DEGRADED TEXT RECOGNITION USING VISUAL AND LINGUISTIC CONTEXT

by

Tao Hong

A dissertation submitted to the
Faculty of the Graduate School of the
State University of New York at Buffalo
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

September, 1995

Committee Members

Chairman:	Dr. Jonathan J. Hull Ricoh California Research Center
Members:	Dr. Sargur N. Srihari Computer Science, SUNY at Buffalo
	Dr. Deborah K. Walters Computer Science, SUNY at Buffalo
Outside Reader:	Dr. Henry S. Baird AT&T Bell Laboratories

This dissertation was defended on September 22, 1995.

To Jie Xu, my wife, and James Jia-Ning Hong, my son.
To my parents and my parents-in-law.
To my sister and my brother-in-law.

....
And I say that life is indeed darkness
save when there is urge,
And all urge is blind save when there is
knowledge,
And all knowledge is vain save when
there is work,
And all work is empty save when there
is love;
And when you work with love you bind
yourself to yourself, and to one another,
and to God.

—– Kahlil Gibran *The Prophet*

Acknowledgements

First of all, I would like to express my sincere gratitude to Dr. Jonathan J. Hull, my adviser, for supervising me in the last four years. Dr. Hull introduced me to the area of visual document recognition and motivated me to conduct the research described in this dissertation. I will be always grateful to his enthusiastic guidance and support which made this dissertation possible.

I am also indebted to Professor Sargur N. Srihari, member of dissertation committee and director of CEDAR, for his interests in the work and his encouragement. It is his support, particularly in the last year, that made this dissertation finished on time.

I would like to thank Professor Deborah K. Walters, member of dissertation committee, for serving on the committee.

I am also grateful to Dr. Henry S. Baird of AT&T Bell Laboratories, for being an external reader of this thesis. His constructive comments and suggestions made the thesis in better shape and will motivate me to do more solid work in the future time.

I would like to thank Dr. Tin Kam Ho for her help while she was in CEDAR. I would also like to thank Jason Zheng-Nan Zhu, Chi Fang, Siamake Khoubyari, Yan-Hong Li, Ginige DeSilva, Thomas E. Pilarski and Afshin Shafie for sharing ideas in our friendly and open research group led by Dr. Hull.

This research was done in CEDAR, a pleasant and friendly environment for me to learn and study. Special thanks to many friends – Jian Zhou, Liang Li, Sheila Xin Zhao, Dar-Shyang Lee, Zhou Hong, P.K. Wong, Rong-Hui Zhu, Giovanni Seni, James Lii, Kripa Sundar, Debashish Niyogi, Zhong-Fei Zhang, Zhi-Xin Shi and Jing-Zhu Han. Special thanks to peoples from the JOCR group – Geetha Srikantan, S.W. Lam, Qun-Feng Liao and Victor Zandy. Also thanks Mr. David Bortnik and Dr. Venugopal Govindaraju, for the convenience they provided when I worked for the RCP group.

Special thanks to faculty of the Department of Computer Science, for the nice graduate program I got in past five years. Also thanks to Chongguang Tao, Yiming Gong and Biao Cheng for their friendship and help.

Special thanks to the Caere Corp. for providing their OCR package for the experiments described in this thesis.

Special thanks to Joe Bates, who kindly helped proofread the manuscript.

Finally I would like to extend my gratitude to Heike Berlind, Eugenia Smith, Natalie Leaver and Rita Hamilton, secretaries of CEDAR, and Eloise Benzel, Gloria Koontz, Sally Elder, Elenora M. Heffner, Joann Glinski of the Department of Computer Science, for their kindness and help through the last five years.

Abstract

Recognition of degraded text is a challenging problem. To improve the performance of an OCR system on degraded images of text, postprocessing techniques are critical. The objective of postprocessing is to correct errors or to resolve ambiguities in OCR results by using contextual information. Depending on the extent of context used, there are different levels of postprocessing. In current commercial OCR systems, word-level postprocessing methods, such as dictionary-lookup, have been applied successfully. However, many OCR errors cannot be corrected by word-level postprocessing. To overcome this limitation, passage-level postprocessing, in which global contextual information is utilized, is necessary. In most current studies on passage-level postprocessing, linguistic context is the major resource to be exploited.

This thesis addresses problems in degraded text recognition and discusses potential solutions through passage-level postprocessing. The objective is to develop a postprocessing methodology from a broader perspective. In this work, two classes of inter-word contextual constraints, *visual constraints* and *linguistic constraints*, are exploited extensively.

Given a text page with hundreds of words, many word image instances can be found to be equivalent, either entirely or partially, by image matching. Formally, several types of visual inter-word relations are defined. Relations at the image level must be consistent with the relations at the symbolic level if word images in the text have been interpreted correctly. Based on the fact that OCR results often violate this consistency, methods of visual consistency analysis are designed to detect and correct OCR errors.

Supposing that we know in advance that the text to be recognized is normal text written in a certain natural language (e.g.: English), high-level linguistic knowledge sources such as lexicography, syntax, and semantics, can be used to detect OCR errors and correct them using alternatives aided by their linguistic context. Here, we focus on the word candidate selection problem. In this approach an OCR provides several alternatives for each word and the objective of postprocessing is to choose the correct decision among these choices. Two approaches of linguistic analysis, statistical and structural, are proposed for the problem of candidate selection. A word-collocation-based relaxation algorithm and a probabilistic lattice parsing algorithm are proposed.

There exist some OCR errors or ambiguities which are not easily recoverable by either visual consistency analysis or linguistic consistency analysis. However, integration of image analysis and language-level analysis provides a natural way to handle difficult words. An interactive model for degraded text recognition is proposed that implements this strategy. In the model, initial word recognition results are provided by an OCR system and they are treated as hypotheses to be tested further; by integrating visual and linguistic consistency analysis, a word hypothesis can be proposed, modified, rejected, confirmed or selected; finally, a decision for each word image is determined.

Under the framework, experiments on OCR postprocessing using visual inter-word constraints and word candidate selection using linguistic and visual constraints are reported. The extension of the approach to Chinese/Japanese text recognition is also discussed.

Contents

Chapter 1

Introduction

The objective of visual text recognition is to correctly transform an arbitrary image of text into its symbolic equivalent. Recent technical advances in the area of document recognition have made automatic text recognition a viable alternative to manual key entry for many applications. Given a high quality text page, a commercial document recognition system can recognize words on the page at a high level of accuracy [22, 125, 126, 124]. However, given a degraded text page, such as a multiple-generation photocopy or facsimile, performance usually drops abruptly [8, 111].

The research presented here addresses problems in degraded text recognition and focuses on how to enhance text recognition by exploiting different high-level knowledge sources and visual contextual information in a text image. We constrain ourselves to the problem of English machine-printed text recognition although the approach can be extended to similar tasks such as handwritten text recognition and machine-printed non-alphabetic-language (e.g.: Chinese and Japanese) text recognition.

1.1 Problem Definition

The task of text recognition can be decomposed into two sub-tasks: *word recognition* and *passage-level postprocessing.* For each word image from a degraded text page, a word recognition algorithm can be applied to generate one or several *candidates* (or so-called "*word hypotheses*"). A word candidate is usually a valid word which can be found in a large dictionary, but sometimes it is just a string which is the best guess provided by the word recognizer. Each word candidate is also provided with a confidence score. Due to the existence of noise, the identity of a word image may be a candidate with a low confidence score, or may not even appear in the candidate set.

The problem to be examined in this research is defined as follows: given a set of candidates for each word image in a text page, determine the correct decision for each word image by passage-level postprocessing, in which visual contextual information and linguistic constraints are exploited. Here, we do not assume the candidate set always contains the correct word. During passage-level postprocessing, word candidates can be re-evaluated, modified or rejected, and new candidates can be proposed.

1.2 Motivation

In a poor quality text page, degradation causes many problems: adjacent characters can touch one another; a character may be broken into several pieces; random noise or ink smears may make a character distorted. With the presence of such problems, for many word images, it is difficult to correctly determine their identities by an omni-font OCR system based only on the visual information from the noisy word images themselves. Figure 1.1.(a) is a fragment of a degraded text page. Figure 1.1.(b) is the output generated by a leading commercial OCR system. There are many recognition mistakes made by the OCR. For

Third was the loss of life at the South African-guarded Calueque Dam project close to the Angolan border, when Cuban Mig-23 jets attacked in June. The latter engagement established the Angolan and Cuban air forces' superiority over South Africa's for the first time.

South Africa's military setbacks resulted in a marked shift in the locus of its strategizing vis-à-vis Angola and Namibia—away from the military hardliners, such as Defense Minister Magnus Malan, armed forces chief Gen. Jannie J. Geldenhuys, and others in the State Security Council, and back to the South African cabinet and the more pragmatic approach of Foreign Minister Pik Botha and Neil Van Heerden, director-general of the Department of Foreign Affairs.

(a)

Third was the loss of lifee at die South
Abican-guarded Calueque Dam protect
close to the Angolarl border, when Cuban
Mig 23 jets attacked in June. The latter
engagement established the Angolail and
Cuban air forces' supenonty over South
Aica's for the list time.
South A&ica's mflitary setbacks reO
Suited in a marked sit m the locus of its
strategizing vis-vis Angola and Nami-
biaaway Coin the military hardliners,
such as defense Minister Magnum Ma-
anned forces chief GexL Janrue J.
Gelderiuys, and others in the State Se-
cunty Council, arid back to the South Af-
rican cabinet and the more praanauc apt
preach of Foreign Minister Pik Botha and
Neil Man Heerden, director-general of
the [)eparanent of Foreign Affairs.

(b)

Figure 1.1: A fragment of text page(a) and its OCR output(b)

example, in the first line, the word "*life*" was recognized as *lifee* and the word "*the*" was recognized as *die*; in the second line, the word "project" was recognized as *protect*; in the third line, the word "*Angolan*" was recognized as *Angolarl*; in the last line of the first paragraph, the word "*first*" was recognized as *list*; etc.

What is shown in Figure 1.1.(b) are the OCR decisions for each word. The OCR results are also represented by the word lattice shown in Table 1.2 (for the first paragraph of Figure 1.1). For each word image, there is an OCR decision. For some word images, there are also lists of suggested candidates in addition to the decisions. Each candidate is associated with a score that measures the confidence of the recognizer in that choice. To generate proper candidates for a word image, lexical constraints have been exploited through word-level postprocessing inside the OCR so that all those candidates are valid words, if possible. For example, there are three word candidates, "*protect*," "*project*" and "*prefect*," for the image "*project*" in the second line. Although the OCR decision "*protect*"

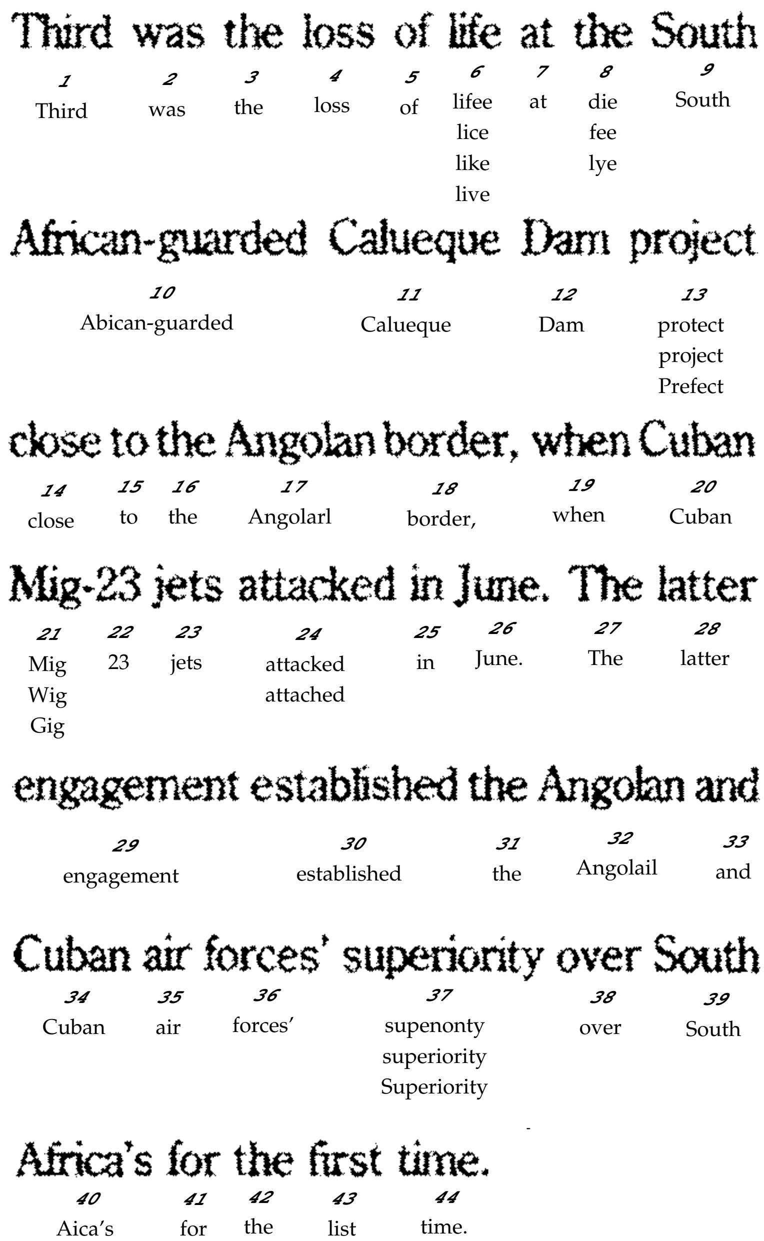


Figure 1.2: Word candidates for the first paragraph in the example page. There are 44 words. For each word, there is at least one word candidate. The first candidate is always the OCR's decision; other candidates, if any, are suggested alternatives to the OCR decision.

is incorrect, the correct word "*project*" is among the suggested candidates.

In current document recognition systems, text recognition is usually treated as a sequence of isolated word image recognitions. Postprocessing in those systems is limited to isolated words (so-called "*word-level postprocessing*"). Therefore, many recognition errors and uncertainties remain unresolved if the text image is highly degraded.

OCR error detection is a difficult problem. Two types of information can be employed to locate those words that might have errors. The first is the low confidence of the character segmentor or the character recognizer. For a word image, if the OCR has difficulty in character segmentation or character recognition, the image is quite possibly recognized incorrectly. The second is the failure of dictionary look-up. Assuming that the text to be recognized is normal English text, if the sequence of characters for a word image is not a valid word defined in a large English dictionary, the word image is probably recognized incorrectly. However, the OCR confidence and the dictionary look-up criterion are sometimes not reliable. For example, a word may be recognized incorrectly although its OCR confidence number is quite high; a word may be mis-recognized as another valid word which can be found in the dictionary. By using the dictionary look-up method, some words, such as proper nouns, foreign words, abbreviations and acronyms, which are actually recognized correctly may be marked as suspect because they are not in the dictionary.

Even if an OCR error is detected, correcting it is still difficult. Generally, there are three types of word recognition errors:

1. The OCR decision is incorrect, and the correct result is among the other candidates;
2. The OCR decision is incorrect, and no other suggested candidates are provided;
3. The OCR decision is incorrect, and the correct result is not among the alternatives provided by the OCR.

To correct the first type of error a postprocessing method has to find a way to select the correct candidate in the candidate list to replace the OCR decision. Candidate selection is still a difficult problem. It is more difficult to correct the second and the third types of errors because the correct candidate has to be proposed and selected. Current OCR systems usually have no capability to correct these types of errors. After locating the suspicious words, the systems usually let users verify them manually.

To detect and correct word recognition errors automatically, "*word-level postprocessing*" is not sufficient. More global contextual constraints have to be exploited. Depending on the window size of context to be considered, those constraints can be the ones between adjacent words, within a sentence, a paragraph, a text page or an article, According to the characteristics of contextual constraints, two classes of them, *linguistic/symbolic constraints* and *visual constraints* have been identified. For example, in Figure 1.2, the candidate "*superiority*" for image 37 can be decided by considering the linguistic context formed by its adjacent words. Similarly, the candidate "*project*" will be more preferred as the decision for word image 13 if its linguistic context is used. Visual context can also be helpful for postprocessing. For example, the recognition error of image 8 can be detected and corrected by checking the recognition results of its visually equivalent images, such as images 3, 16, 31 and 42. Similarly, recognition errors of images 10 and 40 can be detected because their left parts are very similar at the image-level and quite different at the symbolic level. In this reseach, we generally call those contextual constraints beyond isolated words as *passage-level contextual constraints*.

As you can see in the paragraph above, *passage-level postprocessing* techniques are necessary for this purpose. A system with passage-level postprocessing has to select one word candidate as its decision for each word image so that the decision best fits its context.

Passage-level postprocessing uses contextual information available within the passage to improve word recognition performance. In a given text image, there exists abundant contextual information beyond the isolated words. Contextual information sources can be classified into two types: *visual contextual information* and *high-level linguistic knowledge*. Since the 1980s, there have been a few experiments that used passage-level postprocessing for visual text recognition. In those experiments linguistic knowledge was the major source of contextual information used. Those experiments showed that linguistic contextual information can improve OCR accuracy significantly. However, they also showed that there exists significant areas for further improvement.

1.3 Objectives of This Research

The primary goal of this research is to investigate contextual constraints that can be useful for visual text recognition. This research has identified two types of contextual constraints, visual and linguistic, as useful constraints for passage-level postprocessing.

The second goal of this research is to explore how to use the identified visual and linguistic contextual constraints effectively in passage-level postprocessing for visual text recognition. For visual inter-word constraints, several types of visual inter-word relations are defined and algorithms to calculate them are presented. Different uses of visual relations are discussed. For linguistic contextual constraints, two typical classes among them – the statistical model of word co-occurrence and language syntax information, are investigated. To achieve high accuracy for both theoretical and practical consideration, a computational framework in which visual and linguistic constraints are integrated needs to be outlined and tested.

1.4 Outline of the Dissertation

The remaining nine chapters of this dissertation discuss background information, define the proposed approach, and discuss in detail the portions of an interactive model. This is followed by conclusions and future directions for this research.

Visual text recognition is an application area of image processing, pattern recognition and natural language processing. In Chapter 2, research in several related areas is surveyed. The focus is on current methods of postprocessing.

Chapter 3 presents a computational framework for visual text recognition. The outline of an interactive model for passage-level postprocessing is proposed. In contrast to the traditional approach, postprocessing, according to the model, is not only a symbolic process, but also a visual process. Visual inter-word constraints and linguistic knowledge sources are integrated in the model.

Chapter 4 discusses visual contextual constraints available in text pages. Six types of visual inter-word relationships are defined. The algorithms to compute them are described and their potential uses are discussed. The principle of consistency between visual and symbolic inter-word relations is identified as the major principle to guide passage-level postprocessing described in the model. Chapters 5 and 6 reports two experiments on improving OCR performance by using visual inter-word constraints. In the first experiment, word equivalence constraints are exploited in a consensus voting algorithm to correct OCR errors on a large portion of a given text page. In the second experiment, partial similarity constraints, together with equivalence constraints, are utilized to correct errors in all words from a page.

The applications of linguistic knowledge sources in the model are discussed in Chapters 7 and 8. Passage-level postprocessing is formalized as a problem of word candidate selection. Chapter 7 presents a relaxation algorithm in which one type of statistical constraint, word collocation, is exploited for candidate selection. Chapter 8 describes the design of a probabilistic word lattice parser for candidate selection. In the parser, statistical and structural constraints are combined. In those experiments, initial word recognition results were provided by a word shape analysis algorithm.

Chapter 9 discusses the integration of visual and linguistic constraints for visual text recognition. In Chapter 10, the approach is extended to two ideographic languages– Chinese and Japanese.

Chapter 11 summarizes the approach presented in this dissertation, and derives conclusions from this work. In Chapter 12, future directions of the work will be discussed.

Chapter 2

Background

Visual text recognition is an application area of image processing, pattern recognition and natural language processing. In this chapter, research in several related areas is surveyed.

2.1 Text Recognition

To automatically recognize the text in a document image, the first step is page layout analysis [6, 19, 109, 155]. The process of layout analysis detects and extracts text blocks and text lines. Each text line can be further segmented into a sequence of word images. Because of the space between words in machine-printed English text, it is not difficult to find word boundaries. After layout analysis, a sequence of word images is extracted for recognition.

Words are the basic units of text. Interpreting each word image correctly is the goal of text recognition. The process of word recognition generates one or more candidates for each word image.

2.1.1 Text Recognition through Isolated Word Recognition

Text recognition can be accomplished by a sequence of isolated word recognitions. Given a word image, there are basically two approaches towards isolated word recognition. One is the analytical approach, or "character-based word recognition"; another is the holistic approach, or "word shape recognition" [144].

2.1.1.1 The OCR Approach

By taking the analytical approach, word recognition can be accomplished by a three-stage process: character segmentation, character recognition, and postprocessing [79]. The approach is traditionally called OCR(*Optical Character Recognition*), which has a long history and is the dominant paradigm in current text recognition research [106].

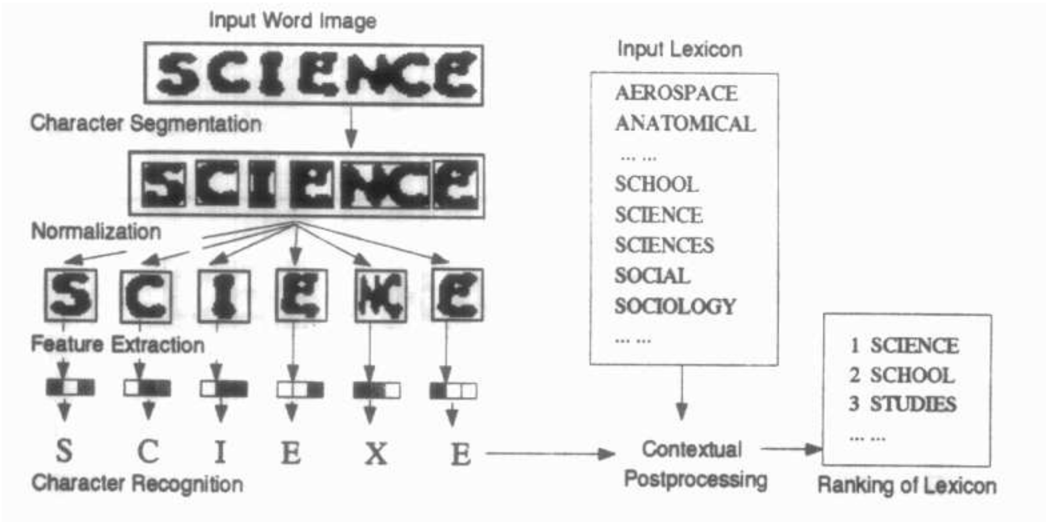


Figure 2.1: Word recognition by using OCR (T.K. Ho 1992)

Character recognition is the key component of OCR-based word recognition. Given a character image, a character recognition algorithm generates decisions about the character's identity. This can be done by template matching or structural analysis. To recognize any of the hundreds of typefaces in common use, current omni-font OCR engines take the structural analysis approach [9, 11, 79]. Different features, including various statistical and structural aspects, have been invented to concisely describe the structure of characters in various typefaces and font sizes [79, 106, 143]. Many classification methods, such as nearest-neighbor classifiers [32], Bayesian classifiers [34], neural network classifiers [94, 132], decision tree classifiers [105] and syntactic classifiers [41], have been designed to classify character images. While very high accuracy has been achieved on high-quality character images, the performance of OCR on poor-quality character images is still not satisfactory [8, 111].

Improving performance on poor quality documents is a challenging problem to which much OCR research is now devoted. New feature representations and new classification methods have been proposed [51, 111]. The role of training sets for classification has been investigated and it has been suggested that the quality of training sets, rather than classification methodology, is the determining factor in achieving higher accuracy [52]. By taking advantage of local typeface homogeneity, OCR accuracy can be improved [9]. The combination of several classifiers has been shown as a way to make recognition more accurate [50, 164].

Besides problems of isolated character classification, improper character segmentation has been identified as one of the major sources of incorrect recognition [18]. As an important preprocessing step, character segmentation partitions word images into sequences of character images so that OCR techniques can be applied [17]. The performance of a text recognition system can depend heavily upon the performance of its character segmentation step.

The simplest method of character segmentation is the use of the small space between characters as a segmentation point. This strategy may not work well when there are touching or broken characters, which often occur in degraded text images such as multiple-generation photocopies or facsimiles. In this situation, several characters may be segmented as one character image or one character image may be split into two or more pieces.

Many methods have been developed to improve character segmentation accuracy. Many current methods are based on connected component analysis, aspect ratio estimation, or profile analysis [11, 35, 79, 93]. There are also some methods that integrate character segmentation with character recognition under the belief that segmentation decisions are tentative until confirmed by the successful recognition of segmented image pieces [18, 154].

2.1.1.2 Commercial OCR Systems

Since 1992, the Information Science Research Institute (ISRI) of University of Nevada has conducted an annual test of OCR systems [125, 126, 124]. Given a binary image of any document page, an OCR system, also called as "page reader", identifies the machine-printed characters on the page. In a test, the accuracy of each system is measured by comparing its OCR result with the truth text. In 1995, about 1,500 sample pages, with millions of characters and with various print quality, were scanned from business letters, technical documents, magazines and newspapers at different resolutions (200dpi, 300dpi and 400dpi). Many companies, such as Caere Corp., ExperVision, Xerox Imaging Systems,Inc. and Recognita Corp., participated in the tests in the past four years. Test results show that current OCR systems can achieve high accuracy (more than 95% correct rate at the word level) on good quality document pages. But it was also reported that the accuracy of the OCR systems declined dramatically when the images are degraded or the resolution of the images was reduced from 300 to 200dpi.

Current OCR systems use lexical constraints and domain knowledge to detect and correct character recognition errors [79, 11]. In OCR systems such as Caere's OmniPage, Calera's WordScan Plus, ExpertVision's TypeReader and XIS's TextBridge, for the characters or words on which the recognition results are not confident, suspicion marks are provided so that the user can verify and edit them manually with a graphical interface [147, 38] (see Figure 2.2 for an example). To verify a marked word, the user usually has to take a look at the original word image and its linguistic context.

Figure 2.2: Correct an OCR error in a "*pop-up verifier*" (Calera's WordScan Plus). For a marked word in the OCR result, the verifier can display the original word image so that the user can verify the recognition result easily. Here, a word is marked because it includes uncertain characters or it can not be found in the dictionary provided by the OCR system or by the user.

2.1.1.3 Word-level Postprocessing

Because OCR is error-prone when the input image is noisy or the typefaces used in the image are not known a-priori to the OCR, contextual postprocessing is usually needed to detect and correct errors in character recognition.

Contextual constraints which have been utilized successfully in practical systems are limited to the word-level [35, 54, 90]. The lexical constraint is the one that is used most widely. Methods of dictionary look-up [158], confusion-matrix-based transformation and string editing [36, 79], probabilistic relaxation [44], and character n-gram frequency analysis [74, 142] have been developed to exploit the lexical constraint efficiently. Such methods work well on non-word error detection. If a string generated by an OCR for a word image does not appear in a dictionary, which can be very large for wide coverage, a non-word error can be detected. Correction of a non-word error finds the dictionary word which is most similar to the string.

However, the dictionary look-up scheme has its limitations: (1) some words without recognition errors may be detected as errors because those words are not included in the dictionary; (2) some word errors may not be detected because they are actually valid words defined in the dictionary; and (3) several alternative words may be suggested by the dictionary because they are very similar. It seems that a complete dictionary can prevent the first type of mistake. But it is impossible to compile a complete list to cover all possible words which can appear in a text. New words are developed continuously, and proper nouns, foreign words, abbreviations and acronyms often appear in text. It is evident that by increasing the size of dictionary the number of these mistakes can be reduced, although the number of mistakes caused by the presence of visually similar alternatives in the dictionary may increase.

For a noisy word image, the OCR result is sometimes nonsense which has several character errors (for example, the image of "*pragmatic*" in Figure 1.1 recognized as "*praanauc*"). Based on such a string, correction suggested by using lexical knowledge is usually misleading. Also, for a detected word error, more than one word candidate might be suggested because all are quite similar to the non-word pattern. In that case, selecting an appropriate candidate is difficult if only lexical knowledge is used. Therefore, many OCR errors cannot be detected or corrected by those methods because word-level constraints are so local. To correct those errors, more global context must be taken into consideration.

Given a noisy word image, ambiguity or uncertainty may arise from any or all of the three stages: character segmentation, character recognition and word-level postprocessing. Character segmentation may give several competitive break points in some regions; character recognition may generate several alternatives for each segmented character image; and postprocessing may provide a set of word candidates to correct a detected OCR error. As a general framework to resolve ambiguity at several levels, a lattice has been identified as a useful structure to keep records of possible segmentation points and character alternatives according to their geometric positions. Various lattice-based methods have been applied [6, 135, 150, 154]. In previous studies, each entry in a lattice is only a representation at the character level.

Sponsored by ARPA (Advanced Research Projects Agency), RAF Technology, Inc. is developing ILLUMINATOR, a new framework for document decomposition, recognition, storage and interchange [119]. It provides a format for breaking down documents into standardized entities, defining entities boundaries and attributes and tagging or labeling their contents and attributes. The Document Attribute Format Specification (DAFS), is being jointly defined by ARPA and RAF, and being promoted by ARPA as a standard [118]. Based on DAFS, ILLUMINATOR also provides tools which make OCR postprocessing more efficient and accurate [117, 119].

2.1.1.4 The Whole Word Recognition Approach

Lexical constraints can be directly exploited in the process of word recognition by applying the word-shape analysis method [55]. In this way, character segmentation is bypassed. Given a word image, its word-shape feature will be directly matched with word-shape features of prototypes for words in a lexicon (see Figure 2.3). The first n best matches will be given as the recognition result. For degraded word recognition, the rate at which the correct word appears in the top-n neighborhood can be high although the accuracy of the top candidate is low. While the method is suitable for specific recognition tasks in which the number of allowable words and the number of typefaces in use are limited, the drawbacks of the approach are also obvious: it requires at least one prototype for each dictionary word or maybe even several prototypes (in different typefaces) for each word. Using word-shape-analysis, a word image cannot be recognized correctly if its identity is not included in the lexicon. Therefore, the approach is not appropriate for the recognition of proper nouns, numbers and foreign words, which often appear in normal text.

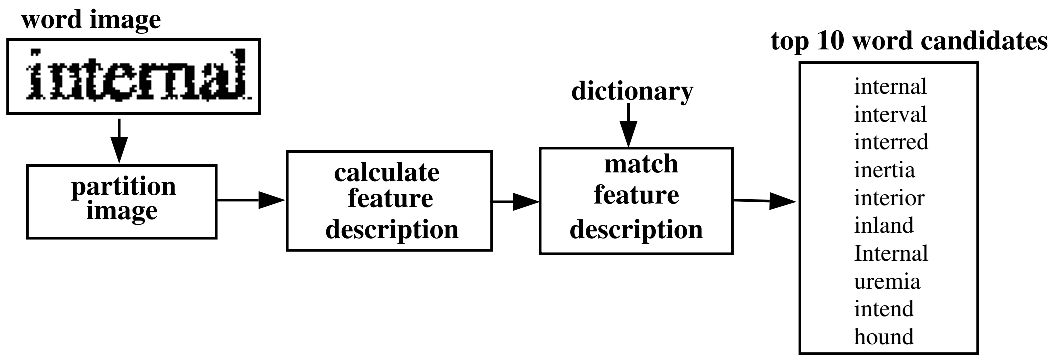


Figure 2.3: Outline of word shape analysis method (T.K. Ho et al 1992)

2.1.1.5 Summary

Isolated word recognition is a popular technique for text recognition. By exploiting the visual information from a word image and lexical constraints, one or more candidates can be generated for each word image. To verify word recognition results and make final decisions, more global contextual constraints have to be utilized.

2.1.2 Text Recognition by Decoding Based on Visual Constraints

Although text recognition is based on word recognition, it is not equivalent to the task of isolated word recognition. Text recognition can take advantage of the typographic uniformity of paragraphs, or other layout components [107]. To utilize visual constraints in a text passage, *image clustering* techniques have been introduced for text recognition. There are two basic classes of methods used for word image or character image based clustering.

Because there are many occurrences of the same words in a text, word images can be clustered based on visual similarity. For each cluster, a prototype can be created to represent the members in the cluster. Then, a recognition algorithm can be applied to recognize the prototypes. If there are several degraded word images in a cluster, the image quality of the prototype for the cluster usually is much better than that of the original noisy word images, so it is easier for a recognition algorithm to recognize the prototype than the original word images [72, 84]. The method has been applied successfully to tasks such as content word detection and font identification [85]. In the word image clustering method, one type of visual inter-word constraint – the word equivalence constraint– is taken into consideration. Use of the constraint with other contextual constraints, especially linguistic constraints, and generalization of the method for other types of visual inter-word constraints are interesting problems we propose for further investigation.

