lower MAP value on the training set consistently, it provides higher MAP scores on the test set at all cases. This conforms with the findings of the Telugu experiment and makes the Intersection model preferable over the Union model. The highest MAP scores on the test set are obtained for the patch scale estimation using the box height. The best test configuration uses the box scale approach coupled with the Intersection model and provides a MAP score of 0.473. The second best MAP score 0.392 is obtained using Intersection model coupled with the line scale approach.

Table 7.8. Experimental results for the Ottoman dataset for three different patch size selection approaches.

Patch Scale	Training Model	Dataset	Scale factor	λ_S	λ_M	MAP
Box	Intersection	OTTO-1	0.60	0.002	0.91	0.574
		OTTO-2	0.60	0.002	0.91	0.473
	Union	OTTO-1	0.75	0.005	0.99	0.590
		OTTO-2	0.75	0.005	0.99	0.365
Line	Intersection	OTTO-1	0.40	0.002	0.91	0.539
		OTTO-2	0.40	0.002	0.91	0.392
	Union	OTTO-1	0.35	0.005	0.99	0.552
		OTTO-2	0.35	0.005	0.99	0.275
SIFT	Intersection	OTTO-1	-	0.002	0.91	0.579
		OTTO-2	-	0.002	0.91	0.385
	Union	OTTO-1	-	0.005	0.99	0.604
		OTTO-2	-	0.005	0.99	0.363

The results shown in Table 7.8 indicate that the best values for the parameter λ_S is a small number as in the case of Telugu experiments. This implies that the training instances provide strong associations between the letter bigrams and their visual terms and smoothing is not of help to improve the accuracy. This is true even though there are not a large number of training instances (<10K labeled word images). The estimated values for the λ_M are quite large - 0.99 and 0.91 for the Union and Intersection models respectively. This indicates that the dependence model gives a higher weight to the existence of letter bigrams compared to their relative positions on the horizontal axis. Further analysis indicate that some of the basic shapes used for annotating the script are quite small. Some of them simply correspond to simple loops and pieces of ink. They might be a part of many different characters in the alphabet. The width of a shape might be as small as 5-10% of the line height. It should be noted the sliding interval for the Gaussian window is set to 0.5 times the bounding box height in all experiments for simplification purposes. Estimation of the exact position of shape code pairs is therefore not quite reliable compared to Telugu and Latin scripts. Smaller values for the sliding window interval is expected to give better localization performance with additional computational cost.

The experiments show that the line height information is less reliable compared to the box height. Further analysis indicate that the line height estimations might differ significantly (5-15%) even for the documents written in the same font size. This is true not only across document images, but also the height of the lines inside the same document. There are several other contributing factors, such as page skew, page deformations, ink bleeding and text underlining. All these factors make the line height estimations more error prone. The line height estimations are not precise especially for text lines with variable lengths. Even a slight page skew has a negative impact in the line height estimations. For example, consider a document which is composed of a number of text lines written in the same font with different length. In such cases the projection based line height estimation methods produce variable line height values for each line. The situation becomes more severe across documents written in different fonts. The result is a reduced number of matching visual terms between the corresponding word image and this is not desirable for matching and retrieval purposes.

The patch scale estimation using the box height approach has also additional complexities. For example, the words “the” and “they” both contain the letter bigram “th” but the height of their bounding boxes are quite different because of the descending letter “y”. Although the text is written in the same font and size, the visual terms extracted from “the” and “they” might not match because of different size image patches used. In those cases, there might be quite a few corresponding visual terms across a given pair of word images containing the same letter bigram. Those matching visual terms might not be sufficient for pair-wise word image comparison purposes. On the other hand, the proposed dependence model learns the distribution of visual terms across several training instances which contain the associated letter bigrams at different contexts. For example, for the letter “th”, the training instances contain the word images labeled as “the”, “they”, “throne” etc. Visual terms appear-

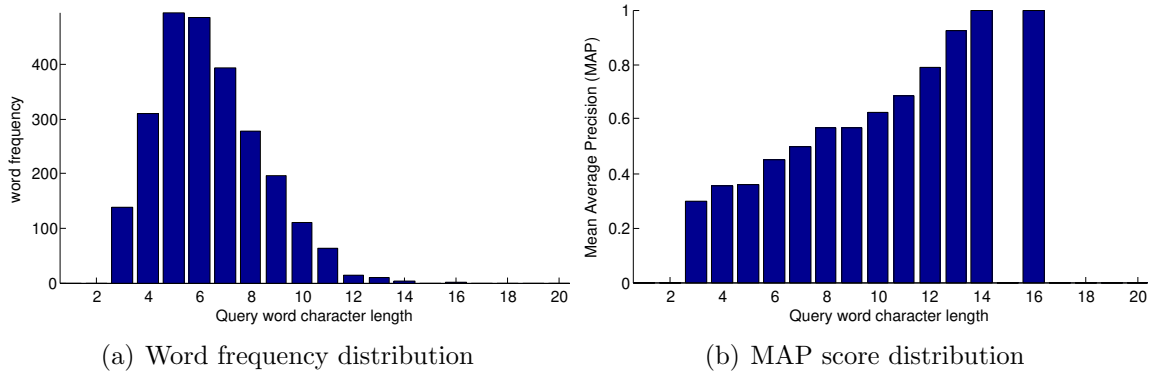


Figure 7.17. MAP score distribution of the best configuration on the OTTO-2 test set as a function of the query word length.