If there are not too many touching or broken characters on a text page, the character images, which usually are individual connected components, can be clustered by image matching or feature vector matching. To find the identity of a cluster so that each character image can inherit the identity of the cluster, there are at least two applicable methods: one is simply to apply a character recognition algorithm; another is to use a deciphering algorithm in which the characters are treated as substitution ciphers and their true values can be solved by a dictionary-based deciphering algorithm [16, 108, 39]. Figure 2.4 illustrates

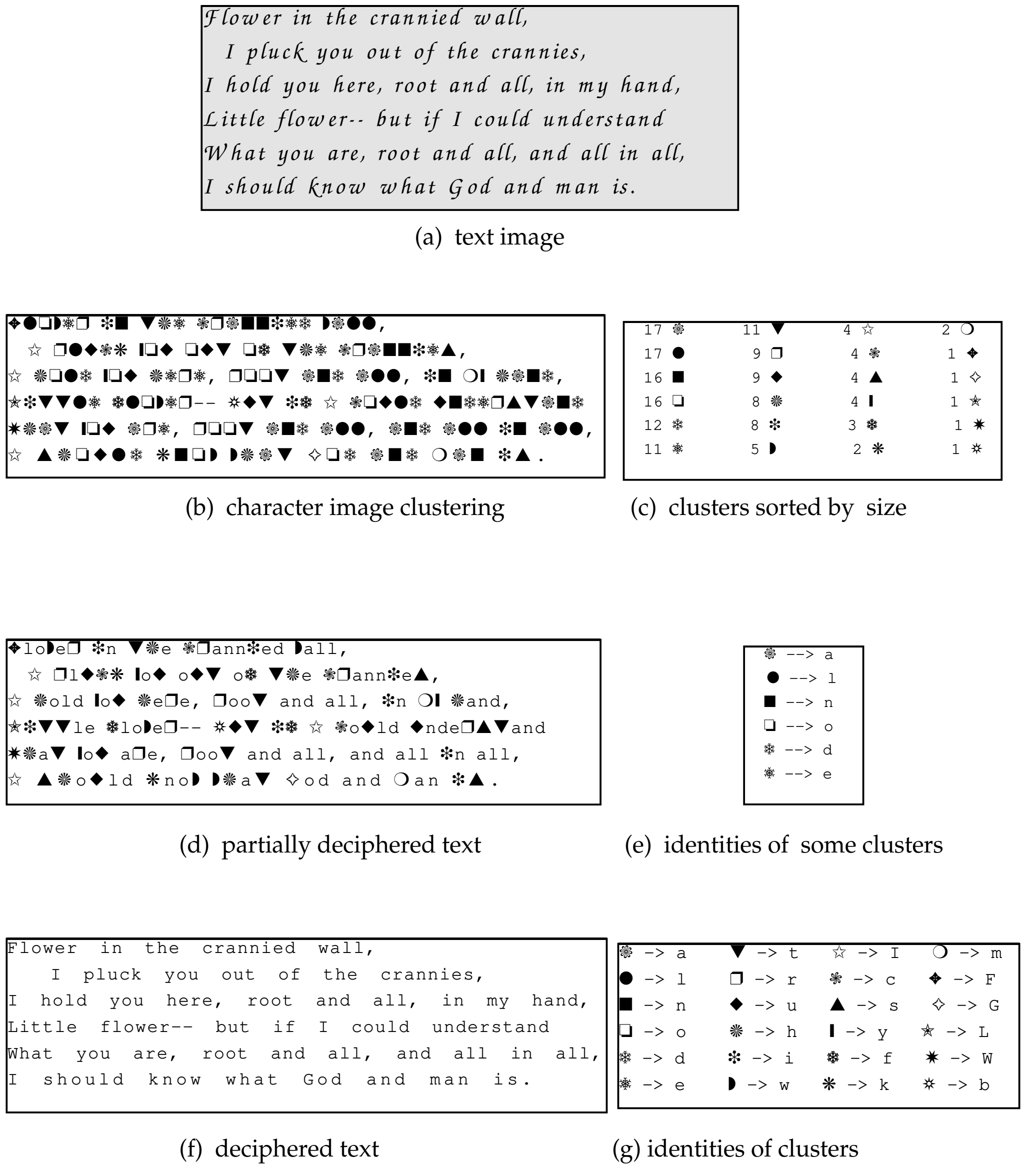


Figure 2.4: An example of text recognition by deciphering. Given a text image (a), character images are clustered based on their visual similarity. In (b), the characters from the same cluster are shown in the same symbol. The size of each cluster is listed in (c). Text recognition here is to find the identity of each character cluster. Lexical knowledge can be helpful to assign identities to clusters incrementally. (d) shows the partially deciphered text after finding identities of some clusters in (e). (f) shows the final result and (g) lists the identities of all clusters.

how the deciphering approach works for a text image. Touching and broken characters, clustering errors, and incompleteness of the dictionary are difficulties that may cause the failure of deciphering algorithms. However, the dictionary-based deciphering algorithm is a nice example which demonstrates how linguistic knowledge and visual contextual constraints can be integrated for text recognition, although the linguistic knowledge used here is limited to the lexical level and the visual constraint is limited to the character level.

Document image decoding using Markov sources is another promising approach that uses visual context[87]. Under this approach, a document image is assumed to be generated by an image generator and a noisy channel. A document image generator is a Markov source (stochastic finite-state automaton) which combines a message source with an imager. The message source produces a string of symbols, or text, which contains the information to be transmitted. The image generator can be modeled as a finite-state transducer which converts the message into an ideal bitmap. The channel transforms the ideal image into a noisy observed image. Therefore, a document recognition system is an image decoder. Given the observed document image, the decoder estimates the message, by finding the *a posteriori* most probable path (so-called "maximum a posteriori (MAP)") through the combined source and channel models using a Viterbi-like dynamic programming algorithm.

2.1.3 Reading by Hypothesis Generation and Testing

Based on previous studies of how people read and the psychology of reading[15, 113, 120], a computational theory for the visual recognition of words of text was proposed [69]. The theory includes three stages: *hypothesis generation*, *hypothesis testing*, and *global contextual analysis*. Hypothesis generation uses gross visual features to provide expectations about word identities. Hypothesis testing integrates the information determined by hypothesis generation with more detailed features that are extracted from a word image. Global contextual analysis provides syntactic and semantic information that influences hypothesis testing.

2.1.4 Document Degradation Modeling

The poor performance of current document recognition systems on degraded images has motivated researchers to study the phenomenon of image degradation. An image page may gain noise from many different sources. The defects may come from the original physical page, from the transformation process (such as photocopying, scanning or facsimile), or from image operations such as digitization. Inside a text page, degradation causes many problems for recognition systems. For example, words near boundaries may have perspective distortion because of how the page was photocopied or scanned from a thick book. Characters may become blurred, distorted, touch together, or split into smaller pieces.

To characterize different sources of degradation on document pages, document degradation models(DDM) have been proposed [7, 8, 81, 80]. Tools based on those DDMs and defect images generated by the methods can be found in the *UW English Document Images Database*, which is available in CDROM form [112]. Those parameterized models can quantitatively describe local or global image defects. They can be used to evaluate the performance of a recognition system for a continuum of degradation levels[53]. Another use of those models is to develop new systems which can explicitly tolerate the noise represented by those models.

To describe the capability of a new method or a new system on degraded text recognition, simply showing its performance on testing samples may not be enough. It is better to explicitly describe the degradation models the new method or system can work well on.

The behavior of an OCR on noisy images may not be stable. It is observed that OCRs are usually sensitive to slight perturbations in their input. If the same page is scanned several times, OCR results for different scanned versions of the same image may exhibit different recognition errors. Consensus voting methods have been applied to improve recognition accuracy by combining outputs of the same OCR on different scanned versions of the same page [95].

2.1.5 Postprocessing Using Passage-level Linguistic Context

Isolated word recognition is error-prone, especially when the input page is seriously degraded. Word-level postprocessing is not sufficient to detect and correct many recognition errors. One or several word candidates associated with confidence scores can be provided by a word recognizer for each word image. For a degraded text page, if a text recognition system just outputs the candidates with the highest scores as the recognition result, like current OCR systems, manual effort is needed to detect and correct those errors.

One technique for improving postprocessing performance is called *context-dependent word correction* [90]. Passage-level linguistic contextual constraints, above the word level, are utilized in this process. Over the last decade, much research has been conducted in the area. According to the ranges of context to be taken into consideration, we can separate contextual postprocessing into several levels (see Table 2.1). In the thesis, passage-level postprocessing is defined as a procedure that uses constraints above the word level.

Figure 2.1 illustrates different levels of postprocessing according to the window size of linguistic context to be considered. Typical methods, constraint types, typical tasks and examples are listed for each level. At the word-level, postprocessing is usually designed to detect and correct character recognition errors and to generate a word candidate list. Constraints used here are limited to lexical constraints. Typical methods are dictionary look-up, character n-gram and string distance editing. At the levels beyond isolated words, the typical task is word candidate selection, that is, for each word with multiple word candidates, choosing the candidate which is most preferred in the context. Given a passage

Level	Typical Tasks	Typical Methods	Constraint Type	Examples
word	character error correction, word candidate generation	character transition, n-gram,relaxation, hidden Markov model, dictionary look-up, string distance editing	lexical	`rnflitary` → `military` `supenonty` → `superiority` `prc?ect` → `protect/project/prefect` `attaclced` → `attacked/attached`
inter -word	word candidate selection	word collocation, POS tag transition, semantic code	lexical, syntactic, semantic	`application farm/form` → `application form` `be/he is` → `he is`
phrase	word candidate selection	hypertag transition phrase parsing	syntactic, semantic	`Mig/Wig -23 jets attacked/attached` ↓ `Mig-23 jets attacked`
sentence	word candidate selection	hidden Markov model, N-best search, sentence parsing lattice parsing	syntactic, semantic, pragmatic	Word Lattice `Places snow me whale Kong Kong it` `Please slow we where Hong Rong is` `Pieces show mo chore Hung Kung Is` ↓ The Best Sequence `Please show me where Hong Kong is`
beyond sentence	word candidate selection	content word recognition by definition overlap	semantic, discourse	Recognition Results for Content Words `pattern cancer chug nurse` `patriot saucer drug noose` `patient corner thug curse` ↓ Candidate Selection `patient cancer drug nurse`

Table 2.1: Levels of postprocessing using linguistic constraints

to be recognized, the difficulty of the task varies, depending on the percentage of word images with multiple candidates. If most of the words have been well-recognized, word candidate selection for a word with multiple candidates becomes easier because it can rely on the context formed by those well-recognized words. If the most of words have multiple candidates (see the word lattice for a sentence in Figure 2.1), the task becomes more difficult because the linguistic context itself is a variable. Another factor that affects the difficulty of word candidate selection is the correct rate of the word candidate lists. The task is much more difficult if the correct rate of the word candidate lists is low, that is, for many words, the correct candidates are not included in the candidate lists.

There are many contextual constraints that can be utilized if we consider information about the relations among words and the structure of documents [77, 144]. Passage-level postprocessing is usually formalized as a candidate selection problem in its simplest case. To select a candidate for each word image, those constraints can play a crucial role. Previous approaches have utilized local word-to-word transitions [68], statistical part-of-speech transitions [70], definitional overlap [40], word collocation constraints [130], transitions between hypertags [141], N-best strategies [31] and semantic constraints from a limited domain, such as *chess* [10]. Post-editing through approximation and global correction was demonstrated as a useful method in which a string with errors can be corrected because it is very close to some well-recognized words(centroids) in the text [148, 149].

Other techniques developed in *natural language understanding* (NLU), such as parsing, can be integrated with text recognition. However this is sometimes difficult because some NLU techniques were developed for restricted domains and cannot process unrestricted English text accurately, efficiently and robustly. We will discuss this problem in more detail later.

2.2 Related Studies on Natural Language Processing

To exploit linguistic knowledge, the task of passage-level postprocessing for visual text recognition is an application of natural language processing. In natural language processing, there are many tasks which are similar or relevant to our task of word candidate selection. They are:

- Part-of-Speech (POS) tagging: selecting the most likely sequence of syntactic categories for the words in a sentence;
- word sense disambiguation: determining the most preferred word sense for a word in its context;
- syntactic parsing: judging whether a sequence of words forms a grammatical sentence.

In natural language processing, there are also other applications which are similar in nature to our task. They are:

- speech recognition;
- spelling correction.

In the section, we will study methodologies in areas of natural language processing and techniques developed for applications.

2.2.1 Statistical Language Modeling

Research from artificial intelligence and computational linguistics has shown that statistical methods can achieve promising performance in the tasks of lexical acquisition [75], tagging [25], parsing [42, 99, 100] and sense disambiguation [140] and that these information sources can be used in systems for natural language processing [161], machine translation [14], information retrieval [88], and speech recognition [160].

The statistical information is usually collected by training on a large text corpus. This approach is also called *corpus-based natural language processing.* The Brown Corpus is an English text corpus with about one million words [89]. It has both a raw and a tagged version. In the tagged version, each word is labeled with its part-of-speech(POS) tag. A corpus can be further annotated at the syntactic or semantic level, for example, each sentence can be associated with a parse tree. The Brown Corpus was built in the mid 1960s. In the last three decades, several more large English corpora have been built, such as the AP Corpus, the British National Corpus and the Penn Treebank [28].

The most typical statistical information directly extracted from a large corpus is word collocation, which has a simple format [26, 27]. Given two words x and y, the strength of their collocation can be measured by mutual information $I(x:y)$ which is

$$I(x:y) = log_2 \frac{P(x,y)}{P(x)P(y)}$$

where $P(x)$ and $P(y)$ are the probabilities of observing x and y in a corpus; $P(x,y)$ is the probability of observing word x and y together inside a window (in other words, the distance between x and y must be no larger than a given window size). Strong word collocation reflects some relation between those two words. It might be some grammatical relation in the language (for example, the word pairs $< to, be >$, $< an, observation >$ and $< set, up >$). It might be some ontological relation that exists in a real or conceptual world (for example, the word pairs $< doctor, nurse >$ and $< wine, drink >$). It might also be some idiosyncratic relation which exists but is difficult to explain(for example, the word pairs $< strong, tea >$ and $< powerful, car >$). As Church pointed out, there are a variety of interesting linguistic phenomena can be identified by word collocation information [26].

Word collocation can be used as a contextual constraint for word candidate selection in visual text recognition and speech recognition [26]. For example, it may be difficult for an OCR to distinguish between "`farm`" and "`form`," where the context is either: (1) "`federal ___ credit`" or (2) "`some ___ of`." Word collocation provides the fact that "`farm`" is much more likely in the first context and "`form`" is much more likely in the second.

2.2.2 Natural Language Parsing

In artificial intelligence and computational linguistics, there have been many efforts to build computational models and practical systems for natural language processing (NLP). Different theories have been proposed to model syntactic, semantic, pragmatic, and discourse aspects of language [3, 162]. Natural language processing systems usually contain several essential parts, such as a lexicon, a grammar, and a parser.

A lexicon contains knowledge about words in a language. Words are the units on which more complex language structures, such as phrases, sentences and discourses, are based. Any description about higher-level structures, either syntactic structures or semantic structures, must be projected from the lexical descriptions of the involved words (Chapter 7 in [116]). Therefore, lexical knowledge is necessary for any natural language processing system. In recent years, there have been many efforts to create machine-readable and machine-accessible lexical knowledge bases in computational linguistics [75]. It is almost impossible to build a large-scale machine-accessible lexicon manually. To overcome the lexical bottleneck, research has proceeded in two directions. One is to compile a machine-accessible lexicon based on on-line lexicons. Another is to extract lexical knowledge from large corpora by using statistical methods.

Almost every NLP system has a grammar and an associated parser. A grammar is a finite specification of a potentially infinite number of sentences, and a parser for the grammar is an algorithm that analyzes a sentence and assigns one or more structural descriptions to the sentence according to the grammar, if the sentence can be characterized by the grammar. The structural descriptions are necessary for further processing, for example, for semantic interpretation.

In the last three decades, many frameworks have been developed to model the syntactic phenomena of English [12, 116, 136, 157]. Among them are *context-free grammars* (CFG), *transformational grammars* (TG), *lexical functional grammars* (LFG), *generalized phrase structure grammars* (GPSG) and *government-binding* (GB) theory. In an NLP system, a grammar is usually represented as a rule base. A grammar can be a set of simple rules as in a CFG, or a set of complex feature-based rules as in a GPSG. Because syntactic phenomena are complex, a grammar with comprehensive coverage can be difficult to obtain.

Many parsing methods have been developed in the last thirty years, such as ATN parsers [163], generalized LR parsers [13, 151], deterministic parsers [101], unification algorithms [83] and chart parsers [82, 166]. Among these techniques, chart parsing is quite popular. The chart in a chart parser is a data structure that stores the information about syntactic structures (or subtrees) already derived from an input sentence. With such a data structure, the same structure will be derived only once. Compared with other parsing algorithms, the chart parsing method is efficient and its worst-case upper bound run time is proportional to the cube of the number of words in the sentence.

Given a sentence, a parser may produce one or more parse trees according to the grammar. Because of *lexical ambiguity*, many sentences are ambiguous at the syntactic level. For example, "*Fruit flies like apples*" is an ambiguous sentence. *Prepositional phrase(PP) attachment* and *conjunction scoping* are two typical sources of syntactic ambiguity. By using semantic information, such as common sense knowledge and domain knowledge, many ambiguities can be resolved [49, 56, 57].

Because of the nature of natural language processing, parsing methods developed so far sometimes are not accurate, robust, or efficient [153], especially when the length of the sentence to be parsed is long (i.e., on average more than 25 words). This places restrictions on applying NLP techniques in many practical tasks, such as unrestricted text analysis, because sentences are usually long in texts from books, newspapers, journals, or technical reports.

Methods for sentence parsing can be generalized to word lattice parsing. As defined by M. Tomita [152], a word lattice is a set of hypothesised words $\{W_1, ..., W_n\}$. Each word W_i is 3-tuple $< b, e, w >$, where b is a begine time, e is an end time and w is a word name. Figure 2.5 gives an example word lattice. Given a word lattice, a lattice parser tries to find a sequence of adjacent words which is valid according to a given grammar. Augmented with semantic and pragmatic knowledge, a lattice parser can select candidates which are not only a grammatical sequence, but also a meaningful one. Lattice parsers were designed for word candidate selection in speech recognition [23, 48, 152]. For example, "`I saw the man`" can be genegrated by a lattice parser as the best word sequence for the word lattice in Figure 2.5. A parsing process sometimes may fail because of the incomplete grammar used in the parser or the ungrammatical constructions within the text or speech to be processed. Methods for repairing parser failures, such as partial parsing, have been reported [47, 56, 102, 128].

2.2.3 Integration of Statistical and Structural Approaches

Statistical and structural approaches have often been applied to different tasks. Many speech recognition systems use statistical language models exclusively, whereas many natural language understanding systems are symbolic. In recent years, there have been many efforts to combine the two approaches in tasks like speech understanding, text understanding and machine translation [4, 100, 103, 114, 146].

In a natural language processing system built with a symbolic approach, linguistic knowl-

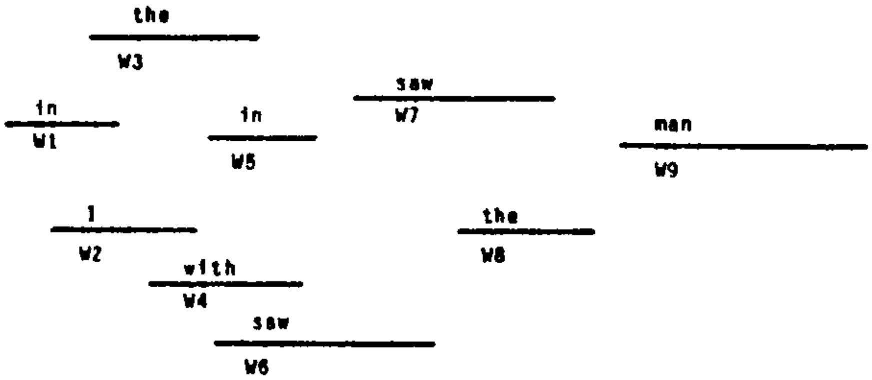


Figure 2.5: An example word lattice (M. Tomita 1986).

edge may be represented by rules. The rules are often written manually, usually with a large human effort. Because phenomena within natural language are very complex and our knowledge about a language is still far from complete, those systems are brittle for real-world tasks.

The statistical approach begins with a statistical language model and automatically estimates the parameters of the model based on real world data. Systems using this approach are more robust, and easier to build and maintain. Statistical language models have been applied to help symbolic systems in many important tasks, such as lexical acquisition, part-of-speech tagging, sentence parsing, sense disambiguation and discourse analysis [20, 98, 123, 122, 139].

2.2.4 The Noisy Channel Model for Speech Recognition, OCR and Spelling Correction

The *noisy channel model* developed in *Information Theory* was originally used to model communication along a noisy channel such as a telephone line. The model can be applied to recognition applications such as speech recognition, OCR and spelling correction [28]. A recognition system can be considered as a noisy channel. Its input is a sequence of good text (I), and its output is usually a sequence of corrupted text(O).

$$I \longrightarrow NoisyChannel \longrightarrow O$$

The goal of postprocessing is to improve the accuracy of the output of the recognition system, O. Given O as input, a postprocessing system tries to output $\widehat{I}$ which is most similar to I.

$$O \longrightarrow Postprocessing \longrightarrow \widehat{I}$$

I is usually unknown. By hypothesizing all possible input texts, $\widehat{I}$ can be computed as the input text with the highest probability, $Pr(O|I)$. Symbolically,

$$\widehat{I} = argmax_I Pr(I|O)$$

where *argmax* finds the argument with the maximum score. $Pr(O|I)$ can be estimated based on statistical language models and the statistical model of the noise channel because different noisy channels have different noise sources (see Table 2.2).

Application	Input	Output
Speech	writer	rider
Recognition	here	hear
Optical	all	a1l (A-one-L)
Character	of	o{
Recognition	form	farm
Spelling	government	goverment
Correction	occurred	occured
	commercial	commerical

Table 2.2: Examples of channel confusions in different applications(Church et al 1993)

In her survey paper [90], Kukich reviewed the techniques for automatic word correction in spell checking and text recognition. Automatic word correction research was viewed as focusing on three progressively more difficult problems: (1) non-word error detection; (2) isolated-word error correction; and (3) context-dependent word correction. The first and the second problems have been studied extensively in the last two decades. But they are partly unsolved because the most difficult cases of those problems are related to the third problem in nature. The third problem is still a research challenge. Since the 1980s, many experiments have been carried out using statistical language models and natural language processing tools.

2.2.5 Speech Recognition

Speech recognition is similar to visual text recognition. The goal of automatic speech recognition is to develop techniques and systems that enable computers to accept speech input [110, 121]. Speech recognition at the phoneme and word level has used *Hidden-Markov-Models(HMM)* [115] and other statistical methods successfully. Practical systems have been built for speaker-independent, continuous, large vocabulary (1,000 words or more) speech recognition in a specified domain, such as an airline travel information service (ATIS) [169]. But speech recognition at the word level is inadequate. Error rates are fairly high even for the best systems such as SPHINX [92]. The SPHINX system has an error rate of 29.4 percent for speaker independent, continuous speech recognition. Normal continuous speech is usually filled with acoustic ambiguity at the level of pure acoustic-phonetic analysis. The acoustic ambiguity may be caused by uncertainty of word boundaries, phonetic ambiguity, syllable omissions and other missing information. Researchers believe that acoustic ambiguity can only be resolved through the use of higher sources of knowledge and NLP techniques.

There have been efforts to integrate speech recognition with NLP [48, 76, 137, 160, 165]. NLP can play two roles. The first role is to interpret the meaning of an utterance. The second, more subtle role, is to reduce acoustic ambiguity by "understanding" the utterance. Besides phonemic and phonologic knowledge, syntactic, semantic, prosodic, pragmatic knowledge sources and knowledge about the real world are applied to speech recognition. Under the approaches of statistical language models and structural analysis, methods such as the N-best search [134], HMM [67] and lattice parsing [23, 48, 152] are designed to select word candidates from multiple choices generated by acoustic-phonetic analysis.

2.3 Conclusions

Text recognition is not only an image processing and pattern recognition task, but also a natural language processing task. When the text to be recognized is highly degraded, it is almost impossible to recognize text simply by applying OCR techniques. For a noisy word image, a recognizer, augmented by word-level postprocessing, can generate a set of potential candidates which usually includes the correct choice. Postprocessing techniques, which depend on more global contextual information, can be applied to detect recognition errors, resolve uncertainty, and finally determine a decision word for each word image.

Current research in passage-level postprocessing is concentrated on using linguistic constraints. Statistical and symbolic techniques developed in natural language processing and in other similar applications, such as speech recognition and spelling correction, can be extended to this task. By formalizing the passage-level postprocessing problem as a candidate selection problem, statistical language models and lattice parsing methods can be applied. As in speech recognition and other natural language applications, integrating the statistical and structural approaches is a promising direction for visual text recognition.

Currently, linguistic contextual constraints are the major sources being exploited in postprocessing for visual text. Another type of constraint is visual context. Character-level visual constraints have been used in deciphering algorithms. Recently there has been some interesting work on using visual word equivalence constraints to improve recognition performance in document images. Other possible visual inter-word constraints have not yet been investigated. The role of visual contextual constraints in passage-level postprocessing should be studied further. Also, integration of visual and linguistic constraints is a new direction that should be investigated.

Chapter 3

A Computational Framework for Degraded Text Recognition

Recently there have been many studies on using linguistic knowledge sources for passage-level postprocessing. Visual contextual constraints in document pages have also been used, however they have not been explored for passage-level postprocessing. In this chapter, a computational framework that integrates visual and linguistic constraints for degraded text recognition is proposed. The function of each component in the model, and interactions between those components are outlined.

3.1 Overall Design

Figure 3.1 is the overall design of the proposed model. The model contains six components. They are: *Layout Analysis*, *Word Recognition*, *Visual Contextual Analysis*, *Postprocessing Using Visual Constraints*, *Postprocessing Using Statistical Language Model and Visual Constraints* and *Postprocessing Using Language Syntax and Visual Constraints.*

Given a text page, the stage of *Layout Analysis* locates text blocks and segments them into sequences of word images. Each word image is input to the *Word Recognition* component to generate an initial word recognition result – a list of word candidates which are

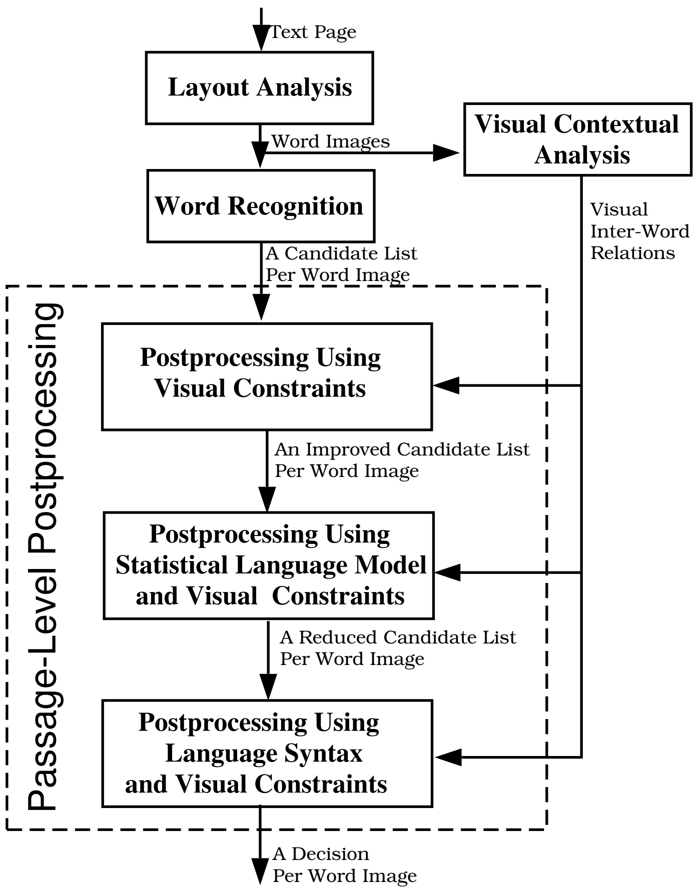


Figure 3.1: The Interactive Model of Text Recognition

ranked by their confidence scores. Word images are also passed to the *Visual Contextual Analysis* component to generate a list of visual inter-word relations which will be used as visual contextual constraints in later stages.

There are then three stages for passage-level postprocessing using visual and linguistic constraints. Candidate lists and visual inter-word relations are the input for these components. In each of these postprocessing stages, the candidate lists for many word images are improved and are input to the next postprocessing component for further improvements. A modified candidate list is usually an improved set of word hypotheses for an image. Here, the improvement can be measured quantitatively by three criteria:

1. The accuracy of the first candidate increases;
2. The correct rate of the candidate list increases;
3. The size of the candidate list is reduced.

During the three stages of passage-level postprocessing, a candidate can be re-evaluated by upgrading its confidence score, or it can be removed from the list if its confidence becomes low. New candidates can be proposed and added to the list. When there are any changes in a candidate list, candidates in the list will be re-ranked according to their confidence scores.

After the stages of passage-level postprocessing, most word images will have only one candidate left in their candidate lists. For each of these word images, this candidate will be its final decision. For the images that still have several competitive candidates, the candidates with the highest confidence will be chosen as the decisions. The sequence of decisions is the final recognition result for the text page.

The implementation of the proposed model is a text recognition system that transforms a document image to its symbolic interpretation. This dissertation concentrates on the stage of *Visual Contextual Analysis* and three stages of passage-level postprocessing. These components can be considered as a stand-alone system for passage-level postprocessing. The layout analysis and word recognition components are prerequisite stages for visual contextual analysis and passage-level postprocessing. However, they are not subjects for this research.

3.1.1 Layout Analysis and Word Recognition

Layout analysis is the stage to find text blocks in a given text image, to segment text blocks into text lines, and to segment text lines into sequences of word images. We assume that, after this stage, the coordinates of the bounding-box for each word image can be provided with high accuracy. This assumption can be satisfied by existing layout analysis algorithms.

For each word image, we assume that a word recognition procedure produces one or several word candidates. Each candidate is associated with a confidence score. A typical OCR algorithm, which includes the steps of character segmentation and character recognition, plus word-level postprocessing, can be used as a word recognizer. Word recognition also can be performed by word-shape analysis under the holistic approach [55].

3.1.2 Visual Contextual Analysis

The visual contextual analysis stage examines visual context by computing *visual inter-word relations*. Visual inter-word relations are phenomena which can be observed from text pages. Inside a text page, we can find many word images that share visual similarity. Word images can be visually equivalent if they match each other at the pixel level. Beside those equivalent images, many images can be partially similar. For example, a word image can occur inside another, larger word image; two word images may have the same left part, or right part; the left part of a word image may be equivalent to the right part of another word image; and so on. Given coordinates of word images from a text page, such visual relations between word images can be analyzed accurately even though the text page is highly degraded. The reason is that although uniform noise sources, such as random noise, can easily destroy the feature structure at the character level, they usually have no major effect on the spatial arrangement of characters inside a word image, so the overall shape of the word image does not change significantly.

Visual inter-word relations reflect language characteristics and typographic characteristics of document images. English is an alphabetic language. Its alphabet set includes 26 lower-case letters, 26 upper-case letters, 10 digits and several punctuation marks. All English words are composed of characters from such a small set that many different words hold one of the relations defined above. For example, many words are actually the inflected forms of a common root made by adding suffixes or prefixes. Syntactic, semantic and pragmatic constraints also cause many words, such as so-called "*function words*" and "*content words*," to appear with high frequency in a natural language text.

Although there are thousands of fonts, there are usually only several of them used in a real document[9]. For those words printed in the same font, typesetting parameters, such as letter space, are usually consistent within a page or an article [91]. The typographic characteristics of document images makes it possible to detect the visual similarity between words.

3.1.3 Postprocessing Using Visual Constraints

The goal of text recognition is to transform word images in a text page into their equivalent strings. The noise on the text page makes it difficult for the transformation to achieve a high level of accuracy if a word image is recognized based only on the information from the word image itself.

A visual similarity between two word images usually implies that their symbolic identities are also similar. That is, if two word images hold a relation at the image level, their truth values should keep the same relation at the symbolic level. This consistency at the image and symbolic levels provides a way to link word images across the text page so that they can be interpreted systematically.

Given a degraded text page, OCR performance is usually quite unstable at the character or word level. Different instances of the same character may be recognized differently within a page. Visually equivalent word images can have different word recognition results. Two word images, which hold a certain relation at the image level, may not have the same relation between their word recognition results.

Visual inter-word relations computed at the visual contextual analysis stage are useful for passage-level postprocessing. Consistency analysis of inter-word relations at the image and symbolic levels can be used to confirm the word images which have been recognized correctly, to locate the word images that are suspected to have errors, to suggest corrections to those suspected images, to propose more appropriate candidates for candidate lists, to remove some inappropriate candidates from candidate lists, etc. For example, if there are several visually equivalent word images and most of them have the same OCR decision, the images with decisions different from the majority decision will be detected as errors and the majority decision will be the correction for those suspected images. Another example: let a word image I_1 be a subpattern of another word image I_2. If I_1's recognition result is a substring of the result for I_2 which is known to be correct, I_1's recognition result can be confirmed.

Another way to utilize visual inter-word relations is to estimate typographic parameters (font and typesetting information) based on visual inter-word relations, then to use them to detect and correct errors. A text recognition system is usually designed for general-purpose use. If font and typesetting information can be learned from document pages, the general-purpose system can be automatically specialized to the input data so that these document pages can be recognized more accurately.

3.1.4 Postprocessing Using Linguistic Constraints

We know that the text to be recognized must be written in a given natural language, such as English. Linguistic research shows that our knowledge about language can be studied from different aspects, such as syntax, semantics and pragmatics. A text is a discourse in which the author tries to say something coherent. It is evident that a text is a hierarchical structure. It is usually composed of several paragraphs. In each paragraph, there may be one or more sentences. A sentence is a sequence of words which starts with a capitalized word and ends with a punctuation mark such as a period or a question mark. As the basic unit to express meaning in a natural language, a sentence is a sequence of words from the lexicon of the language; a sentence is usually legal at a syntactic level, valid at a semantic level, and acceptable at a pragmatic level.

Language-level knowledge can be used as constraint to solve the *word candidate selection* problem for degraded text recognition. Given a text image in which there are several candidates for each word image, the sequence of selected word candidates must form a readable text.

There are two approaches towards the use of linguistic knowledge. One is the so-called "*statistical approach*" under which *statistical language models* can be utilized. Another is the so-called "*structural/symbolic approach.*"

3.1.4.1 Postprocessing Using Statistical Language Model

Word collocation is a simple but powerful statistical language model. By formalizing the word candidate selection problem as an instance of a constraint satisfaction problem [97], we proposed *relaxation(pp. 292-300* in [29]) for word candidate selection. The probabilistic relaxation algorithm uses word collocation trained from large text corpora to re-evaluate the confidence scores of word candidates and to re-rank them based on their new confidence scores. The basic idea of the relaxation algorithm is the use of local word collocation constraints to select the word candidates that have a high probability of occurring simultaneously with word candidates at other nearby locations. The algorithm runs iteratively; in each iteration, the probability of each word candidate is upgraded based on its previous probability, the probabilities of its neighbors, and word collocation data. The initial probability of each word candidate is provided by a word recognizer. The relaxation process terminates when the probability of each word candidate becomes stable. After relaxation finishes, those word candidates with low scores are removed from the candidate sets.

3.1.4.2 Postprocessing Using Language Syntax

To exploit syntactic constraints for the candidate selection problem, a probabilistic word lattice chart parser is proposed. The OCR result can be described as a word lattice, in which word images are organized by their positions according to the reading order, and there can be several competitive word candidates at each position. Assuming the sentence boundaries can be detected correctly, the lattice parser postprocesses OCR results sentence-by-sentence. Given a word lattice W with n word images corresponding to a sentence, where W_i denotes the word image at position i and w_{ij} denotes the *jth* word candidate for image W_i, the lattice parser will find a sequence of word candidates, $w_{1i_1}, w_{2i_2}, ..., w_{ni_n}$ that are involved in the parse tree which has the highest score. The most preferred parse tree for each sentence, which is the by-product of text recognition, can also be output so that further tasks like information retrieval, message understanding and full-text understanding can be carried out.

3.1.5 Integration of Linguistic and Visual Constraints

Visual contextual information are integrated with linguistic constraints inside the relaxation algorithm and the lattice parser to make them work more efficiently and more accurately. In this part, we have discovered how to exploit a previously overlooked synergy between visual and linguistic contexts, in which each each overcomes some limitations of the other. Visual similarity operates over long distances, and so complements the local constraints provided by collocation statistics. And vice versa, in non-obvious ways. Each refines the other.

The process of relaxation depends on the word collocation data collected from large text corpora. Word collocation is a constraint on words that appear nearby each other in a window. The information carried by word collocation is local in nature. Contextual information used in parsing uses structural constraints at the sentence level. Although it is more global than that in relaxation, it is still relatively local compared to a passage of text. Sometimes, those constraints cannot provide enough useful information to choose a word candidate correctly. For example, if there is a sentence "*This [farm/form] is almost the same as that one*," in which the second word position has two word candidates(see the first sentence in Figure 3.2). Both candidates are correct under the word collocation and syntactic constraints. To choose one candidate correctly in this case, we have to consider some passage-level contextual constraints that are more global.

Visual global contextual information provides a way to fill the gap. Word images can be visually correlated whether they are adjacent or not. As we discussed before, relationships at the image level can reflect the corresponding relationship at the symbolic level. Visual global contextual information gives us a way to link the words in a text at the image level and at the symbolic level. For example, in Figure 3.2, word images *2* and *15* are very similar so that they should be recognized as the same word. Both the relaxation algorithm and the lattice parser have to select word candidates which satisfy this constraint. After linking those two sentences using visual constraints, the candidate "*form*" will be selected for word image *2* correctly because of strong language constraints that favored the selection of the candidate "*form*" for image *15*.

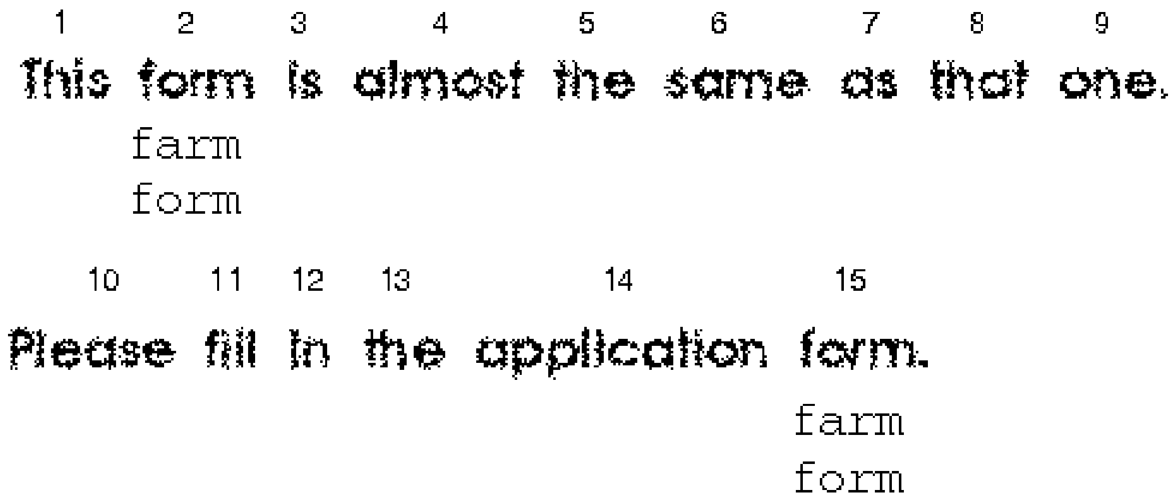


Figure 3.2: Is image 2 *farm* or *form*? (Hint: word images *2* and *15* are very similar at image level.)

Besides the word equivalence constraint shown above, other visual inter-word constraints are also helpful if using together with linguistic constraints.

3.2 The Rest of the Dissertation

The rest of this dissertation focuses on the components of visual contextual analysis and passage-level postprocessing. Algorithms for visual contextual analysis are presented and potential uses of visual inter-word relations are discussed. For postprocessing using visual contextual constraints, two experiments using visual relations to improve OCR performance are reported in detail.

By formalizing the passage-level postprocessing problem as the candidate selection problem, the roles of high-level linguistic knowledge sources are investigated. Under the statistical approach, a relaxation algorithm using word collocation is designed. The experimental result of the relaxation algorithm is described in detail. By taking the symbolic approach, the candidate selection problem is treated as a lattice parsing problem and a probabilistic word lattice chart parsing algorithm is developed for the task. The design of the parser is detailed and its performance is discussed. Inside the lattice parser, statistical language models are integrated with symbolic methods to facilitate parsing.

After discussing potential uses of visual inter-word relations as global constraints, two experiments integrating visual and linguistic constraints for candidate selection are presented in detail. Finally, conclusions and future work are addressed.

Chapter 4

Visual Inter-word Relations and Their Potential Uses

Visual contextual constraints have been identified as useful constraints for text recognition. However, the constraints investigated were limited to *image equivalence constraints* at the character or word level. After clustering character images in a text page, a dictionary-based deciphering algorithm can be applied to find the best interpretation of each cluster so that the character images can be recognized without using OCR [16, 108]. By clustering visually equivalent word images, many function words and content words inside the text page can be found [72, 84].

This chapter studies visual contextual constraints more extensively. English is an alphabetic language. In an English document page, there are usually several hundred words, or several thousand characters, which are printed in a limited number of type-faces. Among these word images, many share certain kinds of visual similarity. Six types of visual inter-word relations are defined. Algorithms that compute these relations are described. Potential applications of visual inter-word relations to such tasks as character segmentation, font detection and word candidate selection are discussed. Unlike character images which are small in size and sensitive to noise, word images are much larger and less sensitive to noise. The existence of visual inter-word relations provides us a way to think of text recognition beyond isolated word recognition, because in this schema word images are related to each other, and therefore can be interpreted systematically at the symbolic level. While language-level knowledge represents experience and expectation, visual contextual information is directly calculated from real images. As we will show in later chapters, integrating visual inter-word relations with traditional OCR techniques and language-level analysis can improve the performance of document recognition significantly.

4.1 Visual Inter-word Relations

4.1.1 Definition

Word images from a text page can be related to each other based on relations defined at the image level. Six kinds of visual inter-word relationships can be defined (see Table 4.1). Examples of them are shown in Figure 4.1.

	Possible Relations between W_1 and W_2	
	at image level	at symbolic level
1 (*equivalence*)	$W_1 \approx W_2$	$W_1 = W_2$
2 (*pattern-occurrence*)	$W_1 \approx subimage\_of(W_2)$	$W_2 = X \bullet W_1 \bullet Y$
3 (*same-left-part*)	$left\_part\_of(W_1) \approx left\_part\_of(W_2)$	$prefix\_of(W_1) = prefix\_of(W_2)$
4 (*same-right-part*)	$right\_part\_of(W_1) \approx right\_part\_of(W_2)$	$suffix\_of(W_1) = suffix\_of(W_2)$
5 (*left-to-right-match*)	$right\_part\_of(W_1) \approx left\_part\_of(W_2)$	$suffix\_of(W_1) = prefix\_of(W_2)$
6 (*same-subregion*)	$subimage\_of(W_1) \approx subimage\_of(W_2)$	$W_1 = X_1 \bullet Y \bullet Z_1$ and $W_2 = X_2 \bullet Y \bullet Z_2$

Note: "$\approx$" means "*approximate match at image level*;" "$\bullet$" means "*concatenation.*"

Table 4.1: Possible word relations at image and symbolic level

The *type-1* relation describes two images that are equivalent. An example of this is shown in case 1 of Figure 4.1. This relation occurs often in normal English text where the same word is used many times. The *type-2* relation defines the occurrence of an image as a *subimage* inside another larger image. The *type-3* and *type-4* relations occur when two images have the same left part and right part respectively, which occurs often because of the use of common prefixes and suffixes. The *type-5* relation defines the match between the left part of an image and the right part of another image. The *type-6* relation generally means that two images share the same image subregion. The first five types of relations are actually special cases of type-6 relations.

Algorithms to calculate these relations from text images will be described in the next section.

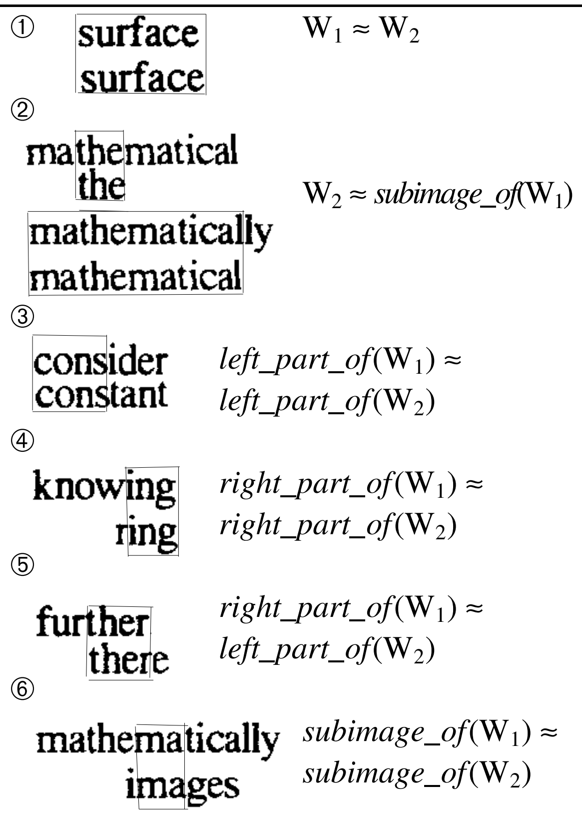


Figure 4.1: Examples of visual inter-word relations

Similarly, inter-word relations can be defined at the symbolic level. Given two word strings, they can be equivalent; one string can be a substring of another string; two strings can share the same prefix or suffix; and so on. Table 4.1 lists 6 types of possible relations which can be calculated by string matching.

4.1.2 Consistency of Relations at Image and Symbolic Level

Inter-word relations at the image level imply relations at the symbolic level. If two word images hold a relation at the image level, their identities should keep the equivalent relation at the symbolic level(see Table 4.1). If two words hold a relation at the symbolic level, they usually have the equivalent relation at the image level if they are printed in the same font with the same typesetting. This is identified as a principle of consistency.

The goal of text recognition is to transform the word images in a text page into their equivalent strings. The noise on the text page makes the transformation difficult to achieve with high accuracy if a word image is recognized based only on the information from the isolated word image. Word relations at the image level, as *visual contextual constraints*, can link the word images in a text page so that they can be interpreted systematically according to the principle of consistency between visual and symbolic relations.

4.1.3 Abundance of Inter-word Relations inside the Text Page

Visual inter-word relations are phenomena which are observed from text pages. They reflect language characteristics and typographic characteristics of visual documents, as we discussed in Chapter 3.1.2. Figure 4.2 is a small segment of text from a real page. In the figure, word image *5* ("*the*") can match the middle part of word image *6* ("*hypothesis*"); word image *9* ("*biological*") can match the left part of word image *1* ("*biologically*"); and word image *8* ("*are*") can match with the right part of word image *11* ("*share*"). Such visual relations between word images can be analyzed accurately even though the text page is highly degraded by uniform noise sources. The reason is that although uniform noise sources such as random noise can easily destroy the structure at the character level, they usually have no big effect on the spatial arrangement of characters or typesetting parameters, so that the overall look of a word image or a part of it does not change much.

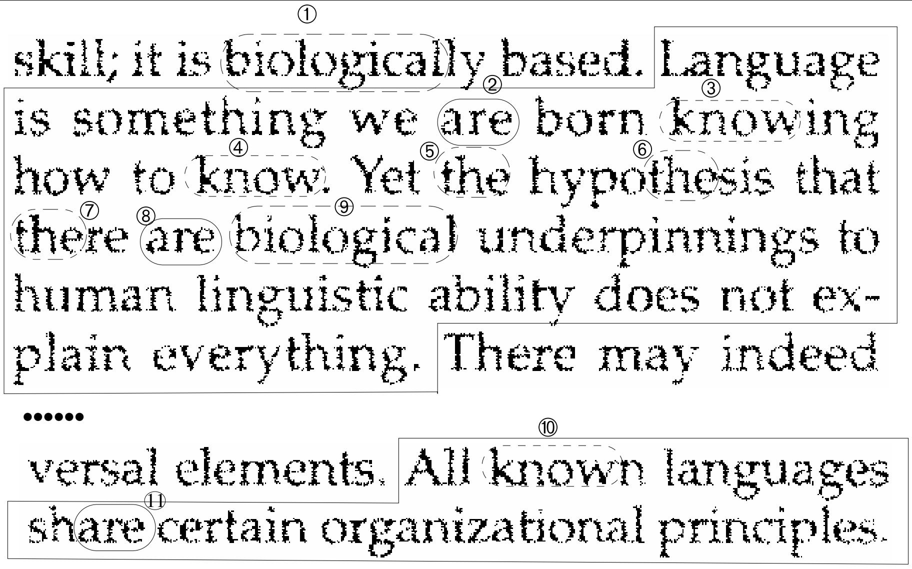
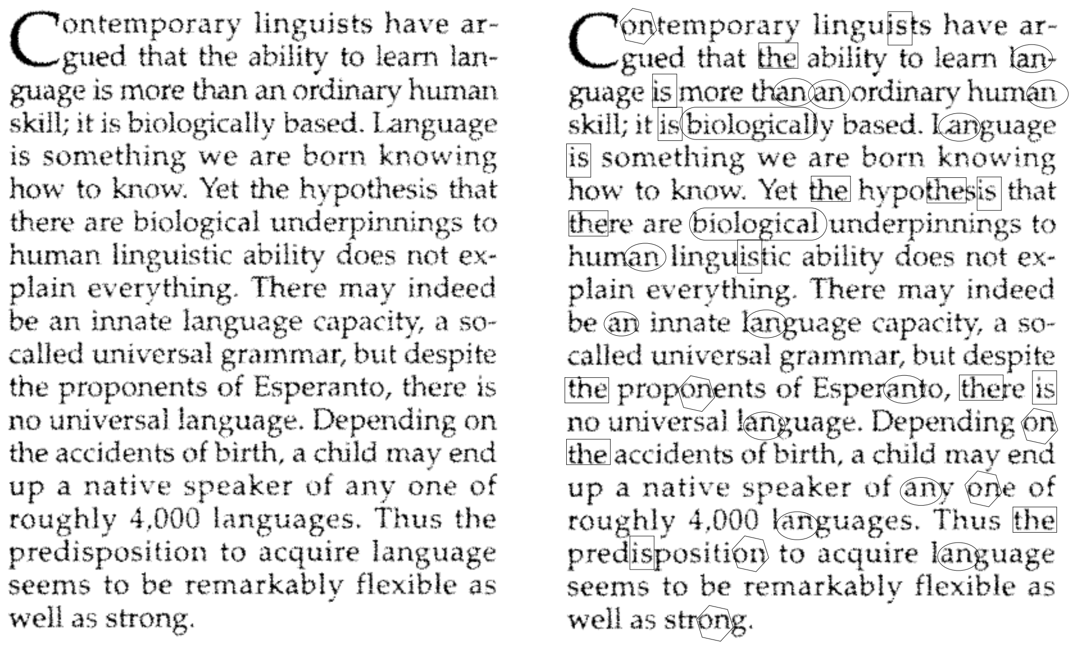

skill; it is biologically based. Language
is something we are born knowing
how to know. Yet the hypothesis that
there are biological underpinnings to
human linguistic ability does not ex-
plain everything. There may indeed
••••••
versal elements. All known languages
share certain organizational principles.

Figure 4.2: Part of text page with three sentences

In an English text, there are many occurrences of the same words. Many of them are function words, such as "*the*," "*of*," "*a*" and "*to*." Those function words are short in length and they often appear inside other longer words as sub-patterns. In Figure 4.3, some short words, such as "*the*," "*an*," "*is*" and "*on*," and their occurrences inside many other words are marked.

4.2 Algorithm Statement

The designs of the algorithms for the six types of visual inter-word relations (see Table 4.1) are described in this section.



Contemporary linguists have argued that the ability to learn language is more than an ordinary human skill; it is biologically based. Language is something we are born knowing how to know. Yet the hypothesis that there are biological underpinnings to human linguistic ability does not explain everything. There may indeed be an innate language capacity, a so-called universal grammar, but despite the proponents of Esperanto, there is no universal language. Depending on the accidents of birth, a child may end up a native speaker of any one of roughly 4,000 languages. Thus the predisposition to acquire language seems to be remarkably flexible as well as strong.

Contemporary linguists have argued that the ability to learn language is more than an ordinary human skill; it is biologically based. Language is something we are born knowing how to know. Yet the hypothesis that there are biological underpinnings to human linguistic ability does not explain everything. There may indeed be an innate language capacity, a so-called universal grammar, but despite the proponents of Esperanto, there is no universal language. Depending on the accidents of birth, a child may end up a native speaker of any one of roughly 4,000 languages. Thus the predisposition to acquire language seems to be remarkably flexible as well as strong.

Figure 4.3: Recurrences of some patterns in a paragraph of text

4.2.1 How to Measure Similarity between Two Binary Images

The visual similarity between two binary images of the same size can be measured quantitatively by how the images match at the pixel level. Let A and B be two $m \times n$ binary images. Inside an image, "1" and "0" denote "*black*" and "*white*" pixel respectively. We measure visual similarity between A and B as

$$r(A,B) = \frac{\sum_i^m \sum_j^n (A_{ij} \wedge B_{ij})}{\sum_i^m \sum_j^n (A_{ij} \vee B_{ij})}$$

where "$\wedge$" and "$\vee$" are *and* and *or* operators respectively. The higher the measurement r, the better two images match. When two images, A and B, are slightly different in size, the similarity between them can be defined by the maximal matching obtained if A is shifted over B. By setting a proper threshold r_0, we can define that two word images are visually equivalent if $r(A,B) > r_0$.

This kind of image comparison based on pixel-to-pixel matching is a time-consuming operation. There are some simple measurements to test whether two images are not visually similar. If two images are significantly different in height or width, we can say that they are different images without pixel-to-pixel matching. Histogram-based image comparison can also be conducted. If the projection histograms of two images on the x and y-axes are similar, image matching as described above will be performed for further testing. If their histograms are different, they cannot be similar visually. In this way, the speed of image matching operations can be improved significantly.

4.2.2 Find Word Equivalence by Image Clustering

If two word images hold a *type-1* relation at image level, they are images which match with each other. Clustering is the grouping of similar objects [46]. A word image clustering algorithm is proposed to extract this kind of relation from a text page(see Figure 4.4). After image clustering, any two word images from a cluster are identified to hold the *type-1* relation at the image level.

It must be pointed out that each cluster can have a prototype which represents images in the cluster. To be more efficient, other types of relations will be computed based on prototypes of clusters. If the prototypes of two clusters hold a relation, it can be inferred that all image pairs from those clusters hold the relation.

4.2.3 Find *Pattern-Occurrence* Relations

The algorithm in Figure 4.5 is for testing whether a word image appears inside another word image as a pattern. An algorithm is described in Figure 4.6 to find *type-2* relations between word images from a text image.

```
extract all word images from the text image;
set QUEUE as an empty set;
put all word images into QUEUE;
set IMAGE-CLUSTER-LIST as an empty set;
while QUEUE not empty do
      pop an image I from QUEUE;
      if image I matches with the prototype of a cluster in IMAGE-CLUSTER-LIST
        then
              add image I as a new member of that image cluster;
        else if
              create a new image cluster with image I;
              add it into IMAGE-CLUSTER-LIST;
      end if
end while
```

Figure 4.4: Algorithm of word image clustering

```
IsSubImage(Image1, Image2, shift) { Image1: m1 × n1 , Image2: m2 × n2 }
            if m1 ≥ m2
              then
                    return (FALSE);
              else if
                    found ← FALSE;
                    for ( x ← 0 ; x ≤ m2 − m1 ; x ++ )
                        extract the region of Image2, start at (x, 0), end at (x + m1, n2);
                        if Image1 and the Region of Image2 match
                          then
                                found ← TRUE;
                                shift ← x;
                                break;
                        end if
            end if
            return (found);
```

Figure 4.5: Algorithm of *pattern-occurrence* matching. Given two images $Image_1$ and $Image_2$, the algorithm tests whether $Image_1$ is a pattern inside $Image_2$. $shift$ is the x-coordinate of the match region in $Image_2$ if the match is successful.

```
Word Image Clustering { see Figure 4.4 for detail };
sort clusters in IMAGE-CLUSTER-LIST by increasing order of the width of their prototypes;
while IMAGE-CLUSTER-LIST has more than one cluster do
        pop a cluster C from IMAGE-CLUSTER-LIST;
        for each cluster D in IMAGE-CLUSTER-LIST do
            if IsSubImage (prototype of C , prototype of D, shift)
              then
                    for each word image x in cluster C do
                        for each word image y in cluster D do
                            x is a subpattern of y;
                        end for
                    end for
            end if
        end for
end while
```

Figure 4.6: Algorithm for extracting *type-2* relations in a text page

4.2.4 Find *Same-left-part* Relations and *Same-right-part* Relations

The algorithm that calculates whether two word images have the same left part (*type-3* relation) is shown in Figure 4.7. To analyze the relations between word images from a text image, an algorithm similar to the one in Figure 4.6 can be designed. In the same way, the algorithm for extracting *type-4* relations can be designed.

4.2.5 Find *Left-to-right-part-match* Relations

The *type-5* relation between images exists if the right part of an image matches with the left part of another image. The algorithm is listed in Figure 4.8.

```
LeftPartMatch (Image_1, Image_2, match_region_width) { find maximal match }
{ Image_1: m_1 × n_1 , Image_2: m_2 × n_2 }
      lower ← 0
      upper ← Min(m_1, m_2)
      { binary search }
      while ( lower < upper ) do
            middle ← (lower + upper)/2
            extract left regions of Image_1 and Image_2 with width middle
            if image regions match
              then
                    lower ← middle
              else
                    upper ← middle
            end if
      end while
      match_region_width ← lower
      if match_region_width > 0
        then
             return (TRUE)
        else
             return (FALSE)
```

Figure 4.7: Algorithm for left-part matching between two images

RightToLeftMatch $(Image_1, Image_2, width)$ { find maximal match };
{ $Image_1$: $m_1 \times n_1$, $Image_2$: $m_2 \times n_2$ }
    $m \leftarrow Min(m_1, m_2)$;
    $found \leftarrow$ FALSE;
    $w \leftarrow m$;
    { *search for maximal match by shift window* }
    while ($(found =$ FALSE$)$ **and** $(w > 0)$) **do**
        extract right regions of $Image_1$ and left region $Image_2$ with width w;
        if image regions match
          then
            $found \leftarrow$ TRUE;
            $width \leftarrow$ w;
          else if
            $w \leftarrow w - 1$;
        end if
    end while
    return $(found)$;

Figure 4.8: Algorithm for right-to-left-part matching between two images

4.2.6 Find *Same-subregion* Relations

It is expensive to directly calculate whether two word images share the same subregion by directly comparing them at the image level. Instead, this relation can be derived from the other relations defined above. Given two word images, if there is another which is a sub-pattern of both of them we can confidently say that those two images hold the *type-6* relation(see Figure 4.9 for the algorithm statement).

Another way to extract the relation requires the step of segmentation. As we will show later, a *type-2* relation between two word images A and B can be used to partition the longer word image B into several parts. For example, the image "*the*" can split the image "*mathematical*" into three image parts, "*ma*," "*the*" and "*tical*." One of those image parts may be a sub-pattern of a word image C. If so, we can say that images B and C share the same pattern. For example, because the image part "*ma*" is a sub-pattern of the image

ShareSameSubPattern ($Image_1, Image_2$)
    if there exists an image x such that
        (**IsSubImage** ($x, Image_1$) = TRUE) **and**
        (**IsSubImage** ($x, Image_2$) = TRUE)
      then
        return (TRUE)
      else if
        return (FALSE)
    end if

Figure 4.9: Algorithm for detecting *type-6* relations based on other relations

"*image*," word images "*mathematical*" and "*image*" have a *type-6* relation.

4.2.7 Improve Efficiency of Algorithms for Visual Relations

The algorithms presented above are slow because they are purely based on pixel-to-pixel image matching. As we discussed before, methods of feature-based matching can be applied to guide the image matching process. The method of shifting an image across another image, one-pixel-by-one-pixel, in order to find the maximal matching area, is also time-consuming. Just like what happened in string matching, much of the computation for matching two slightly shifted regions with a pattern is redundant so it can be avoided. By generalizing fast algorithms designed for string matching (see the Rabin-Karp algorithm, the Knuth-Morris-Pratt algorithm and the Boyer-moore algorithm in [30]), fast algorithms for image matching can also be proposed.

Another promising direction to improve efficiency is to use inter-word relations at the symbolic level to guide the computation of visual relations. Given a text page, an OCR can be applied to generate initial recognition results, which may have many recognition errors. Based on this non-perfect symbolic representation provided by the OCR, hypotheses of visual inter-word relations can be generated by string matching. Then, the image matching algorithms above can be applied to test those hypotheses and the result of hypothesis testing can be used to detect and correct OCR errors. In Chapters 6 and 9, we will discuss the approach more in detail.

4.3 Uses of Visual Inter-word Relations in Document Recognition

Visual inter-word relations in a document page can be utilized in several different ways in document recognition. Here we limit our discussion to their uses for character segmentation, font detection and candidate verification and selection.

4.3.1 Character Segmentation

A novel method has been designed for character segmentation that utilizes visual inter-word constraints from a text image to recursively split word images into smaller image pieces [61]. This method is applicable to machine-printed English text where the same spacing is used between identical pairs of characters.

The intuition behind the method comes from the observation, that, inside an English text (with reasonable length), there are usually many word images that are visually similar. Here, two types of similarity are considered: *pattern-occurrence* and *image equivalence*. For example, in Figure 4.10, a paragraph of a poem by Wordsworth, we can find many visual inter-word relations: word image *6* occurs in part of word image *1*; word image *7* occurs at the right end of word image *3* and the middle of word image *4*; word image *10*, which is the character "`a`", occurs in the middle of word image *9*; word images *11*, *12* and *13* are visually equivalent; and so on.

If image A matches a part of image B, it actually provides one or two segmentation points for image B so that image B can be split into two or three image pieces (see Fig-

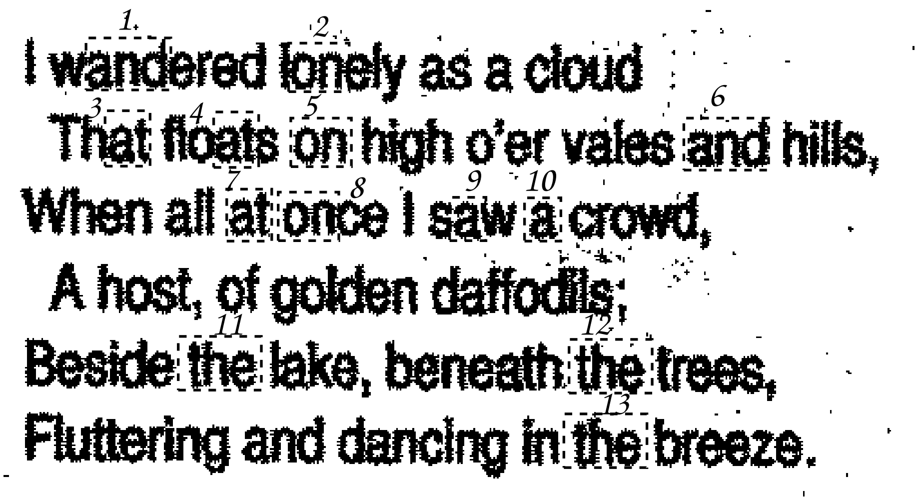


Figure 4.10: An example text block with many visual inter-word relations

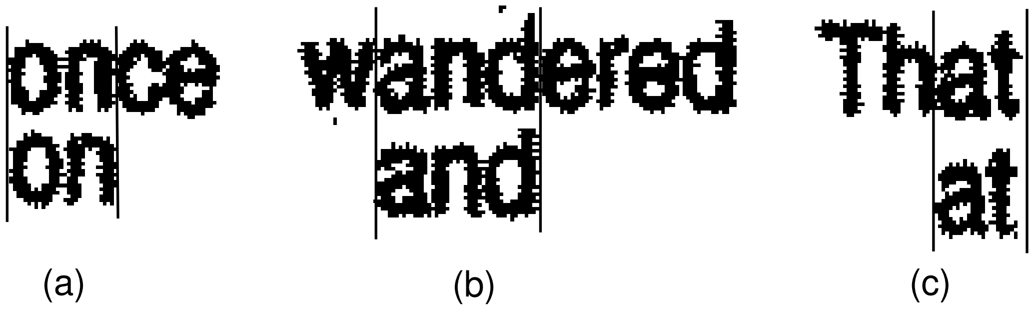


Figure 4.11: Splitting a long word image using another short word image

ure 4.11 (a),(b) and (c) for examples of the different cases). Those image pieces can be used to split other larger images and can themselves can be further split by other smaller images. Figure 4.12 shows examples of how character images can be obtained by one or several steps of such splitting. In Figure 4.12 (a), the character image "`g`" can be obtained from the image "`Fluttering`" using the image "`in`". In Figure 4.12 (b), the character image "`h`" can be obtained from the image "`high`" using the pattern "`g`" that was obtained in the previous step. Character images "`t`" and "`e`" can be further segmented from the image "`the`" by using the image "`h`" in Figure 4.12 (c). This method is based purely on image processing using the visual context in a text page. No recognition is involved.

If images A and B are visually equivalent to each other, they should be the same word and should be segmented in the same way. To speed up character segmentation, it is sufficient to segment one of them and let the others use the same segmentation points.

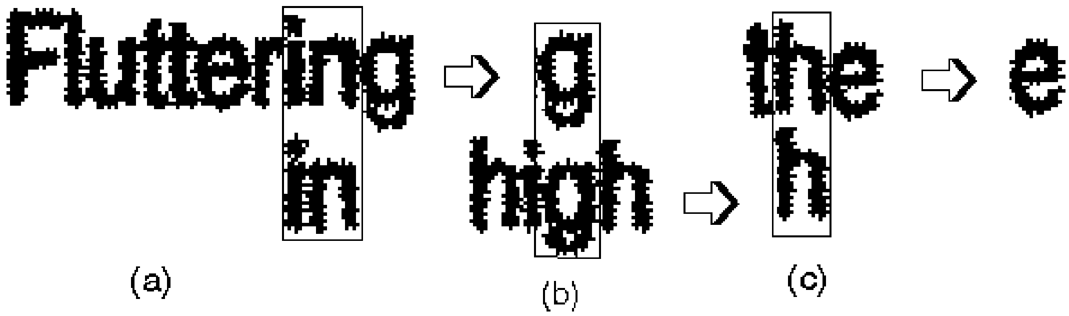


Figure 4.12: Character segmentation by image matching

4.3.1.1 Feasibility Analysis

The proposed method uses *pattern-occurrence* relations to split long patterns repeatly. The success of the method depends on the existence of *pattern-occurrence* relations in normal

English text. The task of character segmentation can be accomplished if such relations are easy to find in a normal text.

```
Plainfield -- James P. Mitchell and Sen.  Walter H. Jones J, last night disagreed
on the value of using as a campaign issue a remark by Richard J. Hughes,
Democratic gubernatorial
```

(a) First 31 words of A06

```
In the past, the duties of the state, as Sir Henry Maine noted long ago, were only
two in number:  internal order and external security.  By prevailing over other
claimants
```

(b) First 30 words of G02

```
In the century from 1815 to 1914 the law of nations became international law.
Several factors contributed to this change.  The Congress of Vienna is a
```

(c) First 26 words of J42

```
Dan Morgan told himself he would forget Ann Turner.  He was well rid of her.  He
certainly didn't want a wife who was
```

(d) First 23 words of N01

```
One day, the children had wanted to get up onto General Burnside's horse.  They
wanted to see what his back felt like -- the General's.  He looked so comfortable
being straight.  They wanted to touch the mystery.  Arlene was boosting them up
when the policeman came
```

(e) First 46 words of R07

Figure 4.13: Fragments of five test articles from the Brown Corpus. By applying the method to each of them, all the words except those highlighted with an underline can be split to the character level.

To determine the number of pattern-occurrence relations that exist in a normal English text, an experiment was conducted with ideal ASCII input. Five articles were randomly selected from the Brown Corpus [89] as test samples for this analysis. They are denoted as A06, G02, J42, N01, and R07. Each of them contains about 2,000 words. The simulation result showed that almost all the words in those samples could be split into individual characters using the proposed approach.

Another analysis estimated the minimal number of words required to correctly segment all the characters in a text sample when ASCII input was used. Figure 4.13 shows text fragments extracted from the beginning of five sample articles. Although there are no more than 46 words in each text fragment, all words, except two dashes and two year-numbers, can be split into characters. This is an encouraging result that shows the abundance of "*pattern-occurrence*" relations. If all those words were printed in the same font, ideally, there would be the same number of "*pattern-occurrence*" relations in the image.

In a normal English text, many words occur with high frequency. For example, a list of some of those words contains "`a, an, and, are, as, at, but, for, in, is, no, not, of, on, or, that, the, to`" and so on. Such short words often occur in other longer words as sub-patterns and therefore can be used to split the longer words at the first step. After obtaining some segmented pieces from the longer words, more words can be split.

In summary, the above discussion shows that the proposed method is applicable to the task of character segmentation for normal English texts because there are many pattern-occurrences in a typical text passage. In next section, we describe an algorithm that computes visual pattern-occurrence relations from a text image. Then a complete description of the character segmentation algorithm is presented.

4.3.1.2 Algorithm Description

Given a text page, there are five major steps in the proposed method for character segmentation:

1. Word image extraction;

```
begin
    Extract word images from page image to generate a word image list
            W = {w_1, w_2, ......, w_n};

    Word image clustering to generate a cluster list
            C = {C_1, C_2, ......, C_m} where C_i = {w_i1, ......, w_ik_i};

    S ⟵ ∅; /* Initialize the prototype list S */
    foreach cluster C_i in C do
            Select an image c_i as the prototype for the cluster C_i;
            c_i.segmentation_point_list ⟵ ∅; /* no segmentation point for the word now */
            S ⟵ S ∪ {c_i};
    endfor

    Q ⟵ ∅; /*Initialize the pattern queue Q */
    foreach word image c_i in S do
            Create an image pattern node p for the word image c_i;
            p.depth ⟵ 0; /* The field depth records the number of steps
                              * to derive the pattern */
            Q ⟵ Q ∪ {p};
    endfor

    /* Segment images in the prototype list S */
    while Q is not empty do
          Get a pattern p from Q;
          foreach word image c_i in S do
                  if p is a sub-pattern of c_i do
                        Add any new segmentation points to c_i's segmentation_point_list;
                        foreach new image fragment f in c_i do
                                if f is large in size do
                                      Create a pattern node q for the fragment f;
                                      q.depth ⟵ p.depth + 1;
                                      Q ⟵ Q ∪ {q};
                                endif
                        endfor
                  endif
          endfor
    endwhile

    /* Propagate the segmentation points of each prototype
    * to word images from the same cluster*/
    foreach word image c_i in S do
            for each word image w in the same image cluster C_i do
                  w inherits c_i's segmentation points;
            endfor
    endfor

    Output segmentation points of each word image;
end.
```

Figure 4.14: Algorithm for character segmentation using inter-word visual constraints

2. Word image clustering;

3. Prototype list generation;

4. Image splitting on prototype list; and

5. Character segmentation for all word images.

The details of the algorithm are described in Figure 4.14. Given a text page, layout analysis extracts images of lines of text and words within lines. After extracting word images, word image clustering is conducted. This step places different instances of the same word image into the same cluster. Because images in the clusters are visually equivalent, they should share the same segmentation points. If one image can be segmented correctly, other images can be segmented in the same way. Therefore, it is sufficient to select an image instance as the prototype for each cluster and apply the proposed method to segment those prototypes. In the algorithm, the prototype list is denoted as S. For each word image in S, there is a segmentation point list which is initially empty.