ing in different training word images contribute to the visual term distribution for the letter bigram class “th”. The proposed dependence model directly uses vistern distributions for each letter bigram class. This helps retrieve relevant word images even though the word images in the rank list have quite distinct set of visual terms each.

The Figure 7.17 shows the distribution of MAP scores as a function of word length. In the case of Ottoman dataset, each word image is encoded by a sequence of shape codes and each code corresponds to a single shape in the corresponding image. Therefore average word length in the Ottoman dataset is not as high as in the Telugu dataset. It is clear that the MAP scores increase as the total number of characters increase in the query word. As discussed in the case of Telugu script, it is desirable to have high accuracy for more complex queries which are typically longer words.

As in the case of Telugu experiments, most words in the vocabulary of the OTTO-2 test set (82.6%) appear only once in the entire context. 97.2% of the words in the vocabulary appear no more than three times in the entire test set. Clearly word spotting approaches are not applicable for the Ottoman dataset as well. A MAP score of 0.473 indicates that the relevant word images are typically retrieved at the

Query	“uJAJFLsLaxU”	“axvAcg”
Rank	Retrieved Images	Retrieved Images
1	● قائممقامی 'uJAJFLsLaxU'	● مجبور 'axvAcg'
2	● قائممقامی 'uJAJFLsLaxU'	● مجبور 'axvAcg'
3	● قائممقامی 'sLuJAJFLxU'	● بیوریلان 'uQduAVg'
4	● قالقشمش 'qFKAJKNsL'	● بیوریلن 'uBNuAVg'
5	● تاریخی 'uLuHUg'	● مجبور 'axvAcg'
6	● کتبخانه لرده کی 'tAAHLtUqVuOfh'	● آرزو سیله 'wAABObcgg'
7	● کتبخانه سی 'tAAHLuOwU'	● محرر 'axvVg'

Figure 7.18. Two example queries on the OTTO-2 test set.

top a few positions of the rank list. Specifically, the expected rank of a relevant word image is approximately two for a query word with only one match.

Figure 7.18 shows two example queries on the Ottoman test set. The example on the left represents a long query word and there are only three matching words in the ground truth. The proposed approach retrieved all the relevant word images at the top of the ranked list, therefore it has an AP score of 1.0. As in the case of Telugu experiments, the proposed approach was able to find annotation errors in the dataset. Notice that the word image ranked 3rd is retrieved correctly by the proposed approach but it has an incorrect label. The 4th retrieved image is not a match but

it is visually quite similar to the true positive examples. The example on the right represents the short query example with an AP score of 0.867. There are only three relevant matches as well. The first two relevant word images are top ranked, however, the last one ranked 5th after two false positives. Notice the word images 3rd and 4th have partial visual similarity with the query word. The first four letters from the right partially match the shapes and characters of the query word. Partial matches might be useful for users especially if the search term does not appear in the test set. Partially matching words are typically inflections or morphological variations of the query word and they might also be considered to be relevant depending on the search task.

In this section a dependence model is introduced for resolving arbitrary text queries in document images with a real time performance. The proposed training models can effectively learn the visual term distributions from noisy training data. The Latin experiments have shown that image features can be used to improve the OCR text search using the global font feature. The effectiveness of the proposed approach is also demonstrated for different books and scripts for which there is no OCR engine available.

CHAPTER 8

CONCLUSIONS AND FUTURE WORK

Millions of books have been digitized so far around the globe for preservation purposes. These books contain the written heritage of human civilization. The information buried in these collections is therefore very important. Several abstraction levels have been discussed for large scanned book collections. One can view the entire collection as a whole and discover linkages between the books. Another approach is to perform information search and mining at the individual book level. In this dissertation, we also demonstrated that one can also view each book as composed of chapters, sections, paragraphs, sentences, words or even characters positioned in a particular sequential order sharing the same global context. The information inherent in the entire context of the book is referred to as global information and its effective use is demonstrated by addressing a number of research questions defined for scanned book collections.