The algorithm has a pattern queue Q that holds all image patterns used to segment word images in the prototype list S. Each pattern has a field *depth* which tells how many steps were needed to derive it. The queue is initialized by a list of patterns that represent word images in the prototype list S. The *depth* of those patterns is zero.

The character segmentation stage is described in steps 15-29 in Figure 4.14. It is an iterative process. At each iteration, a pattern p is popped from the queue Q to see whether it can be used to split word images in S. If the pattern p can match with a sub-image of a word image c_i in S, the match provides one or two segmentation points for c_i. If those segmentation points are new, they will be inserted into c_i's segmentation point list. A new segmentation point can result in one or two new image segments in the word image c_i (see Figure 4.15 for examples). Those segments will be inserted into the queue Q and they will

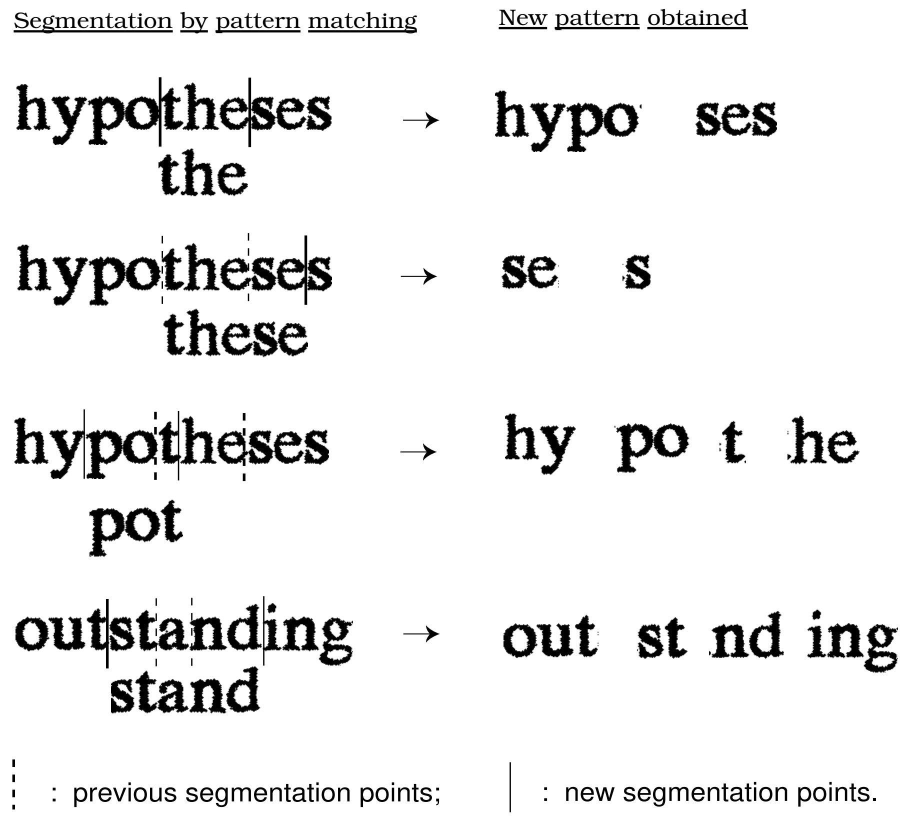
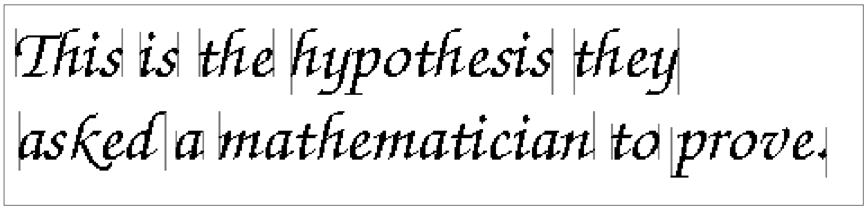
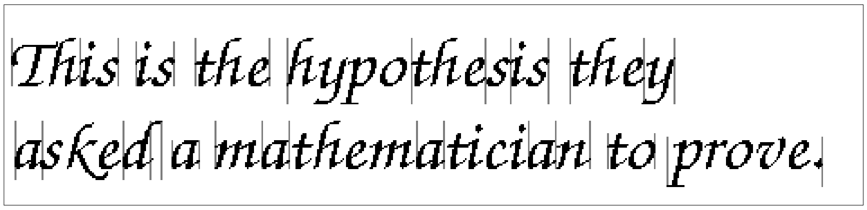


Figure 4.15: Obtain new segmentation points and new image patterns by sub-pattern matching

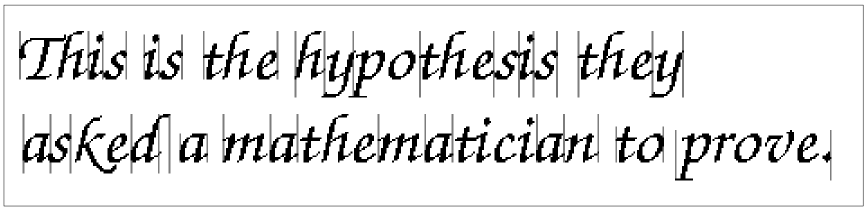
be used as patterns to segment word images in S later. Because it is derived from the pattern p, the *depth* value of a new pattern is set as the *depth* of p plus one. When the queue becomes empty, segmentation stops. Another way to terminate is to set a maximum allowable depth. The algorithm terminates when the depth value of the first pattern in the queue Q becomes larger than this number.



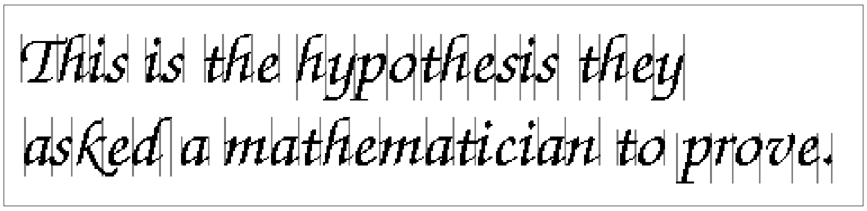
(a) A sentence after word segmentation



(b) After using short words to split long words



(c) After using new obtained segments for further segmentation



(c) Final result of character segmentation

Figure 4.16: An example of running the algorithm

One drawback of the procedure is its fast propagation of potential errors. For example, let's assume that a character image has been split into smaller pieces incorrectly and those image pieces are put into Q. When they are used to segment word images, more wrong segments will be generated and those wrong segments will cause more segmentation errors, and so on. To prevent error propagation a pattern will not be inserted into the queue if it is too small.

After the segmentation steps finish, images inherit the segmentation points of the prototype for each cluster. Finally, segmentation points for each word image are output.

Figure 4.16 is an example that shows the steps that occur when a sentence is processed. The sentence is "`This is the hypothesis they asked a mathematician to prove`"(see Figure 4.16 (a)). There are ten words in the sentence. The occurrence of the word images "`is`", "`the`" and "`a`" is detected in the words "`This`", "`hypothesis`", "`asked`" and "`mathematician`". After segmenting images using these "*pattern-occurrence*" relations, the newly derived segmentation points found are shown as vertical bars in Figure 4.16 (b). Newly obtained image segments, such as "`s`", "`y`", "`m`" and "`n`" are then used to further segment the other images (Figure 4.16 (c) shows the result of using them to further split other words). The final result is shown in Figure 4.16 (d). Most words were correctly segmented to the character level. The only exception is the image piece "`ici`" in the word "`mathematician`". Although several image pieces of "`i`" have been derived previously and they could be used to split "`ici`", they were not inserted into the queue because their size is too small.

4.3.1.3 Preliminary Experimental Results and Discussion

An example of the character segmentation points generated by the algorithm is shown in Figure 4.17. The threshold r_0 for image matching was set to 0.60. Segmentation points located by the algorithm are marked by short vertical lines. Figure 4.18 shows the segmentation result for a paragraph with 110 words. Here, r_0 was set to 0.70. In both paragraphs, although there is space between some characters, this information was not used in segmentation. The algorithm was also tested on a degraded text page with more than 800 words. Similar performance was achieved.

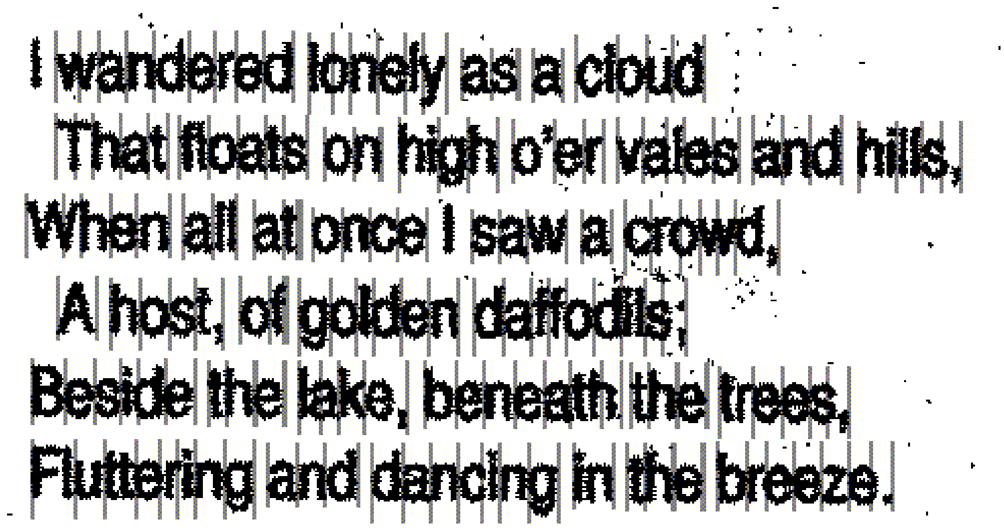


Figure 4.17: Character segmentation result for the text in Figure 1

As shown in Figures 4.17 and 4.18, many words were segmented correctly into characters. However, there are still some image pieces that are under-segmented (i.e., two characters in an image piece), over-segmented (i.e., two image pieces for one character) or mis-segmented. Figure 4.19 shows some typical spurious segmentation points generated by the algorithm on those paragraphs. Typographically, some characters are similar to parts of other characters. For example, "`i`" can match with the right part of "`a`"; "`n`" is similar to part of "`m`"; and "`t`" can match with the left part of "`h`". Incorrect segmentation points can be generated because of partial similarities among different characters. An image piece located by one or two incorrect segmentation points can cause more spurious segmentation points if it is used to split other images. Figure 4.19 (c) shows a segmentation error made by using a wrong

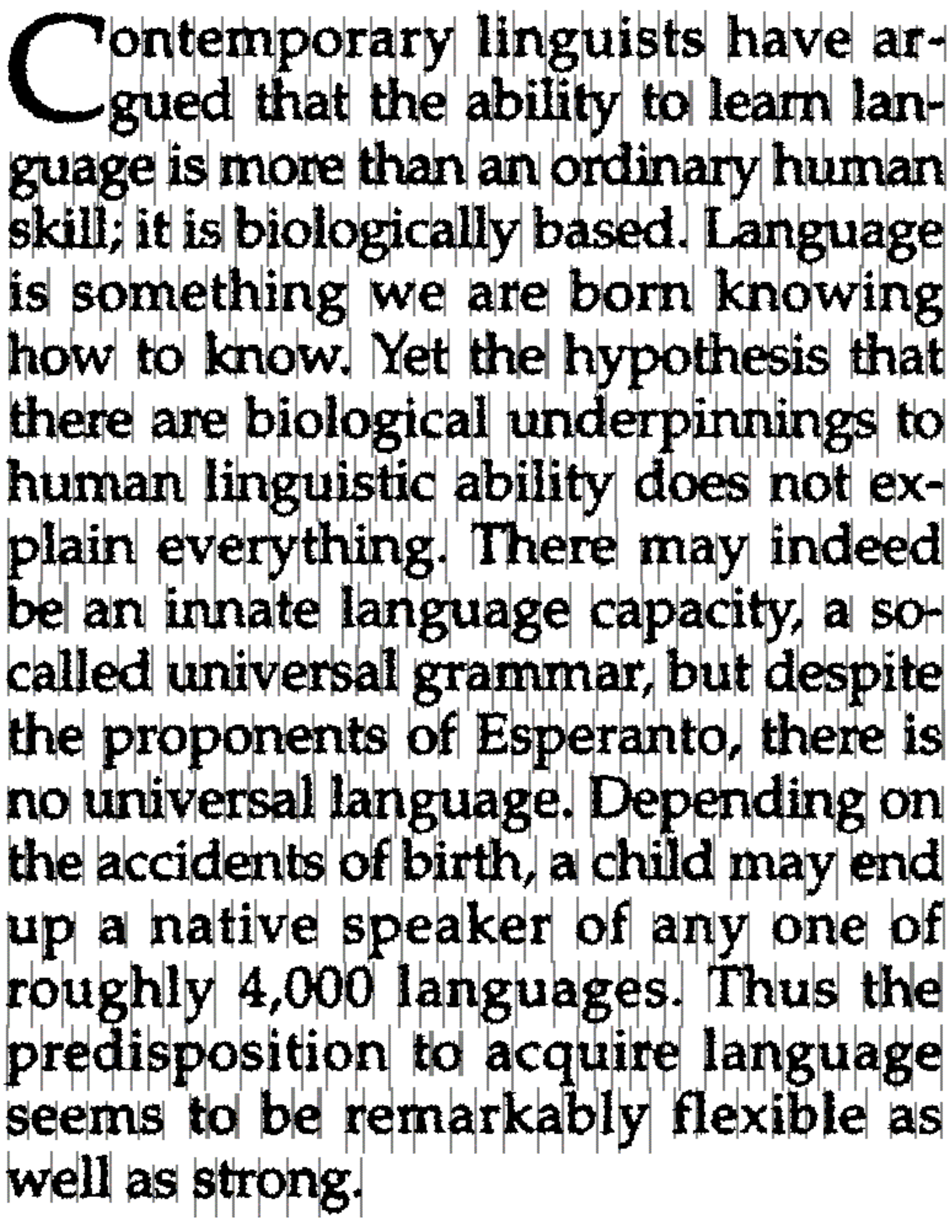

Contemporary linguists have argued that the ability to learn language is more than an ordinary human skill; it is biologically based. Language is something we are born knowing how to know. Yet the hypothesis that there are biological underpinnings to human linguistic ability does not explain everything. There may indeed be an innate language capacity, a so-called universal grammar, but despite the proponents of Esperanto, there is no universal language. Depending on the accidents of birth, a child may end up a native speaker of any one of roughly 4,000 languages. Thus the predisposition to acquire language seems to be remarkably flexible as well as strong.

Figure 4.18: A text block and the result of character segmentation

image piece obtained in (b) of the same figure.

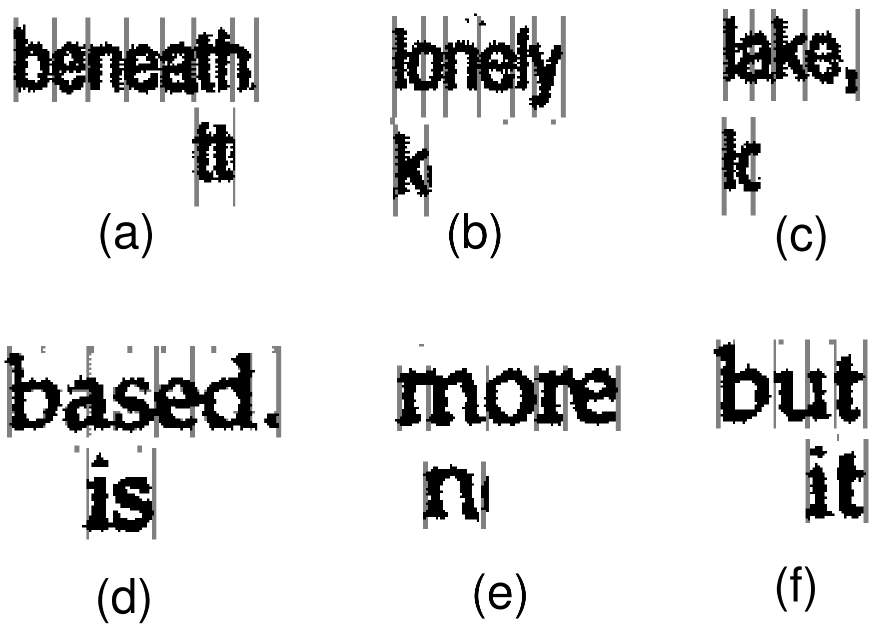


Figure 4.19: Examples of spurious segmentation points generated by the algorithm

Future work could include the integration of the technique proposed here with other methods, such as those based on aspect ratio estimation or profile analysis. For example, after connected component analysis, large connected components can be split into smaller pieces by applying the proposed method. To avoid fast error propagation, character recognition could be applied to test obtained image pieces so that only those image pieces that produce high confidence recognition results are used to split other images.

4.3.2 Font Information Extraction

Previous methods for font detection usually depended on a font library which recorded the prototypes of characters or highly-frequent words in different fonts [85]. The method here does not require a font library. In this method, visual inter-word constraints would be used to cluster the word images which are in the same font. Given a text page which contains words in several different fonts, after the clustering step there will be several clusters, each containing only the word images in the same font.

Visual inter-word relations can be used to infer whether two word images are in the same font. For example, if two word images are almost identical, we can say with a high level of confidence that they are in the same font; if one word image is a sub-pattern of another, we can also confidently say that they are in the same font. The clustering algorithm is described in Figure 4.20. Figure 4.21 illustrates the procedure of font information analysis.

for each word image w_i **do**
    create a font cluster $C_i \leftarrow w_i$
end for
repeat
        if two word images w_m and w_n hold a visual relation
                and $w_m \in C_i$ **and** $w_n \in C_j$
            then
                merge C_i and C_j as C_i
until no more merge

Figure 4.20: Algorithm of font-based image clustering

With same-font image clusters, an omni-font OCR can be applied to extract font information. For example, we can collect prototypes for each character in that font. With such font information, the task of text recognition can be much easier. One advantage of the method is that it provides a way to exploit font information without word recognition and without using a font library. After the clustering process, we can estimate how many fonts are used in the text; and we can know whether any two words are prepared in the some font.

Omni-font OCR can be utilized inside the clustering algorithm to achieve better performance. If two character images are recognized as the same character with a high confidence

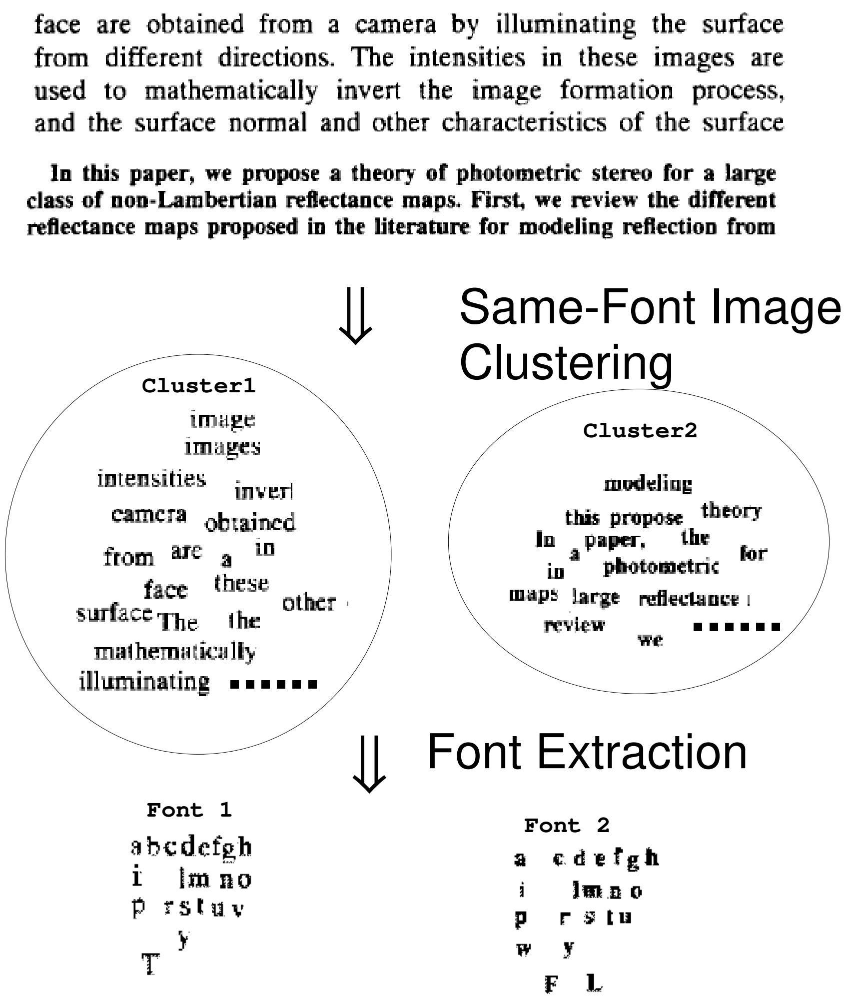


Figure 4.21: Font extraction from text image

value and they are similar at the image level, the word images in which those character images appear can be placed in the same font cluster.

Font information provided by the clustering algorithm can be used to infer a word relation at image level from the relation at the string level. For example, given two word images, A and B, which are clustered in the same font cluster by the algorithm, if a word recognizer generates decision words a and b for images A and B respectively, and we found a and b to hold a relation, we can infer that images A and B hold the same relation at the image level. By returning to image analysis, we can test whether A and B hold the relation. If so, we may have more confidence in saying that a and b are correct. Otherwise, a or b may not be correct and they need to be tested further. In this way, symbolic relations and font information can be used to generate hypotheses about visual relations between word images. Hypotheses can be tested by image processing.

4.3.3 Word Candidate Verification

Given a degraded text image, a word recognizer usually provides several candidates for each word image. For each candidate, it also provides a confidence score. A word recognizer can be an OCR engine plus postprocessing, or a word shape analysis algorithm. It usually uses only the visual information from the word image and knowledge about words. For example, for a noisy word image "*the*," a word recognizer may generate alternatives such as "*toe*," "*the*" and "*tire*." If the word image is noisy, confidence scores for alternatives usually are not reliable so that the correct choice may not be the one with the highest confidence score.

The visual similarity between word images can provide useful constraints to reduce the number of alternatives for a word image. If two word images hold a relation at the image level, their identities should have the same relation at the symbolic level. For example, in Figure 4.2, word images *2* and *8* should have the same identity; the identity of word image

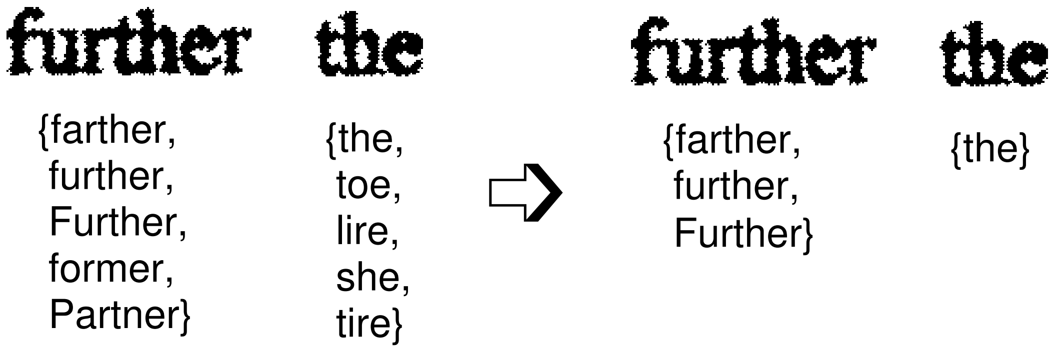


Figure 4.22: Reducing size of candidate sets using inter-word relations

5 ("*the*") should be a substring of the identity of word image *6* ("*hypothesis*"); and so on. Figure 4.22 shows how the size of candidate sets can be reduced for two word images with *type-2* inter-word relation. The size of the candidate set for the first word can be reduced from five to three members and the candidate set for the second word can be reduced from five members to a single word. If a word image holds a visual relation with any other word images, usually its candidate set can be reduced significantly.

4.3.4 Word Candidate Selection

Word candidates are not the final output of text recognition. A decision word must be assigned for each word image so that the output text is readable.

The objective of candidate selection is to find a decision word among alternatives for each word image so that the sequence of word images can be interpreted as sentences, paragraphs or discourses. Although linguistic knowledge sources are useful, sometimes they are not sufficient to determine which candidate should be selected from competitive alternatives for a word image.

Let's consider the sentences in Figure 3.2. Here, all word images except *2* and *16* are assumed to be recognized correctly. We also assume that word images *2* and *16* are visually similar and are classified into a cluster C by image matching. By applying OCR on the image prototype for cluster C, we have two word candidates: "*farm*" and "*form*."

It is difficult to decide whether "*farm*" or "*form*" is the identity of word image *2* if we treat the first sentence in isolation because each candidate makes the sentence meaningful and grammatical. For the second sentence, it is much easier to choose one candidate.

Visual inter-word constraints can be used with linguistic knowledge to make candidate selection work more efficiently and more accurately. In the given context, the word candidate "*form*" will be chosen for word image *16*. Because word images *2* and *16* must share the same identity, we can know that the identity of *2* is "*form*."

4.4 Conclusions

In this chapter we proposed an approach to utilize visual contextual constraints. Six types of inter-word relations were defined and algorithms to calculate them were designed. Their potential uses in text recognition show that it is promising to apply them in order to achieve better performance. The next two chapters will be devoted to experiments in which visual inter-word relations are used to postprocess OCR results. The role of visual inter-word relations as global contextual constraints to facilitate symbolic methods for candidate selection, such as a word-collocation-based relaxation algorithm and a lattice parsing algorithm, will also be discussed in Chapter 9.

Chapter 5

Improving OCR Performance with Word Image Equivalence

In this chapter, a new technique is proposed for improving the performance of an OCR system that uses information about equivalent word images inside a document. Words that are repeated inside a document are grouped into clusters by an image matching algorithm. The decisions of an OCR algorithm about the identities of those words are used to generate a common recognition result for each of the original word images. This technique thus combines information from the document image (word image clusters) with recognition results to correct errors made by OCR systems on different instances of the same word. Experimental results are presented that show about 50% of the words in a document are repeated at least twice. A clustering algorithm is able to reliably locate a large percentage of these words in the presence of noise. Experiments on images degraded with uniform noise show that the correct rate of a commercial OCR system can be improved from 79% to 92% on the words in those clusters. An error analysis is given that shows with further development correct rates in the 98+% range can be achieved.

5.1 Introduction

The objective of visual text recognition is to correctly transform an arbitrary image of text into its symbolic equivalent. Given a high quality text page, current commercial document recognition systems can recognize the words on the page at a high correct rate [22, 125, 126]. However, given a degraded text page, such as a multiple-generation photocopy or facsimile, their performance usually drops abruptly [8, 111].

Figure 5.1 (a) is a fragment of text extracted from a degraded journal page. Such an image can be read correctly by a human but an OCR system makes a significant number of mistakes. The recognition results generated by a commercial OCR system are also shown in the same figure. Figure 5.1 (b) shows the OCR result without using the built-in dictionary and Figure 5.1 (c) is the result with the dictionary.

Figure 5.2 lists some word images and their OCR results with and without postprocessing. For some word images, such as "*WImg.0023*," "*WImg.0061*," "*WImg.0069*" and "*WImg.0077*," recognition results are not correct. By using the dictionary, some errors can be detected and corrected. However, for other word images in the figure, lexical knowledge is not helpful for either detecting or correcting possible errors. For the word images "*WImg.0042*" and "*WImg.0054*," the errors were not detected, and therefore, were not corrected because the recognition results are valid dictionary words. For other images like "*WImg.0007*," "*WImg.0012*," "*WImg.0065*," "*WImg.0071*" and "*WImg.0102*," recognition errors were detected because their postprocessing results are different from the original OCR results. But the decision words were chosen incorrectly among several word candidates suggested by dictionary lookup and string editing.

In this chapter, a method for postprocessing OCR results is proposed that combines visual inter-word constraints with OCR results (see Figure 5.3 for the overall design of the

Third was the loss of life at the South African-guarded Calueque Dam project close to the Angolan border, when Cuban Mig-23 jets attacked in June. The latter engagement established the Angolan and Cuban air forces' superiority over South Africa's for the first time.

South Africa's military setbacks resulted in a marked shift in the locus of its strategizing vis-à-vis Angola and Namibia—away from the military hardliners, such as Defense Minister Magnus Malan, armed forces chief Gen. Jannie J. Geldenhuys, and others in the State Security Council, and back to the South African cabinet and the more pragmatic approach of Foreign Minister Pik Botha and Neil Van Heerden, director-general of the Department of Foreign Affairs.

(a)

```
Third was the loss of lffe at ie South
Abican-guarded Calueque Dam prcect
close to the Angolarl border, when Cuban
Mig 23 jets attaclced in June.  The latter
engagement established the Angolail and
Cuban air forces' supenonty over South
Aica's for the list time.
South A&ica's mflitary setbacks re0
suSted in a marked sit m the locus of its
strategizing vis-vis Angda and Nami-
biaauxy boin the rnilitary hardliners,
sucb as llefense Minister MaBnus Ma-
anned forces chlef GexL Janrue J.
Gelderiuys, and others in the State Se-
cunty Councii, ard back to the South Af-
rican cabinet and the more praanauc aw
proach of Foreign Minister Pik Botha and
Neil Man Heerden, director-general of
the [)eparanent of Foreign Affairs.
```

(b)

```
Third was the loss of lifee at die South
Abican-guarded Calueque Dam protect
close to the Angolarl border, when Cuban
Mig 23 jets attacked in June.  The latter
engagement established the Angolail and
Cuban air forces' supenonty over South
Aica's for the list time.
South A&ica's mflitary setbacks re0
Suited in a marked sit m the locus of its
strategizing vis-vis Angola and Nami-
biaaway Coin the military hardliners,
such as defense Minister Magnum Ma-
anned forces chief GexL Janrue J.
Gelderiuys, and others in the State Se-
cunty Council, arid back to the South Af-
rican cabinet and the more praanauc apt
preach of Foreign Minister Pik Botha and
Neil Man Heerden, director-general of
the [)eparanent of Foreign Affairs.
```

(c)

Figure 5.1: A fragment of a degraded text page and its OCR results. (a) A text block; (b) Its OCR result without using dictionary; (c) Its OCR result with using dictionary

Word	Image	OCR (no Dict.)	OCR (with Dict.)
WImg.0007	the	ie	die
WImg.0012	project	prcect	protect
WImg.0016	Angolan	Angolarl	Angolarl
WImg.0023	attacked	attaclced	attacked
WImg.0039	Africa's	Aica's	Aica's
WImg.0042	first	list	list
WImg.0054	in	m	m
WImg.0061	Angola	Angda	Angola
WImg.0065	from	boin	Coin
WImg.0069	such	sucb	such
WImg.0071	Defense	11efense	defense
WImg.0077	chief	chlef	chief
WImg.0090	and	ard	arid
WImg.0102	ap-	aw	apt
WImg.0103	proach	proach	preach

Figure 5.2: Examples of word recognition errors made by OCR (with or without using dictionary)

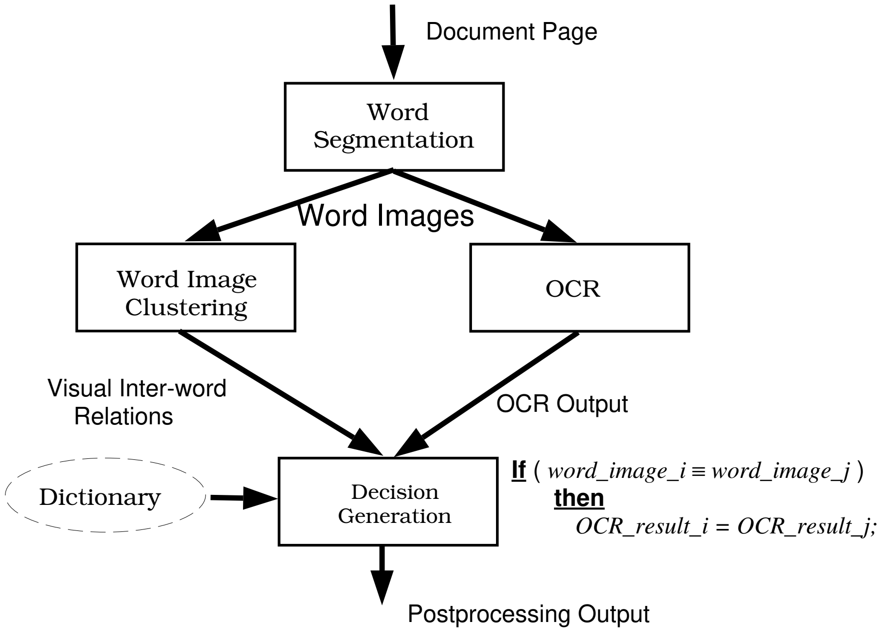


Figure 5.3: Postprocessing OCR results with visual inter-word constraints

method). This approach is based on the following observations about the linguistic and typographic characteristics of document pages and the performance of OCR systems on degraded documents.

- In a text page, there are usually many occurrences of the same words. For example, in a normal English text, many function words and content words occur repeatedly. Because of the fact that the text on a given page is usually printed in a limited number of fonts, images of the same word are usually very similar.

- If two word images are equivalent, their recognition results should be the same. However, a commercial OCR often makes different decisions on different instances of the same word, especially when the document page is seriously degraded (see Figure 5.4). This is understandable since variations in local noise, which may have no significant effect on the overall shape of a word, do make character segmentation and character classification difficult.

Based on these observations, an algorithm is proposed that first locates clusters of equivalent words in a document. A high accuracy rate is hypothesized for clustering since it uses global characteristics of word images that are not affected by local noise. The decisions of the OCR for the words in a cluster are then combined to generate a single consensus decision for these words. This should compensate for the case where an OCR makes different decisions on word images that are the same.