The global sequence information is essential for discovering content overlap and similarity across books. A global text alignment approach using the OCR output is therefore adopted for this purpose. The problem is, conventional global sequence alignment algorithms do not scale for book length documents and they are not robust for aligning noisy texts with large amounts of additional or missing content. As a solution, the sequence of unique words text representation scheme is proposed. It is demonstrated that this representation scheme efficiently aligns and compares long noisy texts such as the OCR output of scanned books. This approach has also been extended for aligning text across languages. This is achieved simply by transforming

the sequence of words in the source book to the language of the target book using a dictionary based approach. Once the two word sequences are in the same language, the two texts are compared or aligned as if they are written in the same language. Given a translation pair, this approach has been shown to map duplicated content in the form of translation despite the fact that the local word order might not be preserved across translations. The sequence of unique words representation scheme is shown to be quite efficient and effective and therefore practical for large scale text alignment and comparison tasks defined for scanned book collections.

The limitation of the sequence of unique words text representation scheme is that it relies on a single occurrence of words in the entire content. If some portion of the text is repeated inside the main body, then the alignment is expected to fail for the repeated parts of the text. The proposed approaches also fail for input texts containing long lists of names and/or items sorted in a particular order. For example, dictionary entries are lexicographically sorted in all dictionaries. Therefore the proposed approach can not be used for finding duplicates of dictionaries, although it can potentially be used to detect dictionaries in scanned book collections.

The proposed partial duplicate detection framework (DUPNIQ) using the sequence of unique words scheme is primarily designed for efficiently aligning or comparing long noisy texts written in some natural language such as books. It is not designed for detecting text reuse or quotations. Synthetic experiments demonstrates that about 15% content overlap can be detected using DUPNIQ. The duplication at the level of a few pages might therefore not be detected. In the case of scanned books, the amount of content overlap is typically between 10 to 80%. It is demonstrated that DUPNIQ effectively finds partial duplicates in scanned book collections at scale without any need for aligning entire texts. The proposed Recursive Text Alignment Scheme serves as an alternative to DUPNIQ for detecting/mapping page or paragraph level content overlap.

The OCR text based approaches, including RETAS, DUPNIQ, TRANSNIQ and RTA, rely on the accuracy of the character recognition. In most cases, the OCR output is sufficiently accurate to address certain research questions such as duplicate and translation detection/mapping. The experiments demonstrate that the proposed approaches are highly robust compared to their alternatives in the literature. On the other hand, there are a large number of books for which there is no OCR output or the recognition accuracy is quite low. For example, the OCR output of German books printed in Fraktur is typically garbled in the IA collection. The reason is that OCR engine used does not recognize Fraktur. OCR text based solutions are therefore not applicable in such cases. Another example is that there are no commercial engines to recognize certain scripts such as Telugu and Ottoman. Some of those scripts are still being used by millions of people. For example, Telugu language has over 75 million speakers today. It is desirable to have automatic access to the textual content of documents printed in those scripts for which there is no OCR engine available. In this respect, two problem domains might be defined in the context of scanned books. First, what can we do with the long noisy OCR text outputs? Second, what can we do to facilitate text search and mining in the scanned pages of books when OCR fails?

In the second part of the dissertation, image search based solutions are investigated to facilitate text search in noisy document images. The global font feature along with the letter sequence information is demonstrated to be useful for facilitating and/or improving text search in noisy page images. Each book is regarded as a collection of word images printed in the same font type. The task is to retrieve all the instances of a given query word in the entire context using visual features. First, an efficient word spotting framework is proposed where a word image instance of the query word is given as a query. The similarity measure between the word images are computed by aligning the sequence of visual terms. Along with the global font feature and

the visual term sequence information, the word images are efficiently retrieved with high accuracy. The general problem with the word spotting technique is that the user has to provide an example image of the query word. Word spotting approaches are therefore not feasible for searching words which appear rarely because finding an example word image in the book is a tedious task. As a remedy to this problem, a dependence model is introduced and it enables searching arbitrary text queries in document images. The dependencies between the letter bigrams in the query text are trained automatically and used for locating letter bigrams in the word images. Letter bigram sequence information is also incorporated into the final matching score to resolve arbitrary text queries. It has been shown that combining the OCR text search techniques with the proposed dependence approach significantly improves text search accuracy for documents printed in Latin script. The effectiveness of the proposed approaches is demonstrated for searching text in noisy document images written in different languages and scripts.

The primary limitation of image search mechanisms is that operations over high dimensional feature vectors are computationally expensive. In this context, the speed limitation is avoided/minimized by quantizing feature vectors into discrete values using efficient clustering techniques. Along with efficient indexing of visual features, the proposed approaches provide real time search performance with high retrieval accuracy. Another limitation of the proposed approaches is that they are not effective to searching document images written in multiple fonts. Word images printed in different fonts might look visually different and this has a negative impact on the retrieval accuracy. Telugu and Ottoman experiments further demonstrate that the proposed approaches perform reasonably well across books printed in different fonts. One possible extension might be to train a number of visual models for different font types to alleviate the font sensitivity problems. It should be noted that the majority of the scanned books are mostly printed in a single global font type.