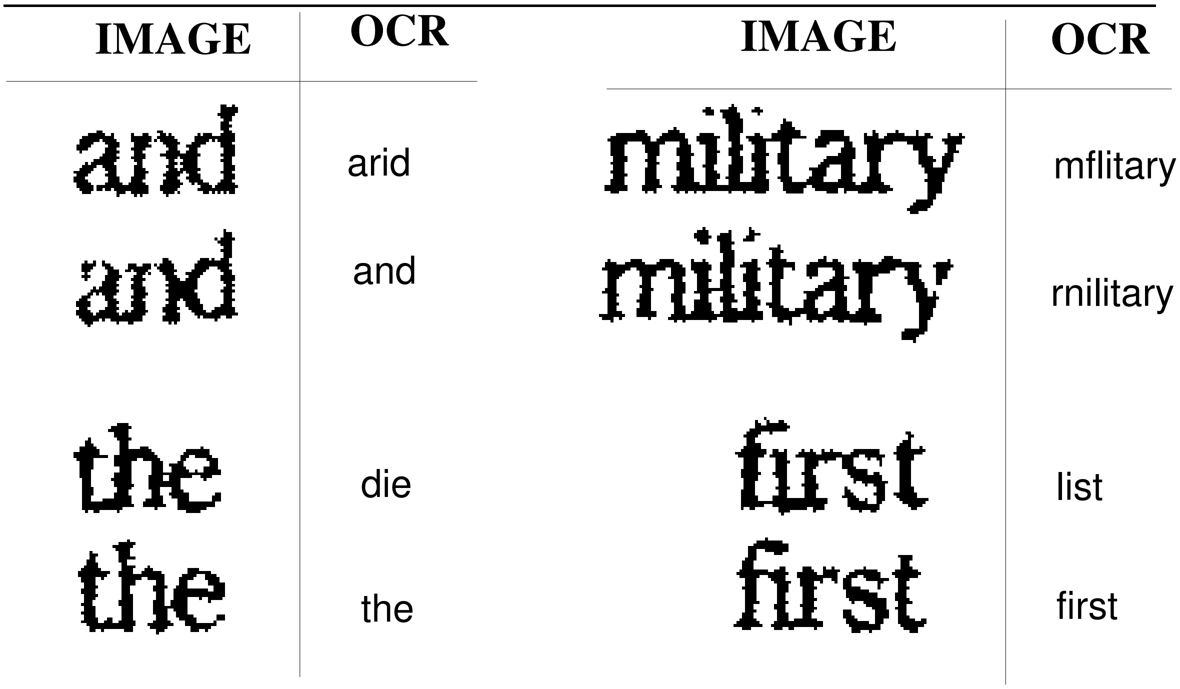


Figure 5.4: OCR sometimes generates inconsistent recognition results on word images which are visually almost the same.

In the rest of this chapter, we present the proposed approach. First we discuss how to measure visual equivalence among word images through word image matching. Then we describe how visual word equivalence constraints are used to postprocess OCR results so that many OCR errors can be detected and corrected. Some experimental results are reported, and finally conclusions and future directions are presented.

5.2 Image Equivalence and Word Image Clustering

The visual similarity between two binary images of the same size can be measured quantitatively by how the images match at the pixel level [73]. Let A and B be two $m \times n$ binary images. Inside an image, "1" and "0" denote "*black*" and "*white*" pixel respectively. We measure visual similarity between A and B as

$$r(A, B) = \frac{\sum_i^m \sum_j^n (A_{ij} \wedge B_{ij})}{\sum_i^m \sum_j^n (A_{ij} \vee B_{ij})}$$

where "$\wedge$" and "$\vee$" are *and* and *or* operators respectively. The higher the measurement r is, the better two images match. When two images, A and B, are slightly different in size, the similarity between them can be defined by the maximal matching obtained if A is shifted over B. By setting a proper threshold r_0, we can define that two word images are visually equivalent if $r(A, B) > r_0$.

Given a sequence of word images from a text page, the visual equivalence among the word images can be computed by word image clustering [84]. After image clustering, images in the same cluster are visually equivalent. After clustering the word images extracted from Figure 5.1, the clusters with multiple images are listed in Figure 5.5.

5.3 Postprocessing OCR Output with Image Equivalence

The postprocessing method is applied to those word images that are equivalent to at least one other image in an input document (i.e, the images that occur in *large* clusters), This method is not applicable to other word images that are included in clusters by themselves.

Figure 5.6 is the outline of the algorithm for postprocessing. For a cluster, if there exists any disagreements among the recognition results for the word images in that cluster, some of the word images in the cluster have not been recognized correctly. After detecting a

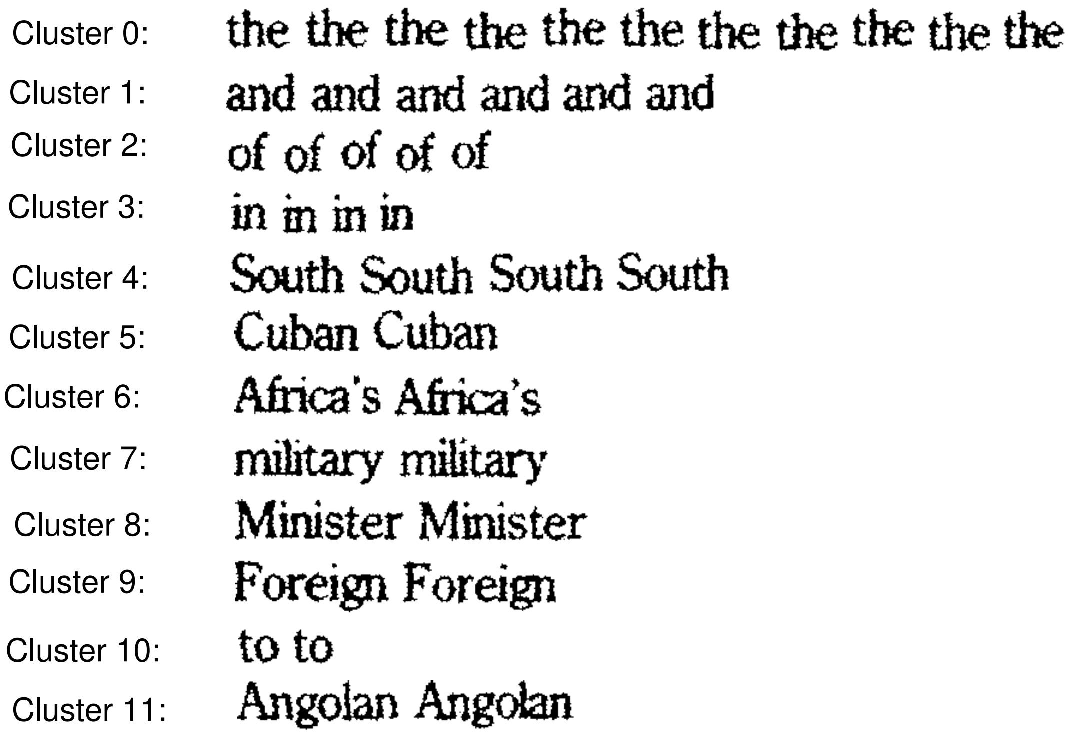


Figure 5.5: Examples of word image clusters.

```
extract all word images from the text image;

word image clustering;

/* error detection through consistency analysis */
for each cluster C, where |C| > 1, do
    if there is disagreement among the images on recognition result
      then
            mark the cluster as inconsistent;
    end if
end for

/* error correction */
for each cluster C, which was marked as inconsistent, do
    select a decision word for the cluster
            using majority voting, dictionary lookup and other criteria;
    all images use the decision of the cluster as their recognition result;
end for
```

Figure 5.6: Outline of the postprocessing algorithm

cluster with such an inconsistency, the error correction step is applied to choose a single decision for all the words in the cluster. There are several criteria for selecting the single decision. They are listed below:

- Majority voting: if there are several images in the cluster and most of them have the same decision as their OCR result, that decision will become the decision for the cluster (see Figure 5.7 .(a));
- Dictionary lookup: if two candidates have the same number of votes, the one which is a valid dictionary entry is preferred (see Figure 5.7 .(b));
- Rejection: if all candidates have an equal number of votes and none of them are valid words, the postprocessing method described here will not select a decision for the cluster (see Figure 5.7 .(c)).

5.4 Experimental Results

Sixteen document pages were selected as testing samples. Among them, twelve pages are from two multiple-page journal articles in CEDAR's journal page database. The remaining four pages are from UNLV's DOE image database which is available on the cdrom released by University of Washington. More details about the testing samples are described in Table 5.1. We list the number of words for each page, and the number of words which have at least another word instance in the page. On average, about sixty percent of words in a page are repeated at least once in some other position.

Noise was added to the original images to simulate the effect of a multiple generation photocopy. We used the University of Washington document degradation model (DDM) [81] to add two different levels of local noise, which are denoted $dd1$ and $dd2$. The parameter settings for $dd1$ are $(420, 0.0, 1.0, 2.5, 1.0, 1.0, 2)$; the parameter settings for $dd2$

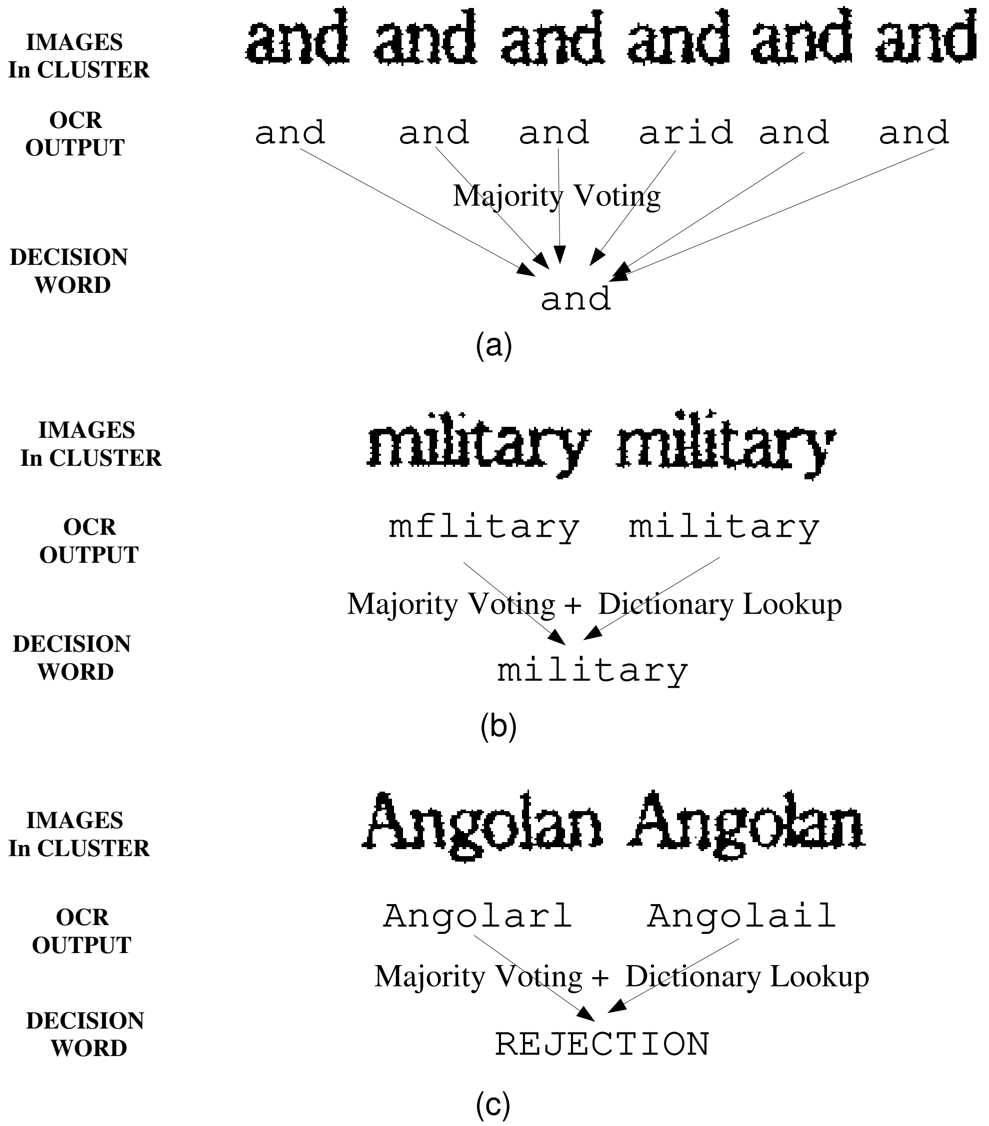


Figure 5.7: Determine a decision for a cluster with multiple image entries. (a). Majority voting; (b). Majority voting + dictionary lookup; (c). Rejection

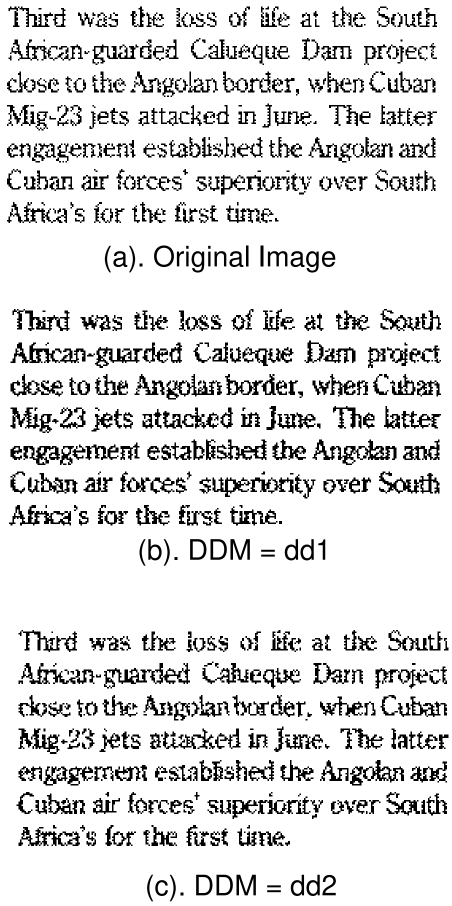
ID	TYPE	NUM OF WORDS	NUM OF REPEATED WORDS	NOTES
A1	Text	822	455	A1-A5 are from
A2	Text	1119	683	a five-page journal
A3	Text	564	355	article AA052532
A4	Text	1230	846	in CEDAR's database
A5	Text	735	444	
D1	Text	350	175	D1-D7 are from
D2	Text	469	217	a seven-page journal
D3	Text	693	406	articles DK074491
D4	Text	835	521	in CEDAR's database
D5	Text	536	260	
D6	Text	683	336	
D7	Text	1021	572	
N1	Reference	856	508	N022BIN in UW's CDROM
N2	Text &Reference	807	477	N036BIN in UW's CDROM
N3	Text	766	515	N03MBIN in UW's CDROM
N4	Text	1052	707	N04FBIN in UW's CDROM
Total		12538	7477	

Table 5.1: Sixteen document pages are used for testing

are $(820, 0.0, 1.0, 1.0, 1.0, 1.0, 3)$. The image generated using $dd2$ is more noisy than $dd1$. Figure 5.8 shows the effect of $dd1$ and $dd2$ on a fragment of an example page.

Caere's *ANYFONT OCR Toolkit* was used to compute OCR results and the coordinates of bounding-boxes for the word images in each document. The coordinates of word bounding-boxes were used in our program for word image clustering. The truth data of the testing samples were generated based on the OCR output on the original images because the Caere OCR performance on those high quality pages was very good (on average, word recognition accuracy is 98.3% and 96.6% according to whether or not the dictionary is used).

For degraded images generated by $dd1$ and $dd2$, the OCR performance dropped significantly in two respects: word segmentation accuracy and word recognition correct rate. For pages with noise at the $dd1$ level, if the OCR's dictionary is turned off the word recognition correct rate is 71.4%. If the dictionary is used, the word recognition correct rate is 80.6%.



Third was the loss of life at the South African-guarded Calueque Dam project close to the Angolan border, when Cuban Mig-23 jets attacked in June. The latter engagement established the Angolan and Cuban air forces' superiority over South Africa's for the first time.

(a). Original Image

Third was the loss of life at the South African-guarded Calueque Dam project close to the Angolan border, when Cuban Mig-23 jets attacked in June. The latter engagement established the Angolan and Cuban air forces' superiority over South Africa's for the first time.

(b). DDM = dd1

Third was the loss of life at the South African-guarded Calueque Dam project close to the Angolan border, when Cuban Mig-23 jets attacked in June. The latter engagement established the Angolan and Cuban air forces' superiority over South Africa's for the first time.

(c). DDM = dd2

Figure 5.8: Degraded document images are generated by using document degradation models (DDM).

For pages with noise at the $dd2$ level, the word recognition correct rate is 60.1% when the dictionary is disabled and 70.9% when it is used.

Word image clustering was performed using the coordinates of the word bounding boxes provided by the OCR package. After clustering all *large* clusters, which contain two or more images, were located. The threshold r_0 was set uniformly as 0.60. About half the word images in a typical page are included in the clusters.

The three different postprocessing strategies were used to select a decision for each large cluster. They are: (1) majority vote; (2) majority voting plus dictionary lookup; and (3) majority voting plus dictionary lookup with the option of rejection. The dictionary used here is the word list collected from the Brown Corpus and the Penn Treebank. There are more than 70,000 words in the list. The best improvement was made by the third strategy.

Tables 5.2 and 5.3 show the performance on the words in large (greater than one member) clusters. On this subset of word images, the improvement of the proposed postprocessing algorithm was significant. For the OCR results on pages with noise at the $dd1$ level and without using the dictionary, the third postprocessing strategy improved the correct rate to 91.5% from the original 78.7%. For the OCR results on pages with noise at $dd1$ level and with using the dictionary, the third postprocessing strategy improved the correct rate to 92.5% from original 86.4%. For the OCR results on pages with noise at $dd2$ level and without using dictionary, the third postprocessing strategy improved the correct rate to 91.1% from original 69.8%. For the OCR results on pages with noise at $dd2$ level and with using the dictionary, the third postprocessing strategy improved the correct rate to 92.3% from original 79.3%;

	OCR		POSTPROCESSING		
ID	OPT	ACCURACY	M	M+D	M+D+R
A1	ND	79.4%(339/427)	88.1%(376/427)	88.8%(379/427)	95.6%(373/390)
	WD	85.0%(363/427)	90.9%(388/427)	91.6%(391/427)	96.0%(387/403)
A2	ND	77.4%(501/647)	91.0%(589/647)	92.6%(599/647)	96.2%(587/610)
	WD	83.6%(541/647)	93.2%(603/647)	94.0%(608/647)	96.3%(595/618)
A3	ND	73.1%(256/350)	74.6%(261/350)	76.6%(268/350)	81.9%(266/325)
	WD	83.4%(292/350)	82.3%(288/350)	83.4%(292/350)	85.6%(290/339)
A4	ND	74.5%(644/864)	77.8%(672/864)	81.1%(701/864)	84.3%(701/832)
	WD	84.3%(728/864)	85.5%(739/864)	87.5%(756/864)	87.9%(756/860)
A5	ND	70.1%(283/404)	70.8%(286/404)	73.3%(296/404)	79.5%(295/371)
	WD	78.5%(317/404)	77.0%(311/404)	80.9%(327/404)	84.5%(326/386)
D1	ND	90.5%(143/158)	94.3%(149/158)	96.8%(153/158)	96.8%(153/158)
	WD	95.6%(151/158)	96.8%(153/158)	96.8%(153/158)	96.8%(153/158)
D2	ND	88.4%(175/198)	94.4%(187/198)	95.5%(189/198)	97.9%(185/189)
	WD	95.5%(189/198)	95.5%(189/198)	95.5%(189/198)	96.9%(187/193)
D3	ND	90.7%(341/376)	92.3%(347/376)	94.4%(355/376)	95.4%(353/370)
	WD	95.2%(358/376)	95.2%(358/376)	95.2%(358/376)	95.7%(356/372)
D4	ND	81.6%(399/489)	91.4%(447/489)	95.3%(466/489)	96.7%(463/479)
	WD	93.9%(459/489)	95.9%(469/489)	96.3%(471/489)	96.9%(468/483)
D5	ND	90.9%(209/230)	94.4%(217/230)	96.1%(221/230)	97.4%(220/226)
	WD	97.4%(224/230)	97.0%(223/230)	97.0%(223/230)	97.0%(223/230)
D6	ND	82.2%(221/269)	88.9%(239/269)	90.3%(243/269)	95.2%(240/252)
	WD	89.6%(241/269)	92.6%(249/269)	92.6%(249/269)	95.4%(248/260)
D7	ND	82.2%(425/517)	90.7%(469/517)	94.4%(488/517)	96.4%(485/503)
	WD	92.5%(478/517)	95.2%(492/517)	95.9%(496/517)	96.9%(493/509)
N1	ND	69.0%(342/496)	74.0%(367/496)	78.2%(388/496)	87.4%(376/430)
	WD	72.5%(361/498)	75.9%(378/498)	80.1%(399/498)	87.8%(387/441)
N2	ND	79.2%(357/451)	86.7%(391/451)	89.6%(404/451)	93.0%(396/426)
	WD	85.6%(386/451)	90.0%(406/451)	90.5%(408/451)	92.6%(401/433)
N3	ND	73.3%(291/397)	80.1%(318/397)	84.9%(337/397)	90.0%(331/368)
	WD	84.1%(334/397)	85.1%(338/397)	87.4%(347/397)	89.8%(343/382)
N4	ND	80.9%(499/617)	85.7%(529/617)	88.2%(544/617)	92.0%(530/576)
	WD	86.7%(535/617)	88.7%(547/617)	89.3%(551/617)	92.5%(540/584)
Total	ND	78.7%(5425/6890)	84.8%(5844/6890)	87.5%(6031/6890)	91.5%(5954/6505)
	WD	86.4%(5957/6892)	89.0%(6131/6892)	90.2%(6218/6892)	92.5%(6153/6651

ND: without using dictionary WN: using dictionary
M: Majority Voting D: Dictionary Lookup R: Rejection

Table 5.2: Postprocessing results on words from large clusters($DDM = dd1$)

	OCR		POSTPROCESSING		
ID	OPT	ACCURACY	M	M+D	M+D+R
A1	ND	66.8%(253/379)	84.7%(321/379)	88.7%(336/379)	94.8%(331/349)
	WD	76.6%(291/380)	89.0%(338/380)	89.5%(340/380)	93.4%(341/365)
A2	ND	64.4%(369/573)	89.0%(464/573)	86.7%(497/573)	97.0%(486/501)
	WD	75.1%(432/575)	88.5%(509/575)	91.5%(526/575)	95.6%(520/544)
A3	ND	70.7%(239/338)	78.1%(264/338)	82.5%(279/338)	87.4%(277/317)
	WD	80.1%(270/337)	84.0%(283/337)	86.1%(290/337)	90.3%(288/319)
A4	ND	69.0%(586/849)	73.4%(623/849)	76.4%(649/849)	83.7%(649/775)
	WD	79.7%(677/849)	84.7%(719/849)	87.4%(742/849)	90.2%(742/823)
A5	ND	63.2%(239/378)	68.5%(259/378)	70.6%(267/378)	76.2%(266/349)
	WD	73.4%(279/380)	77.6%(295/380)	80.3%(305/380)	83.5%(304/364)
D1	ND	87.3%(131/150)	93.3%(140/150)	96.0%(144/150)	98.6%(144/146)
	WD	96.7%(145/150)	97.3%(146/150)	97.3%(146/150)	98.7%(146/148)
D2	ND	79.6%(152/191)	88.5%(169/191)	94.2%(180/191)	96.7%(173/179)
	WD	89.1%(170/191)	93.7%(179/191)	94.8%(181/191)	95.1%(176/185)
D3	ND	84.7%(322/380)	92.1%(350/380)	95.8%(364/380)	96.5%(363/376)
	WD	96.1%(365/380)	96.8%(368/380)	96.8%(368/380)	97.1%(367/378)
D4	ND	72.7%(354/487)	86.9%(423/487)	90.8%(442/487)	94.6%(439/464)
	WD	87.1%(424/487)	93.0%(453/487)	95.1%(463/487)	95.4%(460/482)
D5	ND	80.8%(181/224)	89.7%(201/224)	93.3%(209/224)	96.8%(209/216)
	WD	93.3%(209/224)	95.5%(214/224)	97.3%(218/224)	98.2%(218/222)
D6	ND	62.4%(169/271)	77.9%(211/271)	82.3%(223/271)	89.6%(223/249)
	WD	62.4%(169/271)	77.9%(211/271)	82.3%(223/271)	89.6%(223/249)
D7	ND	73.7%(373/506)	86.2%(436/506)	89.9%(455/506)	93.2%(454/487)
	WD	87.6%(444/507)	90.9%(461/507)	93.3%(473/507)	95.0%(472/497)
N1	ND	57.1%(265/464)	66.2%(307/464)	73.9%(343/464)	88.6%(319/360)
	WD	57.3%(266/464)	66.2%(307/464)	73.9%(343/464)	88.6%(319/360)
N2	ND	69.1%(304/440)	78.4%(345/440)	84.1%(370/440)	93.7%(359/383)
	WD	76.1%(335/440)	83.9%(369/440)	87.5%(385/440)	92.6%(374/404)
N3	ND	67.0%(240/358)	78.5%(281/358)	83.8%(300/358)	91.0%(283/311)
	WD	77.7%(278/358)	83.5%(299/358)	85.5%(306/358)	88.3%(295/334)
N4	ND	70.2%(415/591)	78.7%(465/591)	83.6%(494/591)	91.6%(488/533)
	WD	78.8%(465/590)	86.1%(508/590)	87.0%(513/590)	90.5%(507/560)
Total	ND	69.8%(4592/6579)	79.9%(5259/6579)	84.4%(5552/6579)	91.1%(5463/5995)
	WD	79.3%(5219/6583)	86.0%(5659/6583)	88.4%(5822/6583)	92.3%(5746/6227)

ND: without using dictionary   WD: using dictionary
M: Majority Voting   D: Dictionary Lookup   R: Rejection

Table 5.3: Postprocessing results on words from large clusters($DDM = dd2$)

5.5 Error Analysis and Future Improvements

The potential improvement in performance possible with further refinements of the algorithm presented in this chapter are illustrated by an analysis of the errors made when page A1 with dd2 noise was processed. Out of the 380 words in large clusters, 24 were recognized incorrectly and 15 were rejected when the M+D+R postprocessing method was used. Figure 5.9 shows the contents of the clusters calculated from document A1 that were incorrectly recognized. The decision made by the OCR package, the output of the postprocessing algorithm, and the truth value for each word are given. The decision made by the postprocessing algorithm for each cluster is shown after the identifier for the cluster.

Cluster 16 shows that the first character "A" was correctly recognized in three out of the six words in the cluster and the last four characters "ica's" were correctly recognized in four out of six cases. If the locations of the character segmentation points were available in those word images, the portion of the words between the "A" and the "ica's" could be further inspected and compared to portions of words that were successfully recognized in other words to derive the decision that the missing characters are the ligature "fr". Such a divide-and-conquer strategy may also be applicable to clusters 45, 47, 51, and 80.

Another subset of the errors are attributable to proper nouns that are not in the dictionary. In clusters 41, 43, 59, and 66, the correct choice was among the decisions in the cluster. It just was not chosen by the postprocessing algorithm. This could be solved by including more proper nouns in the dictionary or perhaps detecting proper nouns a-priori [33] and recognizing them with a specialized form of divide-and-conquer. Cluster 52 would be best solved by the second form of this strategy.

The remainder of the errors illustrate the need for improving the rejection strategy or that some errors are probably non-recoverable. Cluster 38 contains three different legal dictionary words. In this case all three words should be rejected. Clusters 68 and 79 contain alternative decisions that are legal dictionary words. The best way to make these choices is probably to employ a different form of postprocessing such as word transition probabilities [63, 64] or parsing [62].

Thus, there is the potential, with further development of the algorithm proposed in this chapter, to correct all but four of the errors. This would yield a 99% correct rate with no errors if the divide-and-conquer strategy always made the correct decision.

5.6 Conclusions and Future Directions

A new OCR postprocessing method based on word image equivalence was proposed. The method combines information about which word images are equivalent with the recognition results calculated on those words by a commercial OCR package. The result is an algorithm that corrects errors made when different instances of the same word are assigned different identities by an OCR.

Experiments on 16 degraded document pages (with over 12,000 words in total) show that on a large portion of the word images (more than half the words in the pages), the method improved the word recognition accuracy from 70% to 92%. By analyzing the errors on one page, it was found that there is still room to further improve the performance of the proposed approach. With further refinements, the proposed algorithm could have an accuracy as high as 99% correct if the errors made on the selected page are representative of the errors in general.

The method works only for a portion of the words in a document page. But the approach can be extended to the more general situation. Word image equivalence is only one of several visual inter-word relations that we have observed [58]. The other types of relations concern partial similarity. For example, one word image can be a subpattern of another word image,

```
CLUSTER 16 ---  DECISION: cats
WImg.0608        OCR:   Afinica's     CORRECTION: cats      TRUTH: Africa's
WImg.0044                  Ahica's                  cats             Africa's
WImg.0172                  f*;ca's                  cats             Africa's
WImg.0723                   kica's                  cats             Africa's
WImg.0133                  A*ica's                  cats             Africa's
WImg.0038                     cats                  cats             Africa's
CLUSTER 36 ---  DECISION: tune
WImg.0042                     tune                  tune                time.
WImg.0549                     tine                  tune                 time
WImg.0738                   tirrxe                  tune                 time
CLUSTER 38 ---  DECISION: hats
WImg.0796                      has                  hats                  has
WImg.0801                      bus                  hats                  has
WImg.0626                     hats                  hats                  has
CLUSTER 41 ---  DECISION: Rejected
WImg.0714              Brazzaville              Rejected          Brazzaville
WImg.0325             13razzaville              Rejected          Brazzaville
WImg.0519               Brazzavdle              Rejected          Brazzaville
CLUSTER 43 ---  DECISION: Pi
WImg.0107                      Pik                    Pi                  Pik
WImg.0188                       Pi                    Pi                  Pik
WImg.0393                        ~                    Pi                  Pik
CLUSTER 45 ---  DECISION: Rejected
WImg.0219                   .Mica.              Rejected              Africa.
WImg.0410                  Airica,              Rejected              Africa,
WImg.0691                   Afros,              Rejected              Africa,
CLUSTER 47 ---  DECISION: notary
WImg.0598                   Sitars                notary             military
WImg.0045                   notary                notary             military
WImg.0065                 rniEtary                notary             military
CLUSTER 51 ---  DECISION: Rejected
WImg.0513                  Mmst.er              Rejected             Minister
WImg.0106                   Mmster              Rejected             Minister
WImg.0070                  Moister              Rejected             Minister
CLUSTER 52 ---  DECISION: Rejected
WImg.0514                    Boeta              Rejected                Botha
WImg.0242                   Boffia              Rejected                Botha
CLUSTER 59 ---  DECISION: Man
WImg.0128                      Man                   Man                  Van
WImg.0111                      Van                   Man                  Van
CLUSTER 66 ---  DECISION: Rejected
WImg.0129                 Heerden,              Rejected             Heerden,
WImg.0112                 HeerdenF              Rejected             Heerden,
CLUSTER 68 ---  DECISION: agam
WImg.0718                     agam                  agam                again
WImg.0683                     agam                  agam                again
CLUSTER 79 ---  DECISION: economy
WImg.0788                  economy               economy             economic
WImg.0825                 egonomic               economy             economic
CLUSTER 80 ---  DECISION: Rejected
WImg.0526                   Abican              Rejected              African
WImg.0656                  A*ican              Rejected              African
```

Figure 5.9: Examples of postprocessing errors from page A1 with DD2 and WD

or two word images can match very well on their left parts, and so on. The coverage of relations based on visual partial similarity is much larger than that of relations based on visual word equivalence. Consistency analysis of relations between visual and symbolic levels (see Figure 5.10) can be very helpful to verify those words that have been recognized correctly, to detect potential word recognition errors and correct them. In Chapter 6, the use of partial relations to further improve OCR performance is explored.

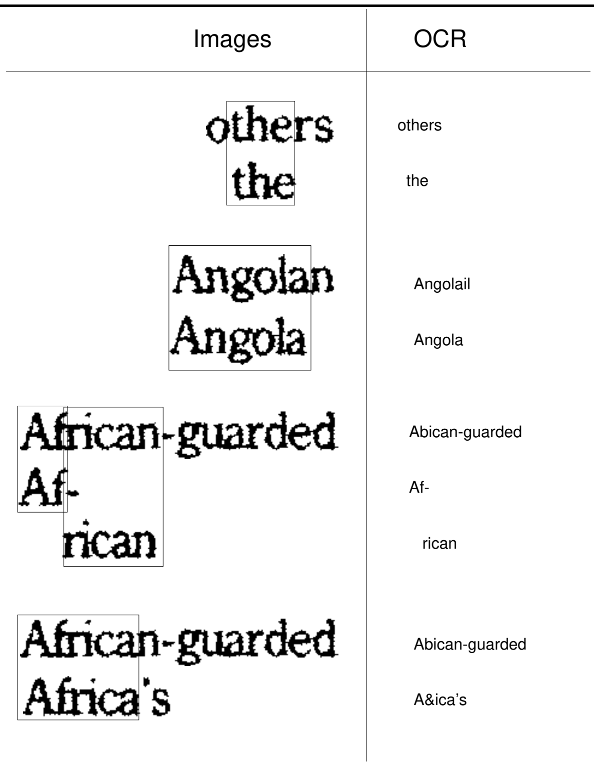


Figure 5.10: Consistency Analysis Using Partial Similarity

Chapter 6

OCR Postprocessing Using Visual Inter-Word Relations

In the previous chapter, we demonstrated that one type of visual inter-word constraint, word image equivalence, can be exploited to detect and correct OCR errors for a large portion of the word images in a document. This chapter continues the exploration of applying visual context to the postprocessing stage of text recognition. A number of other visual relations between words in a document are also useful. The basic idea of detecting equivalent word images is extended to detecting equivalences between portions of words. For example, the first three characters in *theatre* are the same as the first three characters in *thesis*. This information is then used to constrain the decisions of a recognizer so that they are the same in both cases. Also, the network of such relations in a document are used together with the results from a conventional OCR system to detect typographic characteristics of the document and to build an image representation for each of the characters it contains. This is then used in a self-teaching OCR system to recognize the rest of the text in the document.

The approach proposed here is an extension of the character-based clustering and deciphering algorithms used previously [16, 18, 108]. The concept of a self-teaching OCR system has also been used in a character classifier that automatically adapts itself to a single font [9]. The underlying assumption that a given document is printed primarily in a small number of fonts is also utilized in the algorithm proposed here.

The rest of this chapter discusses the proposed algorithm. The procedure for computing visual inter-word relations is discussed. Experimental results that operate directly on the output of a commercial OCR system are presented. The bounding boxes of words provided by the OCR system are used in the calculation of the visual inter-word relations and the character recognition results are used to derive the font representation. The word-level error rate of the OCR system is reduced by 53% on 71% of the word images in the test documents. Future extensions of the approach are discussed that will improve its accuracy.

6.1 Overall Design of an OCR Postprocessing System

The problem of postprocessing can be described as follows: given an image of a text page and its OCR output, verify a word recognition result if the corresponding OCR output is consistent with the appropriate visual inter-word relations. Detect and correct errors if an inconsistency is found between the visual and symbolic information, and learn typographic parameters (font, spacing, etc.) from the verified words and use them together with visual inter-word relations to re-recognize the words that are suspected to be incorrect.

Figure 6.1 shows the overall design of the OCR postprocessing system. There are five modules in the system. Given a text page, an OCR system can be applied to locate a sequence of word images. For each word image, its bounding box and recognition result are generated. There are three modules for visual and symbolic context analysis which provide contextual constraints to be used in the postprocessing model later. Symbolic contextual analysis can be conducted based on the word recognition results provided by the OCR. Word images can be clustered by image matching. In the visual contextual analysis module, the prototypes of the image clusters will be further compared to calculate partial inter-word relations.

Given word images and their OCR results, word image clustering can be accelerated significantly by using the symbolic information from the OCR output, even though the output may contain recognition errors. An OCR-result-guided image clustering algorithm is proposed. The outline of the algorithm is:

1. cluster word images by string matching based on their word recognition results;
2. split clusters by image matching;
3. merge clusters by image matching directed by string distance editing.

The intuition behind the method is minimizing the number of image matching operations. Similarly, other visual inter-word relations can be computed more efficiently by taking advantage of OCR results.

In the postprocessing module, visual and symbolic contextual information is exploited to detect and correct OCR word recognition errors. There are four steps proposed here: voting, font learning, verification and re-recognition. The detailed design of these steps will be discussed in the next section.

6.2 OCR Postprocessing with Visual Inter-Word Relations

A four-step algorithm is proposed in this section that postprocesses OCR results using visual inter-word relations. The objective is to locate word decisions that have high confidence, extract font and typesetting information, verify OCR decisions which seem to be correct, and re-recognize those word images which are suspected to be recognized incorrectly by the OCR. In the course of locating such high confidence decisions, some OCR errors are corrected. These high confidence word decisions are then used to learn images that correspond to individual characters and character sequences. These images are then used to decompose

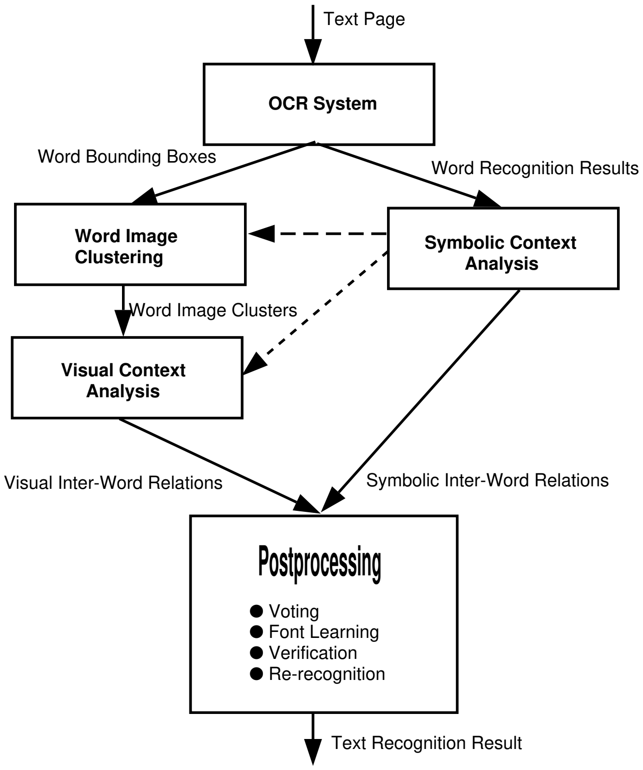


Figure 6.1: Overall design of an OCR postprocessing system

the remaining word images and generate new recognition results for them. Details of the four steps are presented below.

In the first step, a *voting* procedure is used on the whole-word clusters. The word decisions from clusters that contain two or more words are inspected and if a majority of them agree, that decision is output for the words in that cluster.

Figure 6.2 shows an example of a cluster of six instances of the word "`the`." Five of them were recognized correctly and one of them was incorrect (the word *die*). This error is corrected by voting. Experience has shown that voting produces very reliable performance for about half the words in a document.

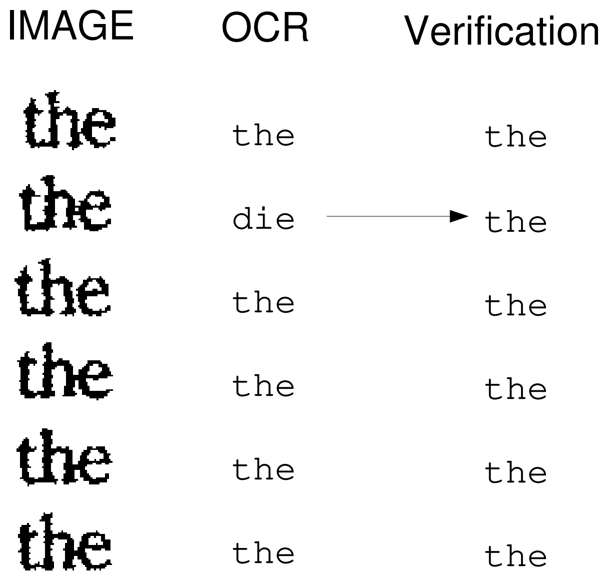


Figure 6.2: Example of the voting procedure.

In the second step, a *font learning* method is performed in which the visual interword relations are used to decompose the prototypes for the clusters that voting produced decisions for. This results in image prototypes for many individual characters.

An example of how the visual interword relations are used to decompose a word image is shown in Figure 6.3.

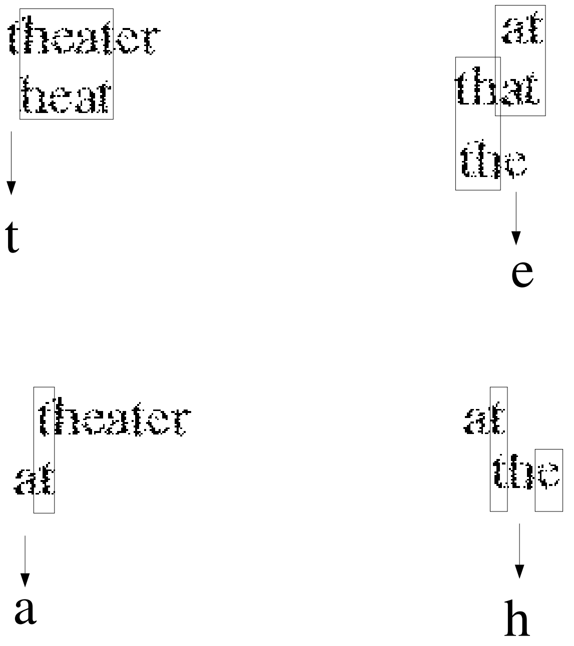


Figure 6.3: Example of font learning.

In the third step, a *verification* algorithm is executed on the word images that voting was unable to make a decision for. Visual inter-word relations are calculated between each image and the prototypes for the clusters output by voting. A word image is "verified" if its decomposition into sub-patterns is mapped onto ASCII decisions that agree with the original OCR result. An OCR error can also be corrected in this step if there are high confidence visual inter-word relations between the input image and portions of the cluster prototypes found during voting. The verification step processes each word in a cluster sequentially and generates a list of alternatives for all the words in the cluster. This is done by appending the verified results for each word.

An example of verification is shown in Figure 6.4. The original OCR decision for the word was *verification*. The image for this word was decomposed into three parts: *verif*, *i*, and *cation*. The *verif* was the left-part of the word *verify* that was recognized elsewhere in the document. The single character *i* was verified by the results of font learning. The right-part *cation* was verified by matching to the right part of the word *vocation* that had been verified before. Error correction is illustrated by the word *African* that was mis-recognized as *A*ican* by the OCR. The reliable match of the left part of this image to the word *Africa* allows for this error to be corrected. The generation of alternatives for words in the same cluster is also illustrated by the two different OCR decisions for the word *first*.

In the fourth step, a *re-recognition* procedure is executed on all the remaining word images. Every such image is decomposed into sub-parts using visual relations calculated from the images output by voting, font learning and verification. This produces a lattice of possible overlapping sub-images along with their OCR results(see Figure 6.5). Then all the paths through this lattice are computed and each complete path is provided with a score that measures the degree to which each sub-image in the path matches the original word image. All the complete paths that also occur in a dictionary are placed in the candidate list for the word and the complete path with the best cost is output. Appropriate thresholds are incorporated in the algorithm so that character strings not in the dictionary may also be output. This approach is similar to some methods used in cursive script recognition [150]. The primary difference is that the algorithm proposed here learns the character image information it uses from the input page rather than from a previous training step.

An example of re-recognition is shown in Figure 6.6. Overall there are five complete paths that cover the entire word image. These paths along with their strings of ASCII decisions and cumulative matching scores are shown on the right side of Figure 6.6. Each of those paths is looked up in a dictionary and the complete path with the minimum cost (best match) is output. In the example shown, this results in the word *strain* being output. The difference in confidence between the three paths for *strain* show the effect that finding correspondence between longer substrings can have on performance. The best choice had a confidence of 0.90 because it contained a five-character-image string that was matched to

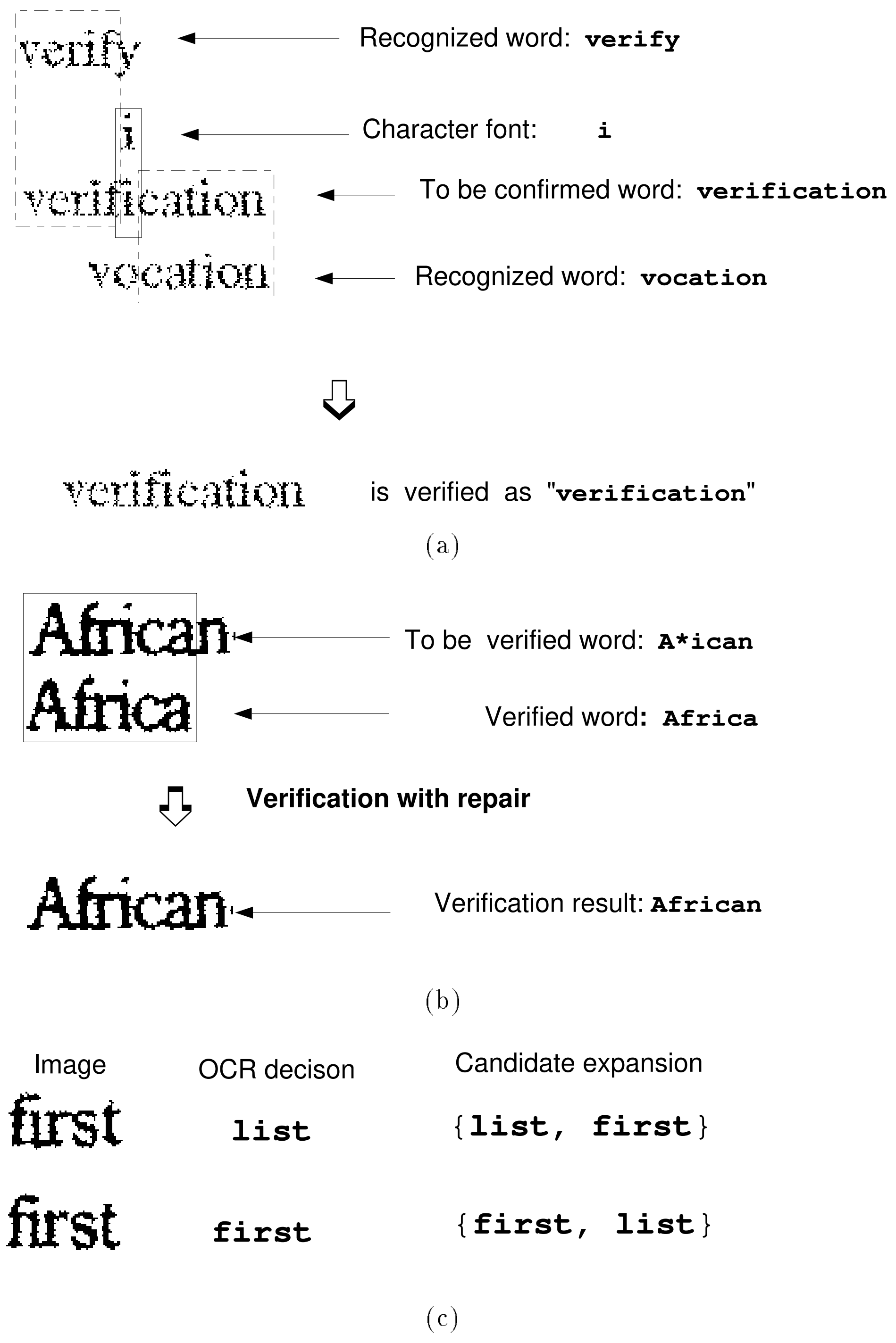


Figure 6.4: Cases of verification: (a) confirmation of OCR decision, (b) error correction, and (c) alternative generation.

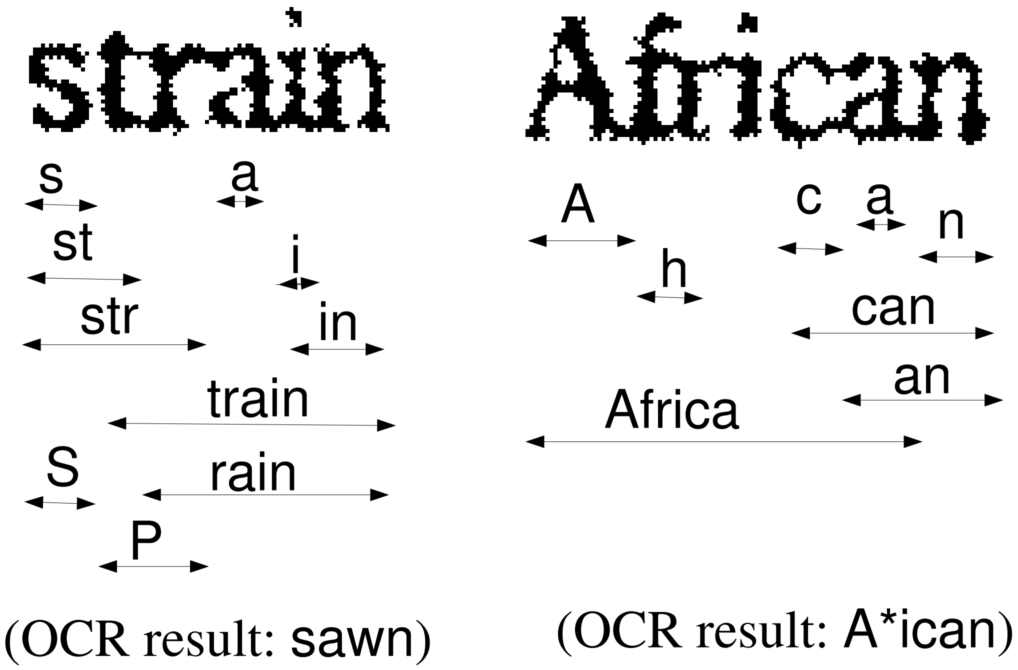


Figure 6.5: Keep partial matching results in a lattice

the word *train*. There was only one more single character (s) that needed to be matched to complete the path.

6.3 An Experimental System and Its Preliminary Results

To utilize visual inter-word relations and the postprocessing methods described above, an experimental system was implemented. It was designed as a stand-alone postprocessing system. The input required for the system is the OCR result for each word, including the OCR decision and a possible list of alternative decisions, coordinates of each word bounding-box, and the text image. Currently the system uses *Caere* 's *AnyFont* package to generate OCR results and word bounding-box information. The output of the system is its decision and alternative list for each word image. The major steps of the system are described as below:

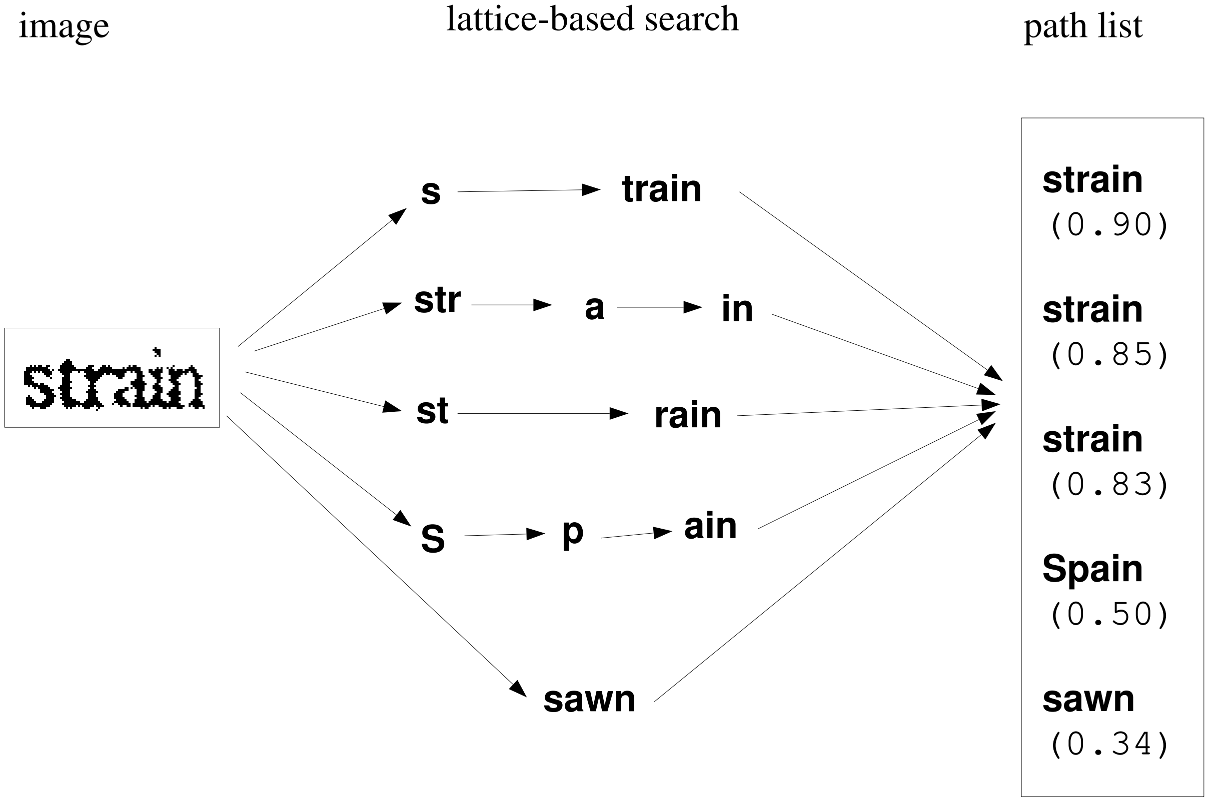


Figure 6.6: Example of re-recognition.

1. Run *Caere*'s *AnyFont* package to generate a text recognition result. For each word image, there is an OCR decision and a possible candidate list.

2. Cluster word images. To improve run time, OCR results are used to guide the process of image clustering.

3. Conduct visual similarity analysis. For each pair of image clusters, the system tests whether they hold any visual inter-word relations, using the algorithm described in Chapter 4.

4. Apply a consensus voting method to determine the identity of each large cluster. If the majority of images in a cluster have the same OCR decision, the decision will be assigned as the decision for the cluster and all images in the cluster will inherit the cluster decision as their decisions. All those word images will collected as a set, which is denoted as S_1. Words in S_1 are labelled as *verified* words. The set for the remaining word images is denoted as S_3. For a word in S_3 which belongs to a multiple-entry cluster but the majority voting method fails to make a decision for the cluster, all candidates for the cluster will be assigned as the candidate list for the word.

5. Learn font and typesetting information using visual inter-word relations based on the image set S_1. After this step, the word images in the set will be segmented into many parts, most of them are character images. Those image parts will be treated as examples of font and typesetting in the page.

6. Verify OCR decisions for the remaining word images in S_3. Assuming that the OCR decision for an image is correct, if its relations with other words are consistent at the image and symbolic level, its OCR result is verified. The word will be put into the set S_2.

7. Re-recognize those words, which are still in S_3, through lattice-based analysis. For a word image, its visual inter-word relations will be recorded in a lattice; the regions matched with font images will also be registered in the lattice. By dictionary-based search, all possible candidates can be generated from the lattice. If there are one or more candidates, the candidate with the highest confidence will be assigned as the decision of the word. Other candidates, plus the original OCR decision and candidates from the cluster which the image belong to, will be assigned as the candidate list. The word will be put into S_2.

6.3.1 Experimental Results

The experimental system has been tested. The input to this system is the output from a commercial OCR (i.e., Caere's AnyFont package) as well as the page images that were provided to the OCR. The commercial device provides at least a single decision for each word and in cases where it is unsure, several alternatives are produced. Also, the bounding box coordinates for each word are output.

Six page images (listed in Table 6.1) were used to test the system. These were scanned at 300 ppi and the binary image produced by the scanning hardware was used. Uniform noise was added to each image using the documentation degradation model (DDM) package from the University of Washington [81]. The parameter set for DDM was $(820, 0.0, 1.0, 1.0, 1.0, 1.0, 3)$.

The accuracy of Caere's AnyFont OCR package on the original pages is very high, more than 98% correct at the word level. After adding uniform noise with DDM, the word correct rate dropped to 73.5%. It was observed that the word alternatives produced by the OCR do not improve performance significantly (see Table 6.2).

Word clustering was then computed using the bounding boxes output by the OCR and inter-word relations were calculated between pairs of clusters. In the present implementation, only the first four visual relations in Figure 1 were used. The threshold for image matching, r_0, was set to 0.60.

Table 6.1 shows the result of visual inter-word relation analysis. On average, about half of the words are in large clusters (containing two or more word images). The number of visual inter-word relations is large and varies from page to page.

page id.	# of words	# of clusters	# of large clusters	# of words in large clusters	# of visual inter-word relations btw clusters (*type-2,3,4*)
P_1	827	520	96	403	4269
P_2	1129	690	139	578	7917
P_3	826	494	90	422	17406
P_4	535	389	45	191	8784
P_5	686	467	78	297	14358
P_6	1019	607	113	525	25745

Table 6.1: Results of visual inter-word relation analysis.

After applying the proposed postprocessing system, the word images are divided into three sets: *voting*, *verification* and *rerecognition*. The system generates one decision for each word in the *voting* set and there are no other candidates for each word. The accuracy of the words in the *voting* set was improved from 92.2% to 98.0%. The accuracy of the words in the *verification* set was improved from 83.1% to 88.5% and the correct rate of the word alternatives was improved from 83.3% to 93.3%.

The correct rate of the words in the combination of the *voting* and *verification* sets was improved from 85.6% to 92.3% and the accuracy of their alternative lists was improved from 86.9% to 95.2%. It is important to note that the images in these sets account for about 71% of the words in the original text pages.

The complete re-recognition step is still under development. At the present time, only the generation of alternatives by tracing paths through the lattice has been implemented. This increased the accuracy of the candidate lists from 44.1% to 52.6%.

		OCR			Postprocessing		
word set	# of words	decision corr. rate	corr. rate of candidate list	avg. # of candidate	decision corr. rate	corr. rate of candidate list	avg. # of candidates
voting	1403	1293 92.2%	1298 92.5%	1.6	1375 98.0%	1375 98.0%	1.0
verification	2160	1752 81.1%	1799 83.3%	2.6	1912 88.5%	2016 93.3%	3.2
rerecognition	1459	644 44.1%	675 46.3%	3.1	644 44.1%	768% 52.6%	2.4
voting+ verification	3563	3050 85.6%	3097 86.9%	2.5	3287 92.3%	3391% 95.2%	2.4
voting+ verif. + rerecog.	5022	3694 73.5%	3772 75.1%	2.6	3931 78.3%	4159 82.8%	2.3

Table 6.2: Results of postprocessing.

6.4 Conclusions

In this chapter an approach was proposed that used visual relations between word images to improve the performance of an OCR system. This information has only been used to a limited extent in previous OCR techniques.

The proposed algorithm first calculates clusters of equivalent word images and then determines which sub-parts of the prototypes for the clusters are equivalent. This information is then used in a four-step method for postprocessing the OCR results. Experimental results were presented that showed the first three steps of the algorithm effectively reduced the word level error rate by 53% (from 14.4% to 7.7%) on six page images that contained over 5000 word images. This test data had been degraded by a uniform noise model.

Current work includes further development of the fourth step in the postprocessing algorithm (i.e., *rerecognition*). This part of the algorithm uses the visual inter-word relations to produce new recognition results that may be different from those output by the OCR. This algorithm uses images for characters and portions of words that have been recognized with high confidence to produce new recognition results.

Chapter 7

A Relaxation Algorithm Using Word Collocation

7.1 Introduction

Word collocation data is one source of information that has been investigated in computational linguistics and that has been proposed as a useful tool to post-process word recognition results [26, 130]. Word collocation refers to the likelihood that two words co-occur within a fixed distance of one another. For example, it is highly likely that if the word "*boat*" occurs, the word "*river*" will also occur somewhere in ten words on either side of "*boat.*" Thus, if "*river*" had been misrecognized with the neighborhood "*rover, river, ripper*" (i.e., rover is the top choice, river the second choice, etc.), the presence of "*boat*" nearby would allow for the recognition error to be corrected.

Previous work in using word collocation data to post-process word recognition results has shown the usefulness of this data [129]. This technique used local collocation data about words that co-occur next to each other to improve recognition performance. A disadvantage of this approach was that it did not allow for successful results on one word to influence the results on another word.

This chapter proposes a relaxation-based algorithm that propagates the results of word collocation post-processing within sentences. The promotion of correct choices at various locations within a sentence influences the promotion of word decisions elsewhere. This effectively improves on a strictly local analysis by allowing for strong collocations to reinforce weak (but related) collocations. Relaxation has been widely used to solve problems where the search for a globally optimal solution can be broken down into a series of local problems, each of which contributes to the overall result [131].

The rest of the chapter discusses the algorithm in more detail. An experimental analysis is discussed in which the algorithm is applied to improving text recognition results that are less than 60% correct. The correct rate is effectively improved to 83% or better in all cases. For about 40% of the words inside each test article, the accuracy of the decision made by the method can be as high as 99.5%.

7.2 Algorithm Description

Word collocation $WC(x, y)$ measures the likelihood that two words, x and y, co-occur within a given distance of each other. A numerical measure of the collocation strength of word pair (x, y) is given by the mutual information of (x, y):

$$WC(x, y) = MI(x, y) = log_2 \frac{P(x, y)}{P(x)P(y)}$$

where $P(x)$ and $P(y)$ are the probabilities of observing words x and y in a corpus of text and $P(x, y)$ is the joint probability of observing x and y within a fixed distance of one another. The strength of word collocation can also be measured by

$$WC(x, y) = F(x, y)$$

where $F(x, y)$ is the frequency of the word pair (x, y) in a corpus of fixed size. In this research, the distance between words x and y is set to one ($windowsize = 1$).

$MI(x, y)$ is the pure collocation measure which is independent of the word frequency of x and y. Although MI is popular in computational linguistics, this research uses *word-pair-frequency* as the collocation measure because the word frequency factor is useful for word candidate selection.

The relaxation algorithm that incorporates word collocation data receives the neighborhoods for each word in a sentence as input. Those neighborhoods are initially ranked by the confidence values provided by a word recognition algorithm. Each word in a neighborhood w_{ij} is repeatedly re-ranked for k iterations by calculating its new word collocation score:

$$p^{(k+1)}_{(w_{ij})} = \frac{p^{(k)}_{(w_{ij})} + r_{(w_{ij}, w_{i+1,1})} p^{(k)}_{(w_{i+1,1})} + r_{(w_{i-1,1}, w_{i,j})} p^{(k)}_{(w_{i-1,1})}}{\sum_{k=1}^{n} [p^{(k)}_{(w_{ik})} + r_{(w_{ik}, w_{i+1,1})} p^{(k)}_{(w_{i+1,1})} + r_{(w_{i-1,1}, w_{i,k})} p^{k}_{(w_{i-1,1})}]}$$

where $p^{(k+1)}_{(w_{ij})}$ is the probabilistic score of the word candidate w_{ij} at time $k + 1$. The initial score $p^{(0)}_{wij}$ is the confidence (between 0 and 1) provided by a word recognizer for the word candidate w_{ij}. The compatibility functions $r_{(w_{i-1,1}, w_{ij})}$ and $r_{(w_{ij}, w_{i+1,1})}$ are defined as

$$r_{(w_{i-1,1}, w_{ij})} = \frac{WC(w_{i-1,1}, w_{ij})}{\sum_{k=1}^{n} \mid WC(w_{i-1,1}, w_{ik}) \mid}$$

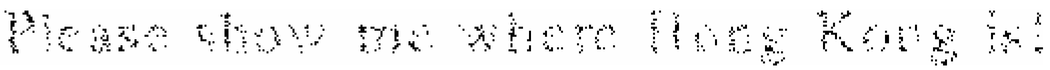
and

$$r_{(w_{ij}, w_{i+1,1})} = \frac{WC(w_{ij}, w_{i+1,1})}{\sum_{k=1}^{n} \mid WC(w_{ik}, w_{i+1,1}) \mid}$$

where $WC(x, y)$ is the *word collocation* of the word pair (x, y). This method of calculating the score for w_{ij} at time $k + 1$ is an improvement over a previous approach that did not incorporate recognition confidence [63].

This measure uses the top-ranked choice of adjacent words to adjust the ranking in each neighborhood. Repeated applications of this measure effectively propagates results across a sentence. In fact, it may require several iterations for the algorithm to converge on a stable state from which further iterations cause no significant changes in the rankings.

Figure. 7.1 shows an example, using actual data, of how the algorithm operates. A degraded image of the sentence "Please show me where Hong Kong is!" is input to a word recognizer. The top three recognition choices are shown in each position. Only two of the seven words are correct in the first choice. After one iteration, six of the seven are correct and after two iterations all seven word decisions are correct.



Initial Word Neighborhoods								
position	1	2	3	4	5	6	7	8
top1	Places	snow	me	whale	Kong	Kong	it	!
top2	Please	slow	we	where	Hong	Hong	is	
top3	Pieces	show	mo	chore	Hung	Kung	Is	
Word Neighborhoods After Iteration 1								
position	1	2	3	4	5	6	7	8
top1	Places	show	me	where	Hong	Kong	is	!
top2	Please	slow	we	whale	Kong	Hong	it	
top3	Pieces	snow	mo	chore	Hung	Kung	Is	
Word Neighborhoods After Iteration 2								
position	1	2	3	4	5	6	7	8
top1	Please	show	me	where	Hong	Kong	is	!
top2	Places	slow	we	whale	Kong	Hong	it	
top3	Pieces	snow	mo	chore	Hung	Kung	Is	

Figure 7.1: An example of the relaxation process (the sentence to be recognized is "*Please show me where Hong Kong is !* "

7.3 Experiments and Analysis

Experiments were conducted to determine the performance of the proposed algorithm. The data used in the experiments were generated from the Brown Corpus and Penn Treebank databases. These corpora together contain over four million words of running text. The Brown corpus is divided into 500 samples of approximately 2000 words each [89]. The part of the Penn Treebank used here is the collection of articles from the *Wall Street Journal* that contain about three million words.

Five articles were randomly selected from the Brown Corpus as *test samples*. They are *A06*, *G02*, *J42*, *N01* and *R07*. *A06* is a collection of six short articles from the *Newark Evening News*. *G02* is from an article "*Toward a Concept of National Responsibility* " that appeared in *The Yale Review*. *J42* is from the book "*The Political Foundation of International Law*." *N01* is a chapter from the adventure fiction "*The Killer Marshal*." *R07* is from the humor article "*Take It Off*" in *The Arizona Quarterly*. Each text has about 2000 words. There are a total of 10,280 words in the testing samples.

The *training data* from which the collocation statistics were calculated was composed of the approximately 1.2 million distinct word pairs in the combination of the Brown corpus and Penn Treebank minus the five test samples. Examples of word pairs and their frequencies are shown below.

the	doctor	64
a	doctor	27
doctor	and	8
doctor	was	8
doctor	who	7
his	doctor	6
doctor	bills	4
ward	doctor	1

A qualitative estimate for the upper bound, or best performance that could be expected with a technique that uses word collocation data, was derived from the test articles. This was done by calculating the percentage of words in the test articles that also appear in the training data. This is relevant since if a word does not occur in the training data there will be no collocations stored for it and the algorithm may not select it. The results in Table 7.1 show that somewhere between 97% and 98% of the isolated words in each of the test articles occur in the training data. Thus, only 2% to 3% of the words in a typical document are not found.

test article	no. of words	no. in training data
A06	2213	2137 (97%)
G02	2267	2201 (97%)
J42	2269	2208 (97%)
N01	2313	2271 (98%)
R07	2340	2262 (97%)

Table 7.1: Isolated word occurrence in the training data

The potential upper bound in performance is also illustrated by the results shown in Table 7.2. These give the number of times each word in the test articles appear adjacent to the same word or words in the training data. Statistics are provided that show how often words in the test articles are collocated with both the word before and the word after somewhere in the training data. Also, the number of times a word is collocated only with either the word before or the word after is given. These results show that about 60% of the words occur adjacent to the same words in both the test data and the training data. About 32% of the words are adjacent only to either the word before or the word after and 4% to 8% of the words have no collocations in the training database. Thus, an algorithm that uses collocation to choose the candidate for a word could achieve a correct rate of between 92% and 96%. However, this would require perfect knowledge about which collocation is correct. Actual performance will be less than this because of other words in the neighborhood that have collocations in the training data and the interactions of those words and their recognition confidence values during iterations of the relaxation algorithm.

test article	no. of words	both before and after	only before or after	neither before nor after
A06	2213	1342 61%	694 31%	177 8%
G02	2267	1355 60%	750 33%	162 7%
J42	2269	1349 59%	762 34%	158 7%
N01	2313	1609 70%	614 27%	90 4%
R07	2340	1499 64%	659 28%	182 8%

Table 7.2: Word collocation occurrence in the training data

7.3.1 Neighborhood Generation

Neighborhoods were generated for each of the 70,000 unique words in the combined corpus of testing and training data using the following procedure. Digital images of the words were generated from their ASCII equivalents first by converting them to an 11 pt. Times Roman font in postscript with the Unix command *ditroff*. The postscript files were then converted into raster images with the *ghostscript* system.

Neighborhoods were generated for each word by first calculating a feature vector for the word known as the stroke direction feature vector [55]. The neighborhoods for each dictionary word were then calculated by computing the Euclidean distance between its feature vector and the feature vectors of all the other dictionary words and sorting the result. The ten words with the smallest distance values were stored with each dictionary word as its neighborhood.

To mimic the performance of a word recognition technique in the presence of noise, the neighborhoods were corrupted. An assumed correct rate in each position in the neighborhood was given. For example, the top choice might be 80% correct, the second choice 10% correct, and so on. The noise model was applied to the text by calling a uniform random number generator for each word in the passage and scaling the result between zero and one. The correct rate distribution was then used to select the position in the neighborhood into which the correct word was moved. Thus, in the above example, 80% of the time the correct word would remain in the top position, 10% of the time it would be moved into the second position, and so on. The eight noise models shown in Table 7.3 were used to generate neighborhoods for the running text of the five test articles.

model	position in neighborhood									
	1	2	3	4	5	6	7	8	9	10
1	.55	.15	.08	.05	.04	.03	.03	.03	.02	.02
2	.65	.15	.06	.04	.02	.02	.01	.01	.02	.02
3	.70	.17	.05	.02	.02	.01	.01	.01	.005	.005
4	.75	.10	.05	.03	.03	.01	.01	.01	.005	.005
5	.80	.07	.05	.02	.02	.01	.01	.01	.005	.005
6	.85	.06	.03	.01	.01	.01	.01	.01	.005	.005
7	.90	.03	.02	.01	.01	.01	.005	.005	.005	.005
8	.95	.02	.01	.005	.005	.005	.002	.001	.001	.001

Table 7.3: Performance models used to generate corrupted word recognition results

7.3.2 Experimental Results

Before testing the algorithm, a baseline for performance comparison was determined. The accuracy rate in each position in the candidate list was calculated by re-ranking using word frequency, i.e., the a-priori probability of the word (see Table 7.4). The result shows that the correct rate of the first position in the neighborhood is around 75% by using word frequency data alone. It should be noted that these results were calculated from the entire training database, including the test articles. Since, as shown earlier, about 2% to 3% of the words in the test data do not occur in the training data, the performance obtained by re-ranking using frequency could be up to 2% to 3% lower.