Future work includes investigation of the concepts and tools developed here for different problem domains. The proposed text alignment techniques provide an efficient and effective way to analyze and compare long texts at scale. Global text alignment approaches are widely used especially for plagiarism and copy detection in digital libraries. Another research direction is to investigate the effectiveness of the proposed approaches on other types of datasets such as web collections. There might also be several other applications in the computer vision domain. Given a sequence representation of images and videos, one can efficiently align them to find duplicated content. Preliminary experiments suggest that the duplicate detection approach presented for texts is applicable for finding duplicates of videos if there is significant content overlap. Searching text in noisy document images has also several potential applications. The frameworks presented here can be easily configured for searching or recognizing text printed in scripts for which there is no OCR engine available. The proposed approaches can also be adapted for addressing handwriting analysis and retrieval problems.

BIBLIOGRAPHY

- [1] The Internet Archive: digital library. <http://www.archive.org>, 2012.
- [2] Project Gutenberg: free ebooks. <http://www.gutenberg.org>, 2012.
- [3] Wordgumbo: free dictionaries. <http://www.wordgumbo.com>, 2012.
- [4] Almazan, Jon, Gordo, Albert, Fornes, Alicia, and Valveny, Ernest. Handwritten word spotting with corrected attributes. In *15th IEEE International Conference on Computer Vision* (2013).
- [5] Altschul, Stephen F., Gish, Warren, Miller, Webb, Myers, Eugene W., and Lipman, David J. Basic local alignment search tool. *Journal of Molecular Biology* 215, 3 (Oct 1990), 403–410.
- [6] Ataer, Esra, and Duygulu, Pinar. Matching ottoman words: an image retrieval approach to historical document indexing. In *CIVR* (2007), pp. 341–347.
- [7] Bai, Shuyong, Li, Linlin, and Tan, Chew Lim. Keyword spotting in document images through word shape coding. In *ICDAR* (2009), pp. 331–335.
- [8] Ball, Gregory R., Srihari, Sargur N., and Srinivasan, Harish. Segmentation-based and segmentation-free methods for spotting handwritten arabic words. In *Tenth International Workshop on Frontiers in Handwriting Recognition* (Oct. 2006).
- [9] Ballesteros, L., and Croft, W. B. Dictionary methods for cross-lingual information retrieval. In *DEXA* (1996), pp. 791–801.
- [10] Ballesteros, L., and Croft, W. B. Resolving ambiguity for cross-language retrieval. In *SIGIR* (1998), pp. 64–71.
- [11] Barrón-Cedeño, Alberto, Rosso, Paolo, Pinto, David, and Juan, Alfons. On cross-lingual plagiarism analysis using a statistical model. In *PAN* (2008).
- [12] Beitzel, Steven M., Jensen, Eric C., and Grossman, David A. Retrieving OCR text: A survey of current approaches. In *Symposium on Document Image Understanding Technologies (SDUIT)* (2003).
- [13] Berger, Adam L., and Lafferty, John D. Information retrieval as statistical translation. In *SIGIR* (1999), pp. 222–229.

- [14] Bernstein, Y., and Zobel, J. A scalable system for identifying co-derivative documents. In *SPIRE* (2004), pp. 55–67.
- [15] Boschetti, Federico, Romanello, Matteo, Babeu, Alison, Bamman, David, and Crane, Gregory. Improving OCR accuracy for classical critical editions. In *ECDL'09* (2009), pp. 156–167.
- [16] Braune, Fabienne, and Fraser, Alexander. Improved unsupervised sentence alignment for symmetrical and asymmetrical parallel corpora. In *COLING (Posters)* (2010), pp. 81–89.
- [17] Brin, S., Davis, J., and Garcia-Molina, H. Copy detection mechanisms for digital documents. In *ACM SIGMOD* (1995), pp. 398–409.
- [18] Broder, A. Z., Glassman, S. C., Manasse, M. S., and Zweig, G. Syntactic clustering of the web. *Computer Networks* 29, 8-13 (1997), 1157–1166.
- [19] Brown, Peter F., Pietra, Vincent J. Della, Pietra, Stephen A. Della, and Mercer, Robert L. The mathematics of statistical machine translation: parameter estimation. *Comp. Ling.* 19 (June 1993), 263–311.
- [20] Charikar, M. S. Similarity estimation techniques from rounding algorithms. In *34th Ann. ACM Symp. on Theory of computing* (2002), pp. 380–388.
- [21] Chen, Stanley F. Aligning sentences in bilingual corpora using lexical information. In *ACL* (1993), pp. 9–16.
- [22] Chowdhury, A., Frieder, O., Grossman, D. A., and McCabe, M. C. Collection statistics for fast duplicate document detection. *ACM Trans. Inf. Syst.* 20, 2 (2002), 171–191.
- [23] Civera, J., and Juan, A. Unigram-IBM Model 1 Mixtures for Bilingual Text Classification. In *Proc. of LREC'08* (May 2008).
- [24] Clough, P. Old and new challenges in automatic plagiarism detection. National UK Plagiarism Advisory Service, http://www.ir.shef.ac.uk/cloughie/papers/pas_plagiarism.pdf, 2003.
- [25] Cooper, J.W., Coden, A.R., and Brown, E.W. Detecting similar documents using salient terms. In *CIKM* (2002), pp. 245–251.
- [26] Crochemore, M., and Porat, E. Computing a longest increasing subsequence of length k in time $o(n \log \log k)$. In *Proceedings of the 2008 International Conference on Visions of Computer Science: BCS International Academic Conference* (2008), VoCS'08, pp. 69–74.
- [27] Croft, W. Bruce, Metzler, Donald, and Strohman, Trevor. *Search Engines - Information Retrieval in Practice*. Pearson Education, 2009.

- [28] Danielsson, Per-Erik. Euclidean Distance Mapping. *Computer Graphics And image Processing 14* (1980), 227–248.
- [29] Delcher, A. L. Kasif, S., Fleischmann, R. D., Peterson, J., White, O., and Salzberg, S. L. Alignment of whole genomes. *Nucleic Acids Research 27*, 11 (1999), 2369–2376.
- [30] Deng, Yonggang, Kumar, Shankar, and Byrne, William. Segmentation and alignment of parallel text for statistical machine translation. *Natural Language Engineering 12*, 4 (2006), 1–26.
- [31] Deorowicz, Sebastian. Solving longest common subsequence and related problems on graphical processing units. *Softw. Pract. Exper. 40* (July 2010), 673–700.
- [32] Durbin, Richard, Eddy, Sean R., Krogh, Anders, and Mitchison, Graeme. *Biological sequence analysis: probabilistic models of proteins and Nucleic acids*. Cambridge University Press, 1999.
- [33] Errami, M., Sun, Z., George, A. C., Long, T. C., Skinner, M. A., Wren, J. D., and Garner, H. R. Identifying duplicate content using statistically improbable phrases. *Bioinformatics 26*, 11 (2010), 1453–1457.
- [34] Errami, Mounir, Wren, Jonathan D., Hicks, Justin M., and Garner, Harold R. etblast: a web server to identify expert reviewers, appropriate journals and similar publications. *Nucleic Acids Research 35*, Web-Server-Issue (2007), 12–15.
- [35] Feng, S., and Manmatha, R. A hierarchical, HMM-based automatic evaluation of OCR accuracy for a digital library of books. In *JCDL* (2006), pp. 109–118.
- [36] Feng, Shaolei, and Manmatha, Raghavan. A discrete direct retrieval model for image and video retrieval. In *CIVR* (2008), pp. 427–436.
- [37] Fischler, Martin A., and Bolles, Robert C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM 24* (June 1981), 381–395.
- [38] Fung, P. Compiling bilingual lexicon entries from a non-parallel english-chinese corpus. In *Annual Meeting of Very Large Corpora* (1995).
- [39] Govindaraju, Venu, and Setlur, Srirangaraj. *Guide to OCR for Indic Scripts: Document Recognition and Retrieval*, 1st ed. Springer Publishing Company, Incorporated, 2009.
- [40] Hajishirzi, H., tau Yih, Wen, and Kolcz, A. Adaptive near-duplicate detection via similarity learning. In *SIGIR’10* (2010), pp. 419–426.