	Article					Average
	A06	G02	J42	N01	R07	
top1	73.6%	74.7%	75.1%	76.6%	76.1%	75.2%
top2	89.0%	88.7%	91.8%	89.8%	90.1%	89.9%
top3	94.6%	94.6%	95.7%	95.5%	95.8%	95.2%
top4	96.9%	97.1%	97.8%	97.7%	97.5%	97.4%
top5	98.0%	98.0%	98.7%	98.8%	98.5%	98.4%
top6	98.4%	98.9%	99.0%	99.4%	99.2%	99.0%
top7	99.2%	99.3%	99.3%	99.6%	99.4%	99.4%
top8	100.0%	99.9%	99.9%	100.0%	100.0%	100.0%
top9	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
top10	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%

Table 7.4: Baseline for comparison (Re-ranking Top10 List by Word Frequency)

The result of applying the relaxation algorithm to the neighborhoods is shown in Table 7.5 and Figure 7.2. The average correct rate at the top choice across all the passages tested is given for ten iterations of relaxation. Table 7.6 and Figure 7.3 shows the results obtained when the test passages are included in the training data. The differences between the two results show the effect that "perfect" collocation data can have on performance.

The top choice correct rate when the training data does not include the test passages is between 83% and 86% correct. The best performance possible, as shown in Table 7.6, raises the correct rate at the top choice to between 91% and 93%. This is interesting when it is considered that the top choice correct rate of the input was initially as low as 56%. In most cases the majority of the improvement in performance is obtained after three iterations.

A drop in performance can be observed after the first iteration in cases where the initial correct rate is high. For example, when the first word has a 94% correct rate, at the first iteration it is improved to 95% and then it drops to 86% after ten iterations. This same effect does not occur when the initial correct rate is lower than 80%. Thus, in practice, the number of iterations should be controlled by an estimate of the overall error rate in word recognition. If low confidences have been assigned to many words in a text passage, then this should indicate that the relaxation should be iterated until the order of the decisions in the neighborhoods ceases to change. If many words have been assigned high confidences, then the relaxation should be terminated after one or two iterations.

Also, many of the cases that were correct on the first iteration but that were placed in a lower position in the neighborhood on a subsequent iteration were short words such as "one," that were confused with function words such as "the." The higher collocation strength of the function word caused it to be chosen incorrectly. This suggests that the relaxation algorithm should be used as part of a larger text recognition system and that this system should include a preprocessing step, such as [86], that detects function words and recognizes them separately. The relaxation algorithm would then be applied to the other words in the text. Since these will be the information-bearing words, the relaxation approach will be more likely to perform correctly.

Initial	Iter. 1	Iter. 2	Iter. 3	Iter. 4	Iter. 5	Iter. 6	Iter. 7	Iter. 8	Iter. 9	Iter. 10
56.4%	74.6%	81.2%	82.5%	82.9%	83.1%	83.2%	83.3%	83.3%	83.2%	83.2%
65.2%	80.3%	83.8%	84.2%	84.0%	84.1%	84.1%	84.1%	84.1%	84.1%	84.1%
70.9%	84.1%	85.6%	85.0%	84.9%	84.8%	84.7%	84.7%	84.8%	84.7%	84.7%
75.2%	85.8%	86.4%	85.4%	85.1%	85.0%	85.0%	84.9%	84.9%	84.9%	84.9%
79.9%	88.4%	87.4%	86.1%	85.5%	85.3%	85.2%	85.1%	85.1%	85.1%	85.1%
84.5%	90.7%	88.3%	86.7%	86.0%	85.9%	85.7%	85.6%	85.6%	85.6%	85.6%
89.7%	93.4%	89.2%	87.2%	86.4%	86.1%	86.0%	85.9%	86.0%	85.9%	85.9%
94.1%	95.4%	90.0%	87.6%	86.8%	86.5%	86.4%	86.3%	86.3%	86.2%	86.2%

Table 7.5: Correct rate at each iteration of relaxation

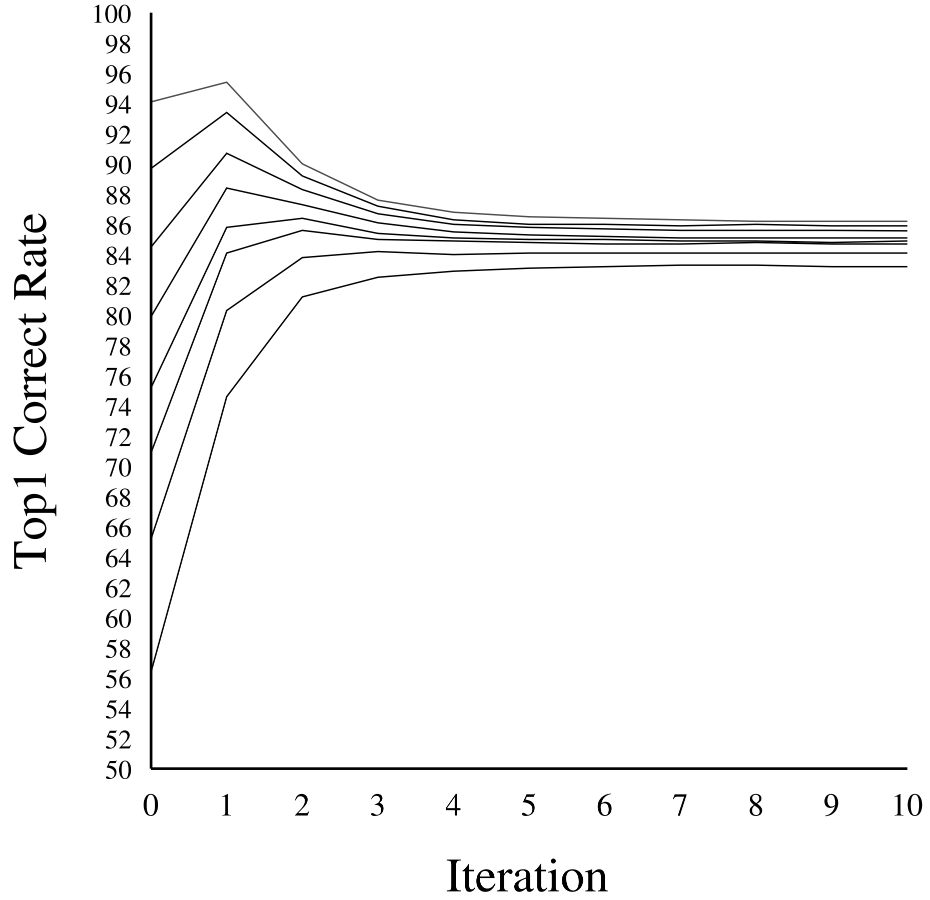


Figure 7.2: Correct rate at each iteration of relaxation

Initial	Iter. 1	Iter. 2	Iter. 3	Iter. 4	Iter. 5	Iter. 6	Iter. 7	Iter. 8	Iter. 9	Iter. 10
56.4%	80.7%	88.5%	90.2%	90.7%	91.0%	91.2%	91.2%	91.3%	91.3%	91.3%
65.2%	85.8%	90.9%	91.6%	91.8%	91.8%	91.8%	91.9%	91.9%	91.9%	91.9%
70.9%	88.9%	92.2%	92.1%	92.0%	92.0%	91.9%	91.9%	91.9%	91.9%	91.9%
75.2%	90.5%	93.0%	92.5%	92.3%	92.3%	92.1%	92.1%	92.1%	92.1%	92.1%
79.9%	92.7%	93.7%	92.9%	92.6%	92.5%	92.3%	92.2%	92.2%	92.2%	92.2%
84.5%	94.5%	94.4%	93.3%	92.8%	92.7%	92.5%	92.5%	92.5%	92.4%	92.4%
89.7%	96.4%	95.0%	93.7%	93.0%	92.8%	92.6%	92.5%	92.5%	92.5%	92.4%
94.1%	97.5%	95.5%	93.8%	93.1%	92.9%	92.7%	92.7%	92.7%	92.6%	92.7%

Table 7.6: Correct rate in relaxation when collocation includes test samples

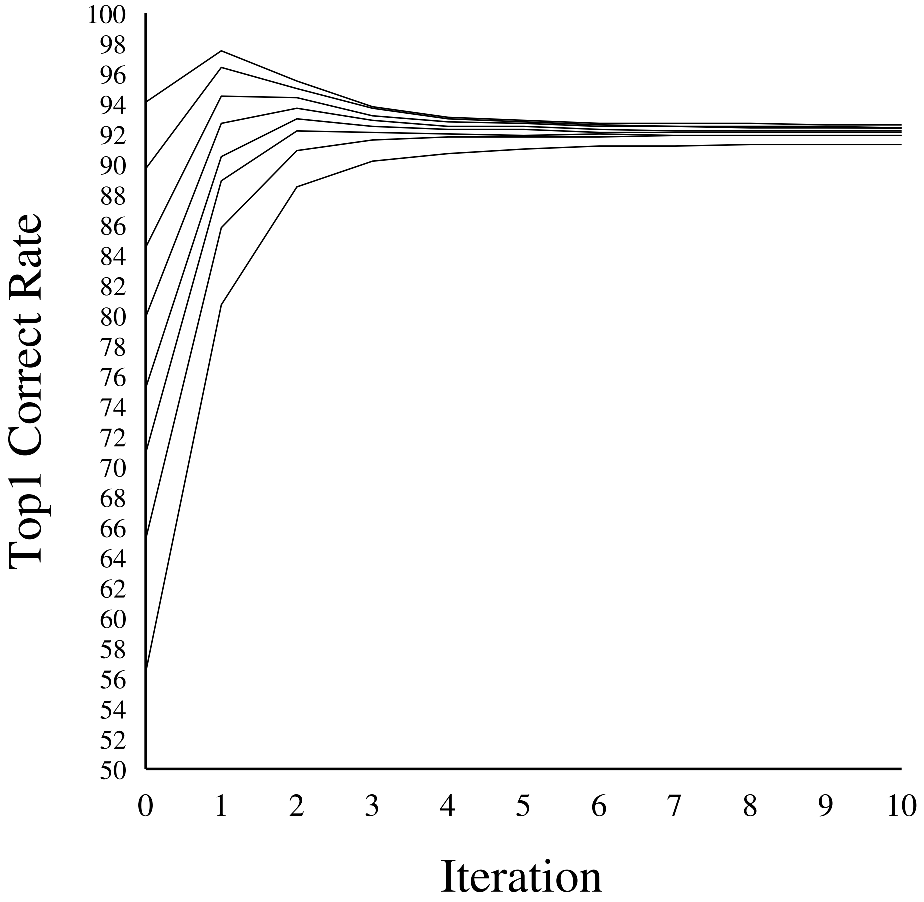


Figure 7.3: Correct rate when collocation includes test samples

The relaxation algorithm can improve the top1 accuracy of candidate lists, as shown above. Another way to show the improvement is to calculate the coverage rate of the top-n. Table 7.7 lists the correct rate of the top10 in the first ten iterations when the initial top1 correct rate is 56.3%. After ten iterations, the correct rate of the top1 becomes 83.2%; the correct rate of the top2 is 92.8%; and the top3 correct rate is 96.1%. This result indicates that if the original top10 list is reduced to a top3 list, the coverage of the reduced candidate list can be as high as 96.1%. In this way, the relaxation algorithm can serve as a tool to reduce the size of candidate lists generated by a word recognizer.

	Initial	Iter. 1	Iter. 2	Iter. 3	Iter. 4	Iter. 5	Iter. 10
top1	56.3%	74.5%	81.2%	82.4%	82.9%	83.0%	83.2%
top2	71.4%	88.5%	91.5%	92.7%	92.9%	92.8%	92.8%
top3	78.8%	93.4%	95.1%	96.0%	96.1%	96.0%	96.1%
top4	83.7%	95.4%	96.9%	97.6%	97.7%	97.7%	97.7%
top5	87.5%	96.7%	97.9%	98.5%	98.6%	98.5%	98.5%
top6	90.4%	97.6%	98.4%	98.9%	99.0%	99.0%	99.0%
top7	93.2%	98.3%	98.9%	99.2%	99.3%	99.3%	99.3%
top8	96.2%	99.1%	99.4%	99.6%	99.6%	99.6%	99.6%
top9	98.2%	99.5%	99.7%	99.8%	99.8%	99.8%	99.8%
top10	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%

Table 7.7: Top10 accuracy of relaxation when the initial top1 correct rate is 56.3%

After applying the relaxation algorithm, the confidence score of each candidate reflects its competitiveness among other candidates in the candidate list. A filtering procedure can be further applied to remove those candidates with low confidence scores in order to reduce the size of each list. Given a candidate list $\{w_{ij} || 1 \leq j \leq n\}$ for the word w_i, let $p(w_{ij})$ be the confidence score of the j-th candidate. The candidates inside a candidate list satisfy following constraints:

$$\sum_{j=1}^{n} p(w_{ij}) = 1.00$$

and

$$p(w_{ij}) \geq p(w_{i,j+1}) \qquad for \;\; 1 \leq j < n$$

A threshold, t, can be chosen to remove the last m candidates if

$$\sum_{k=n-m+1}^{n} p(w_{ik}) \leq t \qquad and \qquad p(w_{i,n-m}) + \sum_{k=n-m+1}^{n} p(w_{ik}) > t$$

By choosing t equal to 0.05, the size of neighborhood list can be reduced significantly(see Table 7.8). Although there are 8.7 candidates for each word image on average, after the filtering step, the average number of candidates becomes 3.6 while the accuracy of the reduced candidate lists drops slightly from 100.0% to 99.2%. Table 7.9 shows the distribution of neighborhood size after candidate reduction. One interesting result is that there are 42.0% of word images with a single candidate left and their correct rate is as high as 99.9%. This result indicates that word collocation constraints are very strong for many words and can achieve almost 100% accuracy in selecting the correct candidate if its confidence score is greater than 0.95.

Article	After Relaxation		After Candidate Reduction	
	Size of Neighborhood	Accuracy	Size of Neighborhood	accuracy
A06	8.9	100.0%	3.7	99.4%
G02	8.9	100.0%	3.7	98.9%
J42	8.9	100.0%	3.7	99.3%
N01	8.5	100.0%	3.1	99.5%
R07	8.3	100.0%	3.4	99.0%
Total	8.7	100.0%	3.6	99.2%

Table 7.8: Candidate reduction

7.4 Conclusions and Future Directions

In this chapter, a relaxation algorithm was described that used word collocation information to improve text recognition results. The experimental results showed that the correct rate

Neighborhood Size	Percentage		Accuracy
1	42.0%	(4783/11402)	99.9%
2	18.2%	(2070/11402)	98.7%
3	8.1%	(926/11402)	97.6%
4	4.5%	(511/11402)	96.7%
5	3.4%	(393/11402)	96.9%
6	1.8%	(203/11402)	98.0%
7	0.7%	(76/11402)	98.7%
8	1.3%	(149/11402)	100.0%
9	9.1%	(1043/11402)	99.8%
10	10.9%	(1248/11402)	100.0%

Table 7.9: Distribution of neighborhood size after candidate reduction

at the top choice of a word recognition algorithm could be improved from 56% to 83% correct. This is a substantial improvement compared to the best that could be obtained by re-ranking the neighborhoods using a-priori probability alone. The performance gap between the results achieved when the collocation data includes the test data and when it does not suggests that word collocation data should be collected from larger and more balanced English corpora. Analysis of the remaining errors showed that many of them could be corrected by using a larger window size and special strategies for processing proper nouns. Modifications of the ranking function should also be considered.

Another encouraging result is that the size of the neighborhood can be significantly reduced by removing candidates with very low confidence scores. As the experiment shows, about 40% of words have only a single candidate left and the correct rate of this candidate is almost 100%. The relaxation algorithm can thus be applied to locate a large portion of words inside the text for which the decision word can be chosen with a very low risk.

Chapter 8

Word Candidate Selection by Lattice Parsing

In Chapter 7, one typical statistical language model – word collocation, was exploited for word candidate selection. Word collocation trained from large text corpora reflects interactions between neighboring words. As a type of linguistic constraint, it is quite local in nature. Theoretically, many more global linguistic constraints can be useful for word candidate selection. In this chapter, we explore syntactic constraints at the sentence level. A lattice chart parser has been designed. The syntactic constraint and the word collocation constraint are integrated to facilitate the process of candidate selection.

8.1 Introduction

Given a degraded text page, a word recognition algorithm can generate one or more word candidates (or so-called "word hypotheses") for each word image. The OCR output can be described as a word lattice, in which word images are organized by their positions according to the reading order, and there are several competitive candidates at each position. Because such a lattice cannot be the final output of a text recognition system, further postprocessing methods have to be applied to select a decision candidate for each word image.

Assuming that we know in advance that the text to be recognized is written in normal English, linguistic constraints can be helpful for word candidate selection. One kind of statistical constraint, word collocation, was exploited in Chapter 7. In this chapter, we will investigate the use of another kind of structural constraint – language syntax.

Language syntax usually can be described by a sentence grammar. Given a word lattice, if sentence boundaries, such as periods, question marks and exclamation marks, have been identified correctly, the sequence of word images between two sentence-end marks is a valid English sentence. If we generate multiple candidates for each word image, there are many possible sequences of words for each sentence. Among them, many combinations are not legitimate sentences. The sequence of candidates which is most grammatical at the sentence level should be the most preferred interpretation for the sequence of word images, and therefore can be selected as the final decision for those word images.

The lattice in Figure 8.1 was generated by a word recognizer. The text is "`Children come to the garden to admire the pretty swan. This makes him very proud and happy. But he will always remember that he was once an ugly duckling.`" In the lattice, those candidates for the same word are visually very similar and all of them are valid words from a large dictionary.

Parsing is the process of analyzing the syntactic structure of a passage in a language, usually at the sentence level. A parser is a program that does parsing automatically. A parser includes three major components: a lexicon, a grammar and a parsing algorithm. The words allowed in the language are defined in the lexicon. The grammar is a set of rules or principles that describe the syntactic phenomena of the language. Given a sequence of words, a parser can determine whether it is a legitimate structure (i.e., a sentence) defined in the language according the grammar. If so, it can also generate the detailed structure of the input sequence (i.e., a parse tree).

position	1	2	3	4	5	6	7
top1	Children	came	to	lire	garden	to	admire
	1.00	0.25	0.50	0.25	0.50	0.50	0.50
top2		coma	tv	the	golden	tv	admits
		0.25	0.50	0.25	0.50	0.50	0.50
top3		come		tire			
		0.25		0.25			
top4		some		toe			
		0.25		0.25			

position	8	9	10	11	12	13	14
top1	lire	poetry	swam	.	This	brakes	him
	0.25	0.50	0.50	1.00	1.00	0.33	0.33
top2	the	pretty	swan			maker	hint
	0.25	0.50	0.50			0.33	0.33
top3	tire					makes	trim
	0.25					0.33	0.33
top4	toe						
	0.25						

position	15	16	17	18	19	20	21
top1	vary	pound	add	floppy	.	Brit	be
	0.50	0.50	0.20	0.50	1.00	0.25	0.33
top2	very	proud	and	happy		Bus	he
	0.50	0.50	0.20	0.50		0.25	0.33
top3			anti			But	lie
			0.20			0.25	0.33
top4			arid			Put	
			0.20			0.25	
top5			end				
			0.20				

position	22	23	24	25	26	27	28
top1	mill	always	remainder	brat	be	war	once
	0.33	1.00	0.50	0.25	0.33	0.33	0.33
top2	will		remember	chat	he	was	ones
	0.33		0.50	0.25	0.33	0.33	0.33
top3	win			mat	lie	woe	ooze
	0.33			0.25	0.33	0.33	0.33
top4				that			
				0.25			

position	29	30	31	32
top1	all	ugly	buckling	.
	0.33	1.00	0.25	1.00
top2	an		chalking	
	0.33		0.25	
top3	on		doubling	
	0.33		0.25	
top4			duckling	
			0.25	

Figure 8.1: Word lattice for three example sentences

For example, the parse tree for the sentence, "`Children come to the garden to admire the pretty swan`," can be found in Figure 8.2. The parse tree in the figure is constructed according to a POS-tag-based lexicon and a simple grammar. "NN" (noun), "VB" (verb), "DT" (determiner) and "JJ" (adjective) in the parse tree are POS tags. "S" (sentence), "S-BODY," "NP" (noun phrase), "VP" (verb phrase), "PP" (prepositional phrase), "TO-INF" (to-infinitive clause) and "S-END" are syntactic categories defined in the grammar. The grammar consists of a set of *context-free* rules such as

```
S --> S-BODY S-END
S-BODY --> NP VP
NP --> DT JJ NN
NP --> JJ NN
NP --> NN
VP --> VB
VP --> VB PP
PP --> IN NP
TO-INF --> TO VP
```

The syntactic analysis in this example is simplified so that many interesting syntactic phenomena, such as singularity of nouns, transitivity of verbs and tense of verb phrases are not represented. To conduct a detailed syntactic analysis, a more complex grammar has to be used.

To generate a correct parse tree for a given sentence, syntactic analysis sometimes is not sufficient because of some structural ambiguities, such as PP attachment and conjunction coordination, have to be resolved by using semantic and pragmatic information.

Syntactic analysis is one of the important steps in natural language understanding.

Parsing sentences from unrestricted English texts is still a challenging task. A large-scale English lexicon which supports the parsing of unrestricted English text is difficult to build. Parsing methods developed so far are still not accurate and efficient, especially when a sentence is long (more than 25 words on average) [56]. Parsing unrestricted English texts also requires a comprehensive English grammar which can cover most syntactic phenomena in unrestricted text. Such a comprehensive English grammar is not readily available.

```
( S
  ( S-BODY
    ( NP
      ( NNS  Children  ) )
    ( VP
      ( VP
        ( VB  come  )
        ( PP
          ( TO  to  )
            ( NP
              ( DT  the  )
              ( NN  garden  )  )  )  )
        ( TO-INF
          ( TO  to  )
          ( VP
            ( VB  admire  )
              ( NP
                ( DT  the  )
                ( JJ  pretty  )
                ( NN  swan  )  )  )  )  )  )
  ( S-END  .  )  )
```

Figure 8.2: The parse tree for the sample sentence, `Children come to the garden to admire the pretty swan.`

In English, there are usually several possible POS tags for a word. For example, the word "`like`" can be a verb or a preposition and the word "form" can be a noun or a verb. With multiple POS tags for each word in a sentence, only one tag will be assigned to each word so that the most preferred parse tree can be generated for the sentence based on those selected tags.

Theoretically, a traditional sentence parser can be generalized as a lattice parser which allows multiple candidates for each word position as input and selects a candidate for each word position after parsing. The goal of lattice parsing is to find the candidate sequence which has the best syntactic interpretation. Under the schema of lattice parsing, text recognition is interleaved with a certain level of text understanding. One advantage of applying the lattice parsing method to text recognition is that the recognition result can be directly used for further document understanding tasks, such as information retrieval, message understanding and full-text understanding.

If the average number of candidates at each position is large, lattice parsing is much more computationally expensive than sentence parsing. When there are multiple candidates at a word position, usually several candidates share the same POS tag. In this situation, syntactic constraints are helpful to reduce the size of the candidate set, but will have difficulty selecting one decision because the candidates in the reduced set all can take the same syntactic role for the position. For example, syntactic constraints are too loose to choose "`floppy`" or "`happy`" as the decision for word 18 in Figure 8.1. The inefficiency and insufficiency characteristics discussed above have to be considered as a serious limitation in applying lattice parsing to text recognition.

In this chapter, we propose a framework of lattice parsing for word candidate selection. Besides language syntax, a statistical language model is integrated to improve the efficiency and accuracy of lattice parsing. A *probabilistic lattice chart parser* is described that uses syntactic and word collocation constraints to find the best candidate for each word image. The English grammar used in the parser is a *Context-Free Grammar* with about 7,000 rules. Large English corpora play an important role in the parser. A large-scale English lexicon (with more than 70,000 different words) was derived from two tagged English corpora, the Brown Corpus and the Penn Treebank, and further sub-categorized by part-of-speech tags.

The statistical data, word collocation and word frequency, which were trained from the corpora, are used in several ways: 1) to reduce the number of word candidates generated by the OCR for each word image; 2) to direct lattice parsing and to reduce the search space so that the parsing process can work more efficiently; and 3) to perform syntactic structure disambiguation so that parsing will be more accurate. The work is aimed at exploring the methodology of unrestricted English text recognition. A broad-coverage grammar and large-scale lexicon are designed for this purpose. Because they are still in raw form, better performance can be achieved simply by improving the quality of the grammar or the lexicon.

8.2 Parser Overview

Generally, there are two types of linguistic constraints used in the parser. One is the global structural constraint which includes syntactic, semantic and pragmatic constraints to be described as a set of grammar rules. Another is local word collocation in which the identity of a word can be used to predict the identity of other nearby words. Local word collocation data can be collected by training on large text corpora [26]. The global structural constraints are exploited with a chart parsing model. The chart data structure provides a flexible framework for parsing [82, 166]. Statistical methods can be easily incorporated in a chart parser [100]. A chart parser can be extended to a lattice parser which allows for several word candidates at the same position, and therefore can be directly used for speech recognition [23, 152] and visual text recognition. The parser chooses the words on a path through the lattice that correspond to a legal sentence with the highest probability of being correct, given the sentences represented in the lattice.

In this approach (see Figure. 8.3), given a page of English text that has been segmented into sentences, a word recognition process generates a candidate list for each word image; next, a relaxation procedure reduces the top-n candidates for each image to the two best candidates by applying word collocation information; the top-2 lists for a sentence form a word lattice which is passed to the probabilistic lattice chart parser that builds all possible parse trees based on the reduced word lattice; finally, the word candidates that occur in the parse tree with the highest a posteriori probability as well as the parse tree itself are output.

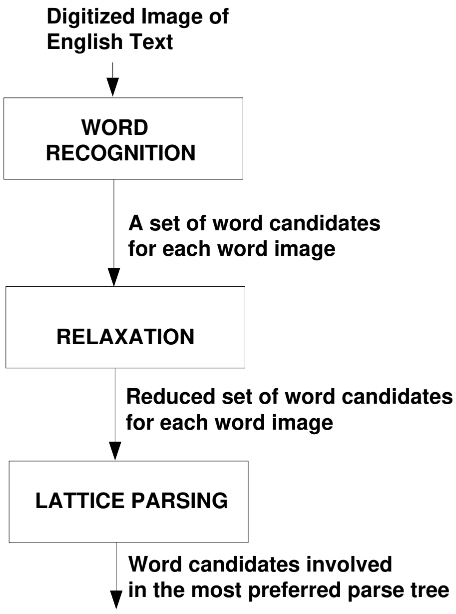


Figure 8.3: Overall Design

In the rest of this chapter, we will briefly present components of the system: the lexicon, the relaxation algorithm, the English grammar, and the parsing algorithm. The role of corpora in the system will also be discussed. In Section 7, the experimental results of the system are reported. Section 8 contains conclusions and future directions.

8.3 About the English lexicon

It is time-consuming to manually build a large-scale English lexicon that can support an unrestricted text understanding system. Automatic acquisition of lexical knowledge is a promising approach towards solving this problem [167]. Following this approach, we built an English lexicon by training on English text corpora that contain more than three million words. The corpora used are the entire Brown Corpus and part of the Penn Treebank. In the tagged corpora, each word is associated with its POS(part-of-speech) tag(such as NN, VB, IN, and so on). The tagsets used in the Brown Corpus and the Penn Treebank are similar. We adopted the tagsets of those two corpora with some modifications. We further subcategorized verbs into 15 subclasses by the type of argument the verb takes (see Table. 8.1). The information was extracted based on verb frames provided by *WordNet* [104].

Type	Arguments of Verb	Example	Note
0	NIL	The baby is *sleeping*.	intransitive verb
1	ADJP	She *looks* beautiful.	
2	CLAUSE	I *said* that I would go there.	
3	NP	He *killed* his enemy.	transitive verb
4	NP ADJP	He *found* the girl beautiful.	
5	NP NP	The girl *gave* the boy a book.	ditransitive verb
6	NP PP	I *put* the book on the table.	
7	NP TO-INF	I *remind* him to do it.	
8	NP CLAUSE	The boy *tells* the girl that he loves her.	
9	PP	She *worries* of her mother.	
10	TO-INF	She *started* to work.	
11	VPG	She *likes* swimming.	
12	NP VPB	He *helps* her do it.	
13	NP VPN/O	He *had* her hair cut.	
14	VPB	He *helped* make it.	

Table 8.1: Subclasses of Verb

For example, for the word "*tells*," the tag in the corpora is "*VBZ*." After subcategorization, in the system, the tags for " *tells*" are "*2VBZ*," "*3VBZ*," "*5VBZ*," "*6VBZ*," "*7VBZ*," and "*8VBZ*." Such subcategorization is useful in developing an English grammar.

Currently, the lexicon has more than 70,000 different words. On average, there are 1.57 POS tags per word. For each word, we have statistical data about: 1) its probability; 2) its potential POS tags and their probabilities; and 3) the words strongly collocated with it and their word collocation scores. For example, for the word "*fire*," the lexicon contains the information shown below:

```
Probability of the word:
```

```
                                0.00004435
 POS tags and probabilities:
                           0VB      0.00000228
                           3VB      0.00000228
                           7VB      0.00000228
                           9VB      0.00000228
                           NN       0.00004207
                           VB       0.00000228
 Word collocation pairs and their frequencies:
                           the/DT   fire/NN         54
                           a/DT     fire/NN         41
                           under/IN fire/NN         18
                           ......
                           to/TO    fire/VB         9
                           ......
                           on/IN    fire/NN         6
                           fire/NN  protection/NN   5
                           fire/NN  fighters/NNS    3
                           ......
```

The relaxation algorithm uses the word probability and word collocations to reduce the number of word candidates for each word image. The POS tags are used in parsing and the probabilities of the tags are used to derive the confidence score of each edge built in the chart during parsing. Tagged word collocation pairs help the parser adjust the probability of the tags for the word in context. For example, if the word "*fire*" follows the word "*to*" in a sentence, the parser raises the confidence value of the tag VB (verb), although generally the word "*fire*" is used as an NN (noun) more often than as a VB in English.

In a text to be parsed or recognized, there may be some unknown words which do not appear in the lexicon. The parser assigns several POS tags to each of them after applying some simple morphological analysis.

8.4 Preprocessing: Reduce Neighborhood Size by Relaxation

The word recognition system generates the top-n candidates for each word image except for frequent function words, such as "*a*," "*the*" and "*of*." For the function words, we assume they can be recognized correctly by a word image matching and clustering procedure [72]. Thus, the function words are "islands" or words with identities that the parser can rely on.

The next step is to apply a relaxation algorithm as a filter to reduce the top-n word candidates at each location to the top-m, where $m < n$. The detail of the method has been described in the previous chapter. The basic idea of the relaxation algorithm is to use local word collocation constraints to select the word candidates that have a high probability of occurring simultaneously with word candidates at other nearby locations. Figure 8.4 lists the candidates which are ranked by their word collocation strength through the relaxation process. The portion of word collocation data used in the relaxation is listed in Figure 8.5. Those candidates with low confidence score can be removed from the lattice without much risk. To further reduce the size of the candidate lists, only those candidates at the top-m positions will remain in the lattice.

As shown by many researchers, word collocation information is a useful statistical approach in NLP. If the text is in the domain well-trained on large corpus, after preprocessing the correct candidates should remain in the candidate pool. In the new top-m, the confidence score of a word candidate measures the strength of that candidate being collocated with its neighboring words.

Figure 8.4 shows the word lattice for three example sentences after relaxation.

Another example of applying the relaxation procedure is given by the sentence, "*In the past, the duties of the state, as Sir Henry Maine noted long ago, were only two in number: internal order and external security*"(see Figure. 8.6), the word recognizer generates the top five candidates for each word shown in Figure. 8.7. The correct choices are indicated by underlines.

After running the relaxation algorithm, the top two word candidates are shown in Fig-

position	1	2	3	4	5	6	7
top1	Children	came	to	the	garden	to	admire
	1.0000	0.3693	0.9998	0.9995	0.7572	0.9999	0.8334
top2		come	tv	toe	golden	tv	admits
		0.3683	0.0002	0.0004	0.2428	0.0001	0.1666
top3		some		tire			
		0.1373		0.0001			
top4		coma		lire			
		0.1251		0.0000			

position	8	9	10	11	12	13	14
top1	the	pretty	swam	.	This	makes	him
	0.9999	0.6666	0.5000	1.0000	1.0000	0.7776	0.9999
top2	lire	poetry	swan			maker	hint
	0.0000	0.3334	0.5000			0.2224	0.0001
top3	tire					brakes	trim
	0.0000					0.0000	0.0001
top4	toe						
	0.0000						

position	15	16	17	18	19	20	21
top1	very	proud	and	happy	.	But	he
	0.9998	0.6177	0.9999	0.9000	1.0000	0.9968	0.9999
top2	vary	pound	add	floppy		Put	be
	0.0002	0.3823	0.0000	0.1000		0.0030	0.0000
top3			anti			Bus	lie
			0.0000			0.0003	0.0000
top4			arid			Brit	
			0.0000			0.0000	
top5			end				
			0.0000				

position	22	23	24	25	26	27	28
top1	will	always	remember	that	he	was	once
	1.0000	1.0000	1.0000	1.0000	0.9971	0.9999	0.9999
top2	mill		remainder	chat	be	war	ones
	0.0000		0.0000	0.0000	0.0024	0.0000	0.0000
top3	win			brat	lie	woe	ooze
	0.0000			0.0000	0.0004	0.0000	0.0000
top4				mat			
				0.0000			

position	29	30	31	32
top1	an	ugly	duckling	.
	0.7273	1.0000	0.6250	1.0000
top2	on		chalking	
	0.1818		0.1250	
top3	all		buckling	
	0.0909		0.1250	
top4			doubling	
			0.1250	

Figure 8.4: Word lattice for three example sentences after relaxation(number of iterations = 10)

	word pair with frequency				word pair with frequency		
1	.	Bus	3	2	.	But	6098
3	.	Put	36	4	.	This	1868
5	But	he	310	6	This	maker	4
7	This	makes	5	8	admire	the	8
9	admits	the	4	10	always	remember	4
11	an	ugly	9	12	and	floppy	1
13	and	happy	4	14	be	war	1
15	be	was	1	16	came	to	240
17	come	to	239	18	garden	to	1
19	happy	.	16	20	he	was	1520
21	he	will	189	22	him	very	2
23	makes	him	8	24	once	all	2
25	once	an	5	26	once	on	4
27	ones	on	4	28	pound	and	13
29	proud	and	4	30	remember	that	24
31	some	to	12	32	that	be	5
33	that	he	1200	34	that	lie	1
35	the	garden	18	36	the	golden	17
37	the	poetry	2	38	the	pretty	6
39	to	admire	6	40	to	the	8675
41	to	tire	1	42	to	toe	6
43	ugly	duckling	4	44	very	proud	5
45	was	once	35	46	will	always	19

Figure 8.5: Related word collocation data used in the relaxation (data were collected from training corpora)

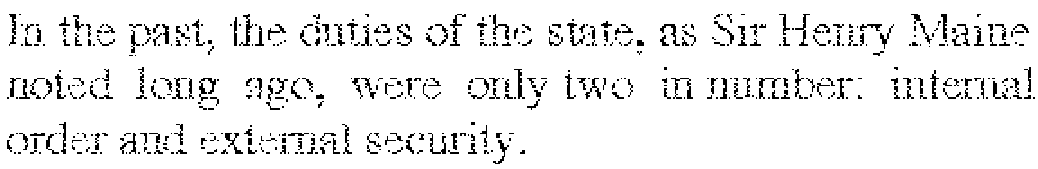

In the past, the duties of the state, as Sir Henry Maine noted long ago, were only two in number: internal order and external security.