- [41] Harding, S.M., Croft, W. B., and Weir, C. Probabilistic retrieval of ocr degraded text using n-grams. In *European Conference on Digital Libraries (1997)*, pp. 345–359.
- [42] Harding, S.M., Croft, W. B., and Weir, C. Probabilistic retrieval of OCR degraded text using n-grams. In *In Proc. of the 1st European Conference on Research and Advanced Technology for Digital Libraries. pp. (1997)*, pp. 345–359.
- [43] Heintze, N. Scalable document fingerprinting. In *USENIX Workshop on Electronic Commerce (1996)*.
- [44] Henzinger, M. Finding near-duplicate web pages: a large-scale evaluation of algorithms. In *ACM SIGIR (2006)*, pp. 284–291.
- [45] Hirschberg, D. S. A linear space algorithm for computing maximal common subsequences. *Commun. ACM* 18 (1975).
- [46] Hoad, T. C., and Zobel, J. Methods for identifying versioned and plagiarized documents. *JASIST* 54, 3 (2003), 203–215.
- [47] Huang, Weichun, Umbach, David M., and Li, Leping. Accurate anchoring alignment of divergent sequences. *Bioinformatics* 22, 1 (2006), 29–34.
- [48] Hunt, J. W., and McIlroy, M. D. An algorithm for differential file comparison. In *Computing Science Technical Report, Bell Laboratories (June 1976)*, vol. 41.
- [49] Hunt, James W., and Szymanski, Thomas G. A fast algorithm for computing longest common subsequences. *Commun. ACM* 20 (May 1977), 350–353.
- [50] Koehn, Philipp. Europarl: A Parallel Corpus for Statistical Machine Translation. In *The tenth Machine Translation Summit (Phuket, Thailand, 2005)*, AAMT, pp. 79–86.
- [51] Konidaris, T., Gatos, B., Ntzios, K., Pratikakis, I., Theodoridis, S., and Perantonis, S.J. Keyword-guided word spotting in historical printed documents using synthetic data and user feedback. *International Journal of Document Analysis and Recognition (IJ DAR)* 9, 2-4 (2007), 167–177.
- [52] Koroutchev, K., and Cebri, M. Detecting the same text in different languages. In *IEEE Information Theory Workshop ITW (2007)*, pp. 337–341.
- [53] Krstovski, Kriste, and Smith, David A. A minimally supervised approach for detecting and ranking document translation pairs. In *6th Workshop on SMT (2011)*, pp. 207–216.
- [54] Kukich, K. Technique for automatically correcting words in text. *ACM Computing Surveys* 24, 4 (Dec. 1992), 377–439.

- [55] Lavrenko, V., Choquette, M., and Croft, W. B. Cross-lingual relevance models. In *SIGIR* (2002), pp. 175–182.
- [56] Lee, Dar-Shyang, and Smith, Ray. Improving book OCR by adaptive language and image models. In *Document Analysis Systems* (2012), pp. 115–119.
- [57] Levow, G., Oard, D. W., and Resnik, P. Dictionary-based techniques for cross-language information retrieval. *Information Processing. Management.* 41, 3 (2005), 523–547.
- [58] Lin, Dekang. An information-theoretic definition of similarity. In *ICML '98* (1998), pp. 296–304.
- [59] Louloudis, G., Gatos, B., and Halatsis, C. Text line detection in unconstrained handwritten documents using a block-based hough transform approach. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition - Volume 02* (2007), ICDAR '07, pp. 599–603.
- [60] Lowe, David G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision* 60 (November 2004), 91–110.
- [61] Lu, Shijian, Li, Linlin, and Tan, C.L. Document image retrieval through word shape coding. *Pattern Analysis and Machine Intelligence (PAMI), IEEE Transactions on* 30, 11 (2008), 1913–1918.
- [62] Ma, Bin, Tromp, John, and Li, Ming. PatternHunter: faster and more sensitive homology search. *Bioinformatics* 18, 3 (Mar. 2002).
- [63] Ma, X., and Liberman, M. Bits: A method for bilingual text search over the web. In *Machine Trans. Summit VII* (1999).
- [64] Manber, U. Finding similar files in a large file system. In *USENIX Winter 1994 Tech. Conf* (1994), pp. 1–10.
- [65] Manmatha, R., Han, Chengfeng, and Riseman, E. M. Word spotting: a new approach to indexing handwriting. In *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96, 1996 IEEE Computer Society Conference on* (1996), pp. 631–637.
- [66] Manmatha, R., and Rothfeder, Jamie L. A scale space approach for automatically segmenting words from historical handwritten documents. *IEEE Trans. Pattern Anal. Mach. Intell.* 27, 8 (2005), 1212–1225.
- [67] Marinai, Simone, Marino, Emanuele, and Soda, Giovanni. A comparison of clustering methods for word image indexing. In *Document Analysis Systems* (2008), pp. 671–676.
- [68] Marti, Urs-Viktor, and Bunke, Horst. Text line segmentation and word recognition in a system for general writer independent handwriting recognition. In *ICDAR* (2001), pp. 159–163.

- [69] Melamed, I. Dan. A portable algorithm for mapping bitext correspondence. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics* (1997), ACL '98, pp. 305–312.
- [70] Metzler, D., and Croft, W. B. A markov random field model for term dependencies. In *Proc. of the 28th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '05)*. (2005), pp. 472–479.
- [71] Mimno, D., Crane, G., and Jones, A. Hierarchical catalog records: Implementing a FRBR catalog. *D-Lib Magazine* 11, 10 (Oct 2005).
- [72] Moore, Robert C. Fast and accurate sentence alignment of bilingual corpora. In *AMTA* (2002), pp. 135–144.
- [73] Murdock, Vanessa, and Croft, W. Bruce. A translation model for sentence retrieval. In *in Proceedings of the Conference on Human Language Technologies and Empirical Methods in Natural Language Processing (HLT/EMNLP)* (2005), pp. 684–691.
- [74] Nie, J. Y., Simard, M., Isabelle, P., and Durand, R. Cross-language information retrieval based on parallel texts and automatic mining of parallel texts from the web. In *ACM SIGIR* (1999), pp. 74–81.
- [75] Nister, David, and Stewenius, Henrik. Scalable recognition with a vocabulary tree. In *Proceedings of CVPR* (2006), pp. 2161–2168.
- [76] Noé, L., and Kucherov, G. YASS: enhancing the sensitivity of DNA similarity search. *Nucleic Acids Research* 33 (July 2005), W540–543.
- [77] Oard, D. W. A comparative study of query and document translation for cross-language information retrieval. In *AMTA* (1998), pp. 472–483.
- [78] Pal, U., and Chaudhuri, B.B. Indian script character recognition: a survey. *Pattern Recognition* 37, 9 (2004), 1887 – 1899.
- [79] Parapar, Javier, Freire, Ana, and Barreiro, Álvaro. Revisiting n-gram based models for retrieval in degraded large collections. In *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval* (Berlin, Heidelberg, 2009), ECIR '09, Springer-Verlag, pp. 680–684.
- [80] Pearson, W. R., and Lipman, D. J. Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences of the United States of America* 85, 8 (Apr. 1988), 2444–2448.
- [81] Pinto, David, Juan, Alfons, and Rosso, Paolo. Using query-relevant documents pairs for cross-lingual information retrieval. In *TSD* (2007), pp. 630–637.

- [82] Potthast, Martin, Barrón-Cedeño, Alberto, Stein, Benno, and Rosso, Paolo. Cross-language plagiarism detection. *Language Resources and Evaluation* 45, 1 (2011), 45–62.
- [83] Rath, T., and Manmatha, R. Word image matching using dynamic time warping. In *Proceedings of CVPR'03* (2003), vol. 2, pp. 521–527.
- [84] Rath, T. M., and Manmatha, R. Word spotting for historical documents. *IJDAR* 9, (2-4) (2007), 139–152.
- [85] Rath, Toni M., and Manmatha, R. Features for word spotting in historical manuscripts. In *ICDAR* (2003), pp. 218–222.
- [86] Rath, Toni M., and Manmatha, R. Features for word spotting in historical manuscripts. In *ICDAR* (2003), pp. 218–222.
- [87] Resnick, P., and Smith, N. The web as a parallel corpus. *Computational Linguistics* 29, 3 (2003).
- [88] Resnik, P. Mining the web for bilingual text. In *ACL* (1999), pp. 527–534.
- [89] Rice, Stephen V. Measuring the accuracy of page-reading systems. In *PH.D. DISSERTATION, UNLV, LAS VEGAS* (1996).
- [90] Rodriguez, J. A., and Perronnin, F. Local gradient histogram features for word spotting in unconstrained handwritten documents. In *Proc. ICFHR* (2008), pp. 7–12.
- [91] Rosten, Edward, and Drummond, Tom. Machine learning for high-speed corner detection. In *European Conference on Computer Vision* (May 2006), vol. 1, pp. 430–443.
- [92] Sankar, K. Pramod, and Jawahar, C. V. Probabilistic reverse annotation for large scale image retrieval. In *CVPR* (2007).
- [93] Sankar K., Pramod, Jawahar, C. V., and Manmatha, R. Nearest neighbor based collection OCR. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems* (2010), DAS '10, pp. 207–214.
- [94] Schleimer, S., Wilkerson, D.S., and Aiken, A. Winnowing: local algorithms for document fingerprinting. In *ACM SIGMOD conference* (2003), pp. 76–85.
- [95] Schwartz, Scott, Kent, W. James, Smit, Arian, Zhang, Zheng, Baertsch, Robert, Hardison, Ross C., Haussler, David, and Miller, Webb. HumanMouse Alignments with BLASTZ. *Genome Research* 13, 1 (Jan. 2003), 103–107.
- [96] Scott, G., and Higgins, Longuet H. An algorithm for associating the features of two patterns. In *Proc. Royal Society London* (1991), vol. B244, pp. 21–26.

- [97] Seo, J., and Croft, W. B. Local text reuse detection. In *ACM SIGIR* (2008), pp. 571–578.
- [98] Shivakumar, N., and Garcia-Molina, H. Scam: A copy detection mechanism for digital documents. In *Ann. Conf. on the Theory and Practice of Digital Libraries* (1995).
- [99] Shivakumar, N., and Garcia-Molina, H. Finding near-replicas of documents on the web. In *Intl. Workshop on the World Wide Web and Databases* (1999).
- [100] Simard, Michel, Foster, George F., and Isabelle, Pierre. Using cognates to align sentences in bilingual corpora. In *Proceedings of the 1993 conference of the Centre for Advanced Studies on Collaborative research: distributed computing - Volume 2* (1993), CASCON '93, pp. 1071–1082.
- [101] Singh, Anil Kumar, and Husain, Samar. Comparison, selection and use of sentence alignment algorithms for new language pairs. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts* (2005), ParaText'05, pp. 99–106.
- [102] Smith, N. From words to corpora: Recognizing translation. In *EMNLP* (2002), pp. 95–102.
- [103] Theobald, M., Siddharth, J., and Paepcke, A. Spotsigs: robust and efficient near duplicate detection in large web collections. In *SIGIR 08* (2008), pp. 563–570.
- [104] Tong, X., and Evans, D. A. A statistical approach to automatic OCR error correction in context. In *Proceedings of the Fourth Workshop on Very Large Corpora (WVLC-96)* (1996), pp. 88–100.
- [105] T.Rath, Manmatha, R., and Lavrenko., V. A search engine for historical manuscript images. In *in the Proceedings of SIGIR'04*, pp. 297 -304 (2004).
- [106] Ukkonen, Esko. Approximate string-matching with q-grams and maximal matches. *Theor. Comput. Sci.* 92, 1 (Jan. 1992), 191–211.
- [107] Ukkonen, Esko. On-line construction of suffix trees. *Algorithmica* 14, 3 (1995), 249–260.
- [108] Uszkoreit, Jakob, Ponte, Jay, Popat, Ashok C., and Dubiner, Moshe. Large scale parallel document mining for machine translation. In *COLING* (2010), pp. 1101–1109.
- [109] van Emde Boas, Peter. Preserving order in a forest in less than logarithmic time. In *FOCS* (1975), pp. 75–84.
- [110] Vedaldi, A., and Fulkerson, B. Vlfeat – an open and portable library of computer vision algorithms. In *Proc. of the 18th annual ACM international conference on Multimedia* (2010).

- [111] Wemhoener, David, Yalniz, Ismet Zeki, and Manmatha, R. Creating an improved version using noisy OCR from multiple editions. In *ICDAR* (2013), pp. 160–164.
- [112] Wshah, Safwan, Kumar, Gaurav, and Govindaraju, Venu. Multilingual word spotting in offline handwritten documents. In *ICPR* (2012), pp. 310–313.
- [113] Wu, Changchang. SiftGPU: A GPU implementation of scale invariant feature transform. <http://cs.unc.edu/~ccwu/siftgpu>, 2007.
- [114] Wu, Dekai. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics* 23, 3 (1997), 377–403.
- [115] Xiao, Chuan, Wang, Wei, Lin, Xuemin, Yu, Jeffrey Xu, and Wang, Guoren. Efficient similarity joins for near-duplicate detection. *ACM Trans. Database Syst.* 36, 3 (2011), 15.
- [116] Xu, J., Weischedel, R. M., and Nguyen, C. Evaluating a probabilistic model for cross-lingual information retrieval. In *SIGIR* (2001), pp. 105–110.
- [117] Yalniz, Ismet Zeki, Altingovde, Ismail Sengor, Gudukbay, Ugur, and Ulusoy, Ozgur. Integrated segmentation and recognition of connected ottoman script. *Optical Engineering* 48, 11 (2009), 117205–117205–12.
- [118] Yalniz, Ismet Zeki, Altingövde, Ismail Sengör, Güdükbay, Ugur, and Ulusoy, Özgür. Ottoman archives explorer: A retrieval system for digital ottoman archives. *JOCCH* 2, 3 (2009), 8.
- [119] Yalniz, Ismet Zeki, Can, Ethem F., and Manmatha, R. Partial duplicate detection for large book collections. In *CIKM* (2011), pp. 469–474.
- [120] Yalniz, Ismet Zeki, and Manmatha, R. A fast alignment scheme for automatic OCR evaluation of books. In *ICDAR* (2011), pp. 754–758.
- [121] Yalniz, Ismet Zeki, and Manmatha, R. An efficient framework for searching text in noisy document images. In *Document Analysis Systems* (2012), pp. 48–52.
- [122] Yalniz, Ismet Zeki, and Manmatha, R. Finding translations in scanned book collections. In *SIGIR* (2012), pp. 465–474.
- [123] Yalniz, Ismet Zeki, and Manmatha, R. UMASS language identification tool for long noisy texts. <http://ciir.cs.umass.edu/downloads/language-identification/>, 2013.
- [124] Yang, C. C., and Li, K. W. Automatic construction of english/chinese parallel corpora. *JASIS* (2003), 730–742.