Figure 8.6: Image for the Example Sentence

position	0	1	2	3	4	5	6	7	8	9
candidate1	Id	the	pass	,	the	danes	of	the	sane	,
candidate2	Ill		past			ditties			scare	
candidate3	In		prams			duties			score	
candidate4	hl		prose			robes			slate	
candidate5	in		purse			tubes			state	

position	10	11	12	13	14	15	16	17	18	19
candidate1	As	Sic	Harry	Mailer	hated	Dog	age	,	mere	duty
candidate2	arc	Sin	Hearty	Maine	noted	Log	ago		vere	holy
candidate3	are	Sir	Heavy	Monte	owed	long	ape		wear	only
candidate4	as	Sit	Henry	blame	tuned	loop	ego		were	tiny
candidate5	us	Six	Hurry	hume	voted	lung	organ		weve	why

position	20	21	22	23	24	25	26	27	28	29
candidate1	ruin	a1	metier	:	inertia	enter	add	eternal	Security	.
candidate2	too	hl	minter		interior	older	and	extend	scanty	
candidate3	torn	id	murder		internal	order	had	extended	scarcity	
candidate4	turn	ill	number		interred	outer	nod	extends	secretly	
candidate5	two	in	timber		interval	tatler	und	external	security	

Figure 8.7: Word recognition output(top 5) for the example sentence

position	0	1	2	3	4	5	6	7	8	9
candidate1		the	pass	,	the		of	the		,
candidate2			past							
candidate3	In					duties			score	
candidate4										
candidate5	in					tubes			state	

position	10	11	12	13	14	15	16	17	18	19
candidate1			Harry				age	,		
candidate2				Maine	noted		ago			
candidate3	are	Sir		Monte		long			wear	only
candidate4	as	Sit	Henry			loop			were	tiny
candidate5					voted					

position	20	21	22	23	24	25	26	27	28	29
candidate1				:						.
candidate2	too					older	and	extend		
candidate3			murder		internal	order	had		scarcity	
candidate4		ill	number							
candidate5	two	in			interval			external	security	

Figure 8.8: The reduced word lattice after applying the relaxation algorithm

ure. 8.8.

8.5 About the English Grammar

English syntax can be modeled by a context-free grammar which is composed of a set of production rules. For example, the following rules are typical.

```
S      <- S-BODY .
S-BODY <- NP VP
NP     <- NN
NP     <- NNS
NP     <- NP CC NP
VP     <- VB NP
VP     <- VB
```

This shows that a sentence(S) can be written as a sentence body(*S-body*) followed by a period. A sentence body is a noun phrase(NP) followed by a verb phrase(VP). A noun phrase is composed of a singular noun(NN), a plural noun(NNS), or two noun phrases with a coordinating conjunction (CC) between them. A verb phrase is composed of a verb(VB) followed by a noun phrase or just a verb.

Currently, there are 5714 rules in the context-free English grammar used for the experiments discussed here. The large rule base is mainly the result of verb sub-categorization. Although there are many rules, it was not difficult to write them manually because most rules can be derived formally from a small set of meta rules, such as those used in *Generalized Phrase Structure Grammar* (*GPSG*) models [127].

The original re-write rules are transformed into Chomsky normal form (CNF) by introducing new intermediate nonterminals. The number of intermediate nonterminals to be introduced can be minimized by an optimization process. After translation, the equivalent CNF grammar contains 7183 rules. Although the parser is based on the transformed grammar in CNF, the transformed grammar is hidden from the user because the intermediate nonterminals are not printed out. So from the user's point of view, parsing is based on the original CFG.

The grammar also includes semantic rules such as

```
PERSON <- Mr. NNP
PERSON <- Sir. NNP
COMPANY <- NNP Inc.
COMPANY <- NNP Company
```

Each of the rules in the grammar is associated with a confidence score which indicates the priority of the rule in comparison to other rules. A rule with a high score means that it is more likely to be used. Figure 8.9 shows a small portion of the large-scale rule-base.

Confidence	Rule	Confidence	Rule
1.00	S ← S-BODY S-END	1.00	S ← WH-S-BODY ?
1.00	S ← YN-S-BODY ?	0.10	S ← S-IM S-END
100.00	S-BODY ← Either S-BODY or S-BODY	100.00	S-BODY ← either S-BODY or S-BODY
10.00	S-BODY ← C-CLAUSE , S-BODY	10.00	S-BODY ← NP , P-CLAUSE , VP
10.00	S-BODY ← NP , PP , VP	10.00	S-BODY ← NP , S-BODY , VP
0.50	S-BODY ← PP S-BODY	0.50	S-BODY ← V-ADJP , S-BODY
0.00001	S-BODY ← RB S-BODY	0.00001	S-BODY ← S-BODY RB
1.00	S-END ← !	1.00	S-END ← .
3.00	WH-S-BODY ← P-CLAUSE WH-S-BODY	3.00	WH-S-BODY ← PP , WH-S-BODY
3.00	YN-S-BODY ← BE EX NP	3.00	YN-S-BODY ← BE NP NP
3.00	YN-S-BODY ← BE NP VBD	3.00	YN-S-BODY ← BE NP VBD NP
3.00	YN-S-BODY ← BE NP VBG	3.00	YN-S-BODY ← BE NP VBG NP
3.00	YN-S-BODY ← C-CLAUSE , YN-S-BODY	3.00	YN-S-BODY ← C-CLAUSE YN-S-BODY
3.00	YN-S-BODY ← DO NP VB	3.00	YN-S-BODY ← DO NP VB NP
100.00	ADJP ← more JJ than NP	5.00	ADJP ← ADJP (ADJP)
5.00	ADJP ← ADJP CC ADJP	5.00	ADJP ← ADJP , ADJP
100.00	ADJP ← Either ADJP or ADJP	100.00	ADJP ← JJ V-INF
1.00	ADJP ← JJS	5.00	ADJP ← RB ADJP
100.00	ADJP ← as JJ as NP	100.00	ADJP ← either ADJP or ADJP
1.00	ADJP ← JJ	1.00	ADJP ← JJR
2.00	ADJP ← ADJP RB	100.00	BE ← either BE or BE
1.00	BE ← was	1.00	BE ← were
5.00	C-CLAUSE ← Although S-BODY	5.00	C-CLAUSE ← Because S-BODY
5.00	C-CLAUSE ← C-CLAUSE CC C-CLAUSE	10.00	C-CLAUSE ← C-CLAUSE , C-CLAUSE
100.00	C-CLAUSE ← Either C-CLAUSE or C-CLAUSE	5.00	C-CLAUSE ← If S-BODY
5.00	C-CLAUSE ← Though S-BODY	5.00	C-CLAUSE ← although S-BODY
5.00	C-CLAUSE ← if S-BODY	5.00	C-CLAUSE ← though S-BODY
5.00	C-CLAUSE ← for S-BODY	5.00	C-CLAUSE ← For S-BODY
5.00	C-CLAUSE ← in case S-BODY	5.00	C-CLAUSE ← In case S-BODY
1000.00	CC ← as well as	1000.00	CC ← so much so that
100.00	IN ← As to	100.00	IN ← Because of
1000.00	IN ← In case of	100.00	IN ← Due to
100.00	IN ← In addition to	100.00	IN ← In order to
100.00	IN ← Prior to	100.00	IN ← Rather than
100.00	IN ← Regardless of	100.00	IN ← according to

Figure 8.9: Examples of rules extracted from the context-free grammar

8.6 About the Parser

The parser is a bottom-up chart parser. It works for both sentence and word lattice parsing. Every constituent or structure derived during parsing is recorded as an edge. Because a CNF grammar is used here, every edge represents a completed constituent. This property restricts the number of edges in the chart during parsing. Edges are organized as trees in which some parent edges may share the same child edge. In more detail, the data structure of an edge can be described as

```
label
rule_used_to_derive_it
children_list
parents_list
(start_position, end_position)
confidence_score
```

When an edge is created, those fields are filled. The label of an edge can be an English word, a part-of-speech tag, or a syntactic or semantic category. Except for those word edges that are loaded by the parser at startup, a new edge usually is created by merging two child edges, or by rewriting a child edge according to some grammar rule. *rule_used_to_derive_it* records which rule was used to derive this edge. *children_list* records the child edges from which this edge was derived. *parents_list* records all parent edges of an edge. The coordinate *(start_position, end_position)* is used to indicate the location of an edge. If the location of edge A is (i, j) and the location of edge B is $(j + 1, k)$, we say edge A and edge B are adjacent edges. Only adjacent edges can be merged according to some rule in the grammar. The *confidence_score* of a new edge is used to measure the likelihood of that edge. It is a summation of the confidence scores of the rule and of its child edges. Any tree can be represented by its root node (which is an edge here).

Edges not only are organized as a forest according to their derivation relations, but also are organized as queues according to their locations. All edges, whose *(start_position, end_position)=(i,j)* , are linked together into a queue $bin[i][j]$. All edges in a queue are sorted by their confidence scores from high to low. The data structure for $bin[i][j]$ is

```
sorted_list_of_edges
state
forbidden_derivations_list
```

The state of a bin can be either ACTIVE or INACTIVE. If a bin is marked INACTIVE, any edges in this bin will be ignored. The *forbidden_derivations_list* of a bin is a list which records which edges has been derived from the edges in this bin so that it can be used to avoid the same edge from being created more than once.

It is evident that if an edge A is in $bin[i][j]$ and another edge B is in $bin[j+1][k]$, A and B are adjacent edges. If the length of the sentence to be parsed is n, there will be $(n+1)n/2$ bins. As we will show later, this type of organization is efficient for statistical parsing and for applying some simple constraints to reduce the search space.

There are 6 steps in the parser. They are:

1. Loading word sequence or word lattice;
2. Loading word tags;
3. Noun phrase parsing;
4. Marking some bins inactive;
5. Sentence parsing;

6. Checking chart to find a most preferred parse tree if it exists.

We will discuss these steps one by one.

Given a sequence of words $w_0 w_1 ... w_{n-1}$, for example:

```
Fruit  flies  like  apples   .
  0      1     3      4      5
```

the parser will load the words by creating an edge for each word. The label of an edge is the word. *rule_used_to_derive_it* will be set to NIL. *children_list* and *parents_list* will also be set to NIL. *confidence_score* will be initialized as 1.00 here. The coordinate *(start_position, end_position)* is used to indicate the position of an edge; here, it will be assigned by the location of the word in the input word sequence to be parsed. For the word "*Fruit*," its coordinates *(start_position, end_position)* will be $(0, 0)$. In our parser, punctuation marks are treated as individual English words.

Because the parser uses a lattice, it allows several words to have the same coordinate *(start_position, end_position)*. In this case, *confidence_score* will be set to the OCR confidence score of the word candidate. So you can see that our lattice parser is general regardless of whether the input is a word sequence or OCR output.

After loading words, the parser loads all possible tags for each word edge. Those tag edges are created as parents of the corresponding word edges. The confidence score for each tag edge is the sum of the confidence scores of its child word edges and the probability of the tag for the word which can be looked up in the lexicon. If a word is not in the lexicon, the parser performs morphological analysis to guess its potential tags. For example, if the unknown word begins with a digit, it will assumed to be *CD* (cardinal number); if it begins with a capital letter, its tags will be assumed as *NNP* (proper noun) and *NN*; if it ended with "*ed*," its tags will be assumed as *VBD*, *VBN*, and *JJ*. For those unknown words for which the morphological analysis provides no hints, the parser assumes the potential tags are *NN*, *VB* and *JJ*.

After loading words and tags, parsing begins. There are two stages of parsing: the first stage is *noun phrase parsing*; the second is *sentence parsing*. Between these two steps, there is a step to mark some bins inactive so that the search space can be reduced significantly. We will discuss that later. In the stage of noun phrase parsing, only a small subset of our grammar is used to derive all possible noun phrases. In the stage of sentence parsing, the whole grammar is used to further derive all possible parse trees. The parsing algorithm for both stages is described in Figure 8.10.

Here, in steps 5 and 6, the edges are enumerated from high confidence score to low confidence score according to the order of the edges in the *sorted_list_of_edges* of bin. The edge which has a higher confidence score usually was generated earlier and will have higher priority to be used for further derivations.

When parsing is finished, the parser decides whether it failed by checking if there exists any edges with label "*S*" in $bin[0][n-1]$, where n is the length of the sentence. If parsing succeeds, it will print the parse tree represented by the "*S*" edge with highest confidence score. If the user asks for all possible complete parse trees, the parser will generate them in order of decreasing confidence score. If parsing fails, the parser will work as a partial parser and print parse trees for fragments of the input word sequence.

If the sentence to be parsed or to be recognized is long, for example more than 20 words, the number of possible parse trees will be large. To parse such sentences using an undirected bottom-up strategy is not practical because the number of edges created will be enormous and the time for parsing will be very long. To make the parser work efficiently in such cases, we developed several methods to reduce the search space.

Assumption: n is the length of the sentence;
an edge is represented by its label.

```
REPEAT
  FOR i = 0 TO n − 1 DO
    FOR j = i TO n − 1 DO
      IF (bin[i][j].state == ACTIVE)
        FOREACH edge x in bin[i][j]
           IF there exists rule "z ← x"
             and it is not in forbidden list
             s ← score of edge x + score of rule "z ← x";
             IF there already exists an edge z in bin[i][j]
               modify edge z if its score is lower than s;
             ELSE
               create a new edge for z and insert it into bin[i][j];
             insert "z ← x" into forbidden list of bin[i][j];

           FOR k = j + 1 TO n − 1 DO
             FOR m = k TO n − 1 DO
               IF (bin[k][m].state == ACTIVE)
                 FOREACH edge y in bin[k][m]
                   IF there exists rule "z ← x y" and it is not in forbidden list
                     s ← score of edge x + score of edge y
                         + score of rule "z ← x y";
                     IF there already exits an edge z in bin[i][m]
                       modify the edge z if its score is lower than s;
                     ELSE
                       create a new edge for z and insert it into bin[i][m];
                     insert "z ← x y" into forbidden list of bin[i][m];
UNTIL no more edge can be created or modified, or memory overflow
```

Figure 8.10: Algorithm for parsing

In the probabilistic parser, what we want to obtain is usually just the most preferred complete parse tree with the highest confidence score. Although sometimes a constituent (an edge with a specific label) can be derived in several different ways, what we have to keep is just the best one. In our parser, if two edges have the same label at the same location, only the one with the higher confidence score will be kept. The parsing algorithm above reflects this strategy. If the parser finds a better way to derive an existing edge, it will just modify the edge without creating a new one.

By observing the correct complete parse trees of several sentences, we found that some partial parse trees generated during parsing contribute nothing to the complete parse trees. Let's consider a sequence of words to be parsed

$$w_0 ...... w_{i-1} w_i w_{i+1} ...... w_{n-1}$$

if w_i is the word "*a*" or "*the*," any constituent beyond *NP* at location $(i+1, j)$, where $i < j < n$, is not useful because "*a*" or "*the*" works as glue closely sticking to the word or the word sequence following it. After noun phrase parsing, we can mark $bin[i+1][j]$ $INACTIVE$, where $i < j < n$, without any cost. Because the frequency of "*a*" and "*the*" in English text is quite high, many bins will be marked *INACTIVE*. Because we have a special method to recognize the high frequency function words correctly, this constraint is still useful for lattice parsing.

Similarly, if a sequence of words to be parsed is

$$w_0 ...... w_{i-1} w_i w_{i+1} ...... w_{j-1} w_j w_{j+1} ...... w_{k-1} w_k w_{k+1} ...... w_{n-1}$$

where w_i, w_j and w_k are the punctuation marks "," (commas), $bin[p, q]$ can be marked as *INACTIVE*, where $i+1 < p < j-1$ and $j+1 < q < k-1$, because constituents in those bins usually cannot be involved in any complete parse trees with a high confidence score.

Therefore, after noun phrase parsing the parser will mark some bins *INACTIVE* before continuing with sentence parsing.

For the example in Figure 8.8, after lattice parsing, the most preferred parse tree is shown in Figure 8.11.

As you can see, the position of the *NP* for "*external security*" in the parse tree is wrong. This is reasonable because the parser does not yet have a special facility for processing conjunctions.

The word candidates involved in this parse tree are selected as the final recognition result for the word images. In this example, they are correct (see Figure 8.12).

8.7 Experimental Results

We chose three texts from the Brown Corpus and two texts from the Penn Treebank as testing samples. The Brown Corpus is a collection of 500 samples of English texts, each of which contains approximately 2000 words [89]. *A06* is a collection of six short articles from the *Newark Evening News. G02* is from an article, "*Toward a Concept of National Responsibility*," from *The Yale Review. J42* is from a book, "*The Political Foundation of International Law.*" The text "*t/t1*" is from the file "*treebank/postext/t/t1*" in the Penn Treebank. It was taken from "*Library of America texts.*" The text *w00* is a collection of articles from the "*Wall Street Journal.*" Some further information about the sample articles is listed in Table. 8.2.

First, the parser was applied to sentences from those articles. The results of parsing on the plain ASCII text are shown in Table. 8.3. For 2835 out of 3354 sentences(84.5%), the parser generated at least one complete parse tree. We found that sentences without a complete parse tree contained some syntactic structures that were not covered by the grammar. Many of the failures were caused by quotations, constituent inversion, or a missing constituent. The parser generated partial parse trees for most of the sentences that were not parsed completely. We manually checked the most preferred parse trees generated

```
( S
   ( S-BODY
      ( PP
         ( IN  in  )
         ( NP
            ( SIMPLE-NP
               ( DT  the  )
               ( NN  past  )  )  )  )
      ,
      ( S-BODY
         ( NP
            ( SIMPLE-NP
               ( DT  the  )
               ( NNS  duties  )  )
            ( IN  of  )
            ( NP
               ( SIMPLE-NP
                  ( DT  the  )
                  ( NN  state  )  )  )  )
         ,
         ( P-CLAUSE  as
            ( S-BODY
               ( NP
                  ( SIMPLE-NP
                     ( NNP
                        ( NNP
                           ( NNP  Sir  )
                           ( NNP  Henry  )  )
                        ( NNP  Maine  )  )  )  )
               ( VP
                  ( VPP
                     ( VB
                        ( VPP
                           ( VBD  noted  )  )
                        ( RB  long  ago  )  )  )  )  )  )
         ,
         ( VP
            ( VPP
               ( VBD  were  )  )
            ( NP
               ( NP
                  ( SIMPLE-NP
                     ( NN
                        ( JJ  only  )
                        ( NN
                           ( CD  two  )  )  )  )
                  ( IN  in  )
                  ( NP
                     ( NP
                        ( SIMPLE-NP
                           ( NN  number  )  )  )
                     :
                     ( NP
                        ( SIMPLE-NP
                           ( NN
                              ( JJ  internal  )
                              ( NN  order  )  )  )  )  )  )
               and
               ( NP
                  ( SIMPLE-NP
                     ( NN
                        ( JJ  external  )
                        ( NN  security  )  )  )  )  )  )  )  )
   ( S-END  .  )  )
```

Figure 8.11: The most preferred parse tree for the word lattice

position	0	1	2	3	4	5	6	7	8	9
top1		the		,	the	duties	of	the		,
top2										
top3										
top4										
top5	in		past						state	

position	10	11	12	13	14	15	16	17	18	19
top1	as	Sir		Maine	noted		ago	,		only
top2			Henry			long			were	
top3										
top4										
top5										

position	20	21	22	23	24	25	26	27	28	29
top1			number	:		order	and			.
top2	two				internal			external	security	
top3		in								
top4										
top5										

Figure 8.12: The words selected from the word lattice after lattice parsing

Article	Num. of Words	Num. of Sentences	Average Length of Sentences	Maximal Length of Sentences
A06	2213	88	25.1	70
G02	2267	88	25.7	68
J42	2269	91	24.9	71
t/t1	18047	1114	16.2	64
w00	45774	1973	23.2	67
total	70570	3354	21.0	71

Table 8.2: Information about the test samples

for the sentences from *G02* to see whether they were correct. For 55 out of 71 sentences, the most preferred parse trees are basically correct if we ignore prepositional phrase attachment and conjunct scoping problems. For the remaining sentences from *G02* with complete parse trees, the most preferred parse trees were partially correct. In Table. 8.3, the high standard deviation(σ) values show that the number of edges created, the parsing time, and the number of complete trees generated vary significantly from sentence to sentence.

Article	Num. of Sentences	Num. of Sentences with Complete Parse Tree	Avg Num. of Edges Generated	Avg Time of Parsing (sec.)	Avg Num. of Complete Parse
A06	88	72	2950 $\sigma = 2484$	34.4 $\sigma = 45.8$	20.1 $\sigma = 58.0$
G02	88	71	2461 $\sigma = 3572$	27.2 $\sigma = 73.9$	34.3 $\sigma = 89.4$
J42	91	62	1904 $\sigma = 1664$	14.4 $\sigma = 17.4$	11.4 $\sigma = 20.6$
t/t1	1114	981	1337 $\sigma = 2469$	11.2 $\sigma = 40.8$	20.2 $\sigma = 270.5$
w00	1973	1649	2772 $\sigma = 3445$	57.3 $\sigma = 137.8$	32.2 $\sigma = 131.1$
total	3354	2835 (84.5%)	2269 $\sigma = 3168$	39.4 $\sigma = 111.4$	27.34 $\sigma = 188.8$

Table 8.3: Results of sentence parsing

Next we used all the sentences from the three articles: *A06*, *G02*, and *J42*, to test the application of the parser to text recognition. After printing those sentences with *ditroff* in a 12 point font on paper using a laser printer, text images were created by digitizing the printouts. The text images were segmented into sentences, and further into word images. Using a word recognition program which is based on word shape analysis [55], the top ten word candidates were generated for each word image (for the frequent function words such as "*a*," "*the*" and "*of*," and punctuation marks, the recognizer generates just one word candidate).

The outline of word shape analysis algorithm is shown in Figure. 2.3. The word shape analysis algorithm attempts to describe and compare the shape of the word as a whole object. Every word image is first partitioned into a fixed 4×10 grid to provide a frame of reference. Features that describe the details of the word shape are then extracted and their relative locations in the grid are recorded in a feature vector. The feature vector is matched to a large lexicon of words (with near 70,000 different word entries here) where each entry contains a similar feature vector for the corresponding word. Ten word candidates that produce the minimum distance to the input feature vector, are generated from the dictionary.

On average, there are 7.78 word candidates for each word image (see Table. 8.4). In this experiment, the probabilities of each word candidate being correct were assumed to be equal. The system (including the relaxation algorithm and parser) was used to select decisions from the top-10 choices. Because there are on average 7.78 word candidates for each word image, if we choose a word candidate by chance, the probability of correctness is about 13 percent.

Article	Num. of Words	Num. of Sentences	Average Num. of Word Candidates per Word Image
A06	2213	88	7.80
G02	2267	88	7.67
J42	2269	91	7.86
total	6749	267	7.78

Table 8.4: Word Recognition Result

The relaxation algorithm generates the two best choices for each word using word collocation constraints. Here, only collocation constraints for adjacent words are used. The data for word collocation constraints were collected by training on the Brown Corpus.

If we use word collocation data collected from the Brown Corpus and include the test articles, the performance of relaxation is about 98% correct(see Table. 8.5). Among 267 sentences, the parser produced at least one complete parse tree for 249(93%) of them. After relaxation, about 97.9% of the correct word candidates still remained in the top-2 lists. The parser selected word candidates from the top-2 correctly for 90.9% of the words and also provided a parse tree for further processing. The average time for parsing a sentence is less than four minutes while the average number of edges created in the parser is about 10,000.

Article	Num. of Word Images	Num. of Corr. Candidates Remaining in top-2 After Relaxation	Num. of Corr. Candidates Selected After Parsing	Avg Num. of Edges Generated per Sentence	Avg Time of Parsing per Sentence (sec.)	Avg Num. of Complete Parses per Sentence
A06	2213	2187 (98.8%)	2078 (93.9%)	9751 $\sigma = 8557$	289.3 $\sigma = 361.5$	110.9 $\sigma = 276$
G02	2267	2196 (96.8%)	2030 (89.5%)	9243 $\sigma = 11604$	145.3 $\sigma = 215.4$	356.0 $\sigma = 980$
J42	2269	2230 (98.2%)	2030 (89.5%)	11578 $\sigma = 13874$	191.6 $\sigma = 242.9$	282.7 $\sigma = 715$
total	6749	6613 (97.9%)	6138 (90.9%)	10206 $\sigma = 11624$	208.5 $\sigma = 286.4$	252.8 $\sigma = 733$

Table 8.5: Result of Word Lattice Parsing(using whole Brown Corpus as Training Data)

If we use word collocation data collected from all the articles in the Brown Corpus except the test articles, the performance of the parser drops(see Table. 8.6). The overall correct rate of relaxation is 86.4% and the parser correctly selects 77.6% of the words. The reason for the drop is that the relaxation algorithm did not have as reliable collocation data as before. This result suggests that the training corpus is not large enough.

Article	Num. of Word Images	Num. of Corr. Candidates Remaining in top-2 After Relaxation	Num. of Corr. Candidates Selected After Parsing	Avg Num. of Edges Generated per Sentence	Avg Time of Parsing per Sentence (sec.)	Avg Num. of Complete Parses per Sentence
A06	2213	1861 (84.1%)	1718 (77.6%)	9985 $\sigma = 9302$	265.1 $\sigma = 379.3$	108.6 $\sigma = 273$
G02	2267	2004 (88.4%)	1800 (79.4%)	9139 $\sigma = 11488$	142.1 $\sigma = 200.6$	323.2 $\sigma = 868$
J42	2269	1972 (86.9%)	1723 (75.9%)	11701 $\sigma = 13943$	193.2 $\sigma = 249.4$	287.9 $\sigma = 785$
total	6749	5837 (86.4%)	5241 (77.6%)	10291 $\sigma = 11808$	200.0 $\sigma = 290.5$	241.6 $\sigma = 704$

Table 8.6: Result of Word Lattice Parsing(using all texts in Brown Corpus except A06, G02 and J42 as Training Data)

We further extended the training set to all the "*Wall Street Journal*" texts(with over two million words) in the Penn Treebank. By using the new word collocation data, the overall performance of the parser improved by about 2% (see Table. 8.7). If we collect word collocation data by training on larger corpora, performance should improve further.

Article	Num. of Word Images	Num. of Corr. Candidates Remaining in top-2 After Relaxation	Num. of Corr. Candidates Selected After Parsing	Avg Num. of Edges Generated per Sentence	Avg Time of Parsing per Sentence (sec.)	Avg Num. of Complete Parses per Sentence
A06	2213	1959 (88.5%)	1807 (81.2%)	9998 $\sigma = 9223$	213.1 $\sigma = 415.9$	82.9 $\sigma = 170$
G02	2267	2003 (88.4%)	1791 (79.0%)	9326 $\sigma = 11920$	140.3 $\sigma = 196.6$	337.1 $\sigma = 866$
J42	2269	2020 (89.0%)	1754 (77.3%)	11695 $\sigma = 13796$	232.2 $\sigma = 377.4$	291.8 $\sigma = 918$
total	6749	6982 (88.6%)	5352 (79.3%)	10355 $\sigma = 11863$	195.6 $\sigma = 346.2$	239.5 $\sigma = 748$

Table 8.7: Result of Word Lattice Parsing(using all texts in Brown Corpus except A06, G02 and J42 plus all "*Wall Street Journal*" texts in Treebank as Training Data)

8.8 Conclusions and Further Directions

In this chapter, we described an experimental approach to using language syntax to post-process word recognition results. Word candidate selection is formalized as a lattice parsing task. A probabilistic lattice chart parser has been designed. A large-scale English lexicon and a broad-coverage English grammar have been edited manually for unrestricted English text processing. Because syntactic constraints sometimes are not strict enough to choose among competitive candidates which are from the same syntactic category and lattice parsing is computationally expensive if the number of candidates for a word image is high, the word collocation constraint is applied to improve the quality of the top choices and to reduce the number of candidates before lattice parsing. The integration of the approaches of structural analysis and statistical language models provides a way to overcome the limitations of each approach.

There are several directions to make lattice parsing more practical and to improve its performance. Better parsing schema, a more realistic English grammar and richer English lexicon, can be used. More complete and balanced word collocation data can also be applied to the task by training on larger English text corpora. In the current lattice parser, only linguistic constraints are utilized. Other contextual constraints, such as visual inter-word relations, can also be integrated.

Chapter 9

Integrating Visual and Linguistic Constraints for Candidate Selection

In previous chapters, visual inter-word constraints were studied for OCR postprocessing, and linguistic constraints were studied for word candidate selection. In this chapter, we will focus on the integration those two types of contextual constraints to achieve better performance in candidate selection. Two experiments are reported in this chapter. In the first experiment, the word-collocation-based relaxation algorithm proposed in Chapter 7 is augmented with image equivalence constraints in order to achieve better performance. In the second experiment, a procedure in which image-based contextual analysis and language-model-based analysis proposed separately in previous chapters are combined, is designed and tested to demonstrate the integrated approach.

9.1 Introduction

A word image from a degraded text page may have touching, broken, distorted or blurred characters, which may make the word image difficult to recognize accurately. After character recognition and correction based on dictionary look-up, a word recognizer will provide one or more word candidates for each word image. Each word candidate has a confidence score, but the score may not be reliable because of noise. The correct word candidate is usually in the candidate set, but may not be the candidate with the highest confidence score. Instead of simply picking the candidate with the highest recognition score, which may make the correct rate quite low, we need to find a method which can select a candidate for each word image so that the correct rate can be as high as possible.

Linguistic knowledge can be used to select a decision word for each word image in its context. Currently, there are two approaches, the statistical approach and the structural approach, towards the problem of candidate selection. In the statistical approach, language models such as a *Hidden Markov Model* and word collocation can be utilized for candidate selection [26, 58, 70]. In the structural approach, lattice parsing techniques have been developed for candidate selection [62, 152].

The contextual constraints considered in a statistical language model, such as word collocation, are local constraints. For a word image, a candidate will be selected according to the candidate information from its neighboring word images in a fixed window size. The window size is usually set as one or two. In the lattice parsing method, a grammar is used to select a candidate for each word image inside a sentence so that the sequence of the selected candidates forms a grammatical and meaningful sentence.

Visual inter-word relations can be used as global constraints in the process of word image interpretation. It is not surprising that word relations at the image level are highly consistent with word relations at the symbolic level (see Table 4.1). If two words hold a relation at the symbolic level and they are written in the same font and size, their word images should keep the same relation at the image level. And also, if two word images hold a relation at the image level, the truth values of the word images should have the same relation at the symbolic level. In Figure 4.2, for example, word images *2* and *8* must be recognized as the same word because they can match each other; the identity of image *5* must be a sub-string of the identity of image *6* because image *5* can match with a part of image *6*; and so on.

Candidate selection can be viewed as a constraint satisfaction problem. Here, we have two distinct types of constraints to be considered: visual constraints and linguistic constraints. Intuitively, as shown above, their integration is promising.

9.2 Relaxation with Word Collocation and Image Equivalence Constraints

Word collocation can be considered as the constraint on candidate selection so that word candidate selection problem can be formalized as an instance of a constraint satisfaction problem. Relaxation is a typical solution for constraint satisfaction problems. Therefore, a relaxation-based algorithm was proposed in Chapter 7 for the task of candidate selection. Figure 9.1 shows a lattice which is the word recognition for two English sentences. Figure 9.2 is the lattice for those sentences after applying the relaxation algorithm.

Because the window size of word collocation is usually small, word collocation is a local constraint. Because word collocation is statistical data trained from text corpora, it usually is incomplete and unbalanced. Those properties limit the effect of word collocation for candidate selection. By analyzing the behavior of the algorithm, three types of errors were identified:

1. The local context can not provide enough information to distinguish the competitive candidates; For example, it is hard to select the candidate "*farm*" or "*form*" correctly in the sentence, "*This [farm/form] is almost the same as that one*" because they both fit the local context.

2. Word collocation data trained from corpora is not complete so that it does not include the statistical data needed to select candidate correctly; For example, the algorithm may not choose the correct candidate "*duckling*" in the sentence, "*He was once an ugly [buckling/ducking]*," because there is no collocation data available about the word pair *(ugly, duckling)*.

3. Word collocation data trained from unbalanced corpora may be biased. For example, in the sentence "*This [farm/form] is almost the same as that one*," where "*farm*" is the correct choice, "*form*" may be preferred wrongly because word collocation data from corpora shows that the collocation of "*(This, form)*" is much stronger than that of "*(This, farm)*."

In summary, as statistical data, word collocation represents local constraints which might not be sufficient for the relaxation algorithm to select candidates with high accuracy. To improve the performance of the algorithm, it is desirable to introduce other more global constraints. Intuitively, visual inter-word relations are such a global constraint. For example, suppose we know that images 4 and 11 in Figure 9.1 are visually equivalent. Figure 9.3 shows the lattice after integrating the confidence score of each candidate for images 4 and 11. After integration, images 4 and 11 are interpreted consistently as the same word.

In the rest of this section, we will exploit only the equivalence relation listed in Figure 4.1 and focus on its role in the relaxation algorithm as a global constraint.

In a normal English text, there are many occurrences of the same words. Language statistics indicate that function words such as "*the*," "*of*," "*to*" and "*and*" account for about 30 percent of words in a English text [89]. Those words which are related to the topics of the text usually also appear with high frequency inside the text. Usually, the main body of a text is prepared in the same font type. Therefore, different occurrences of the same word can be visually similar even if the text image is degraded.

Sentence 1

	1	2	3	4	5	6	7	8	9
top1	Troy	soil	their	form	products	in	toe	market	.
	0.50	0.33	0.90	0.33	0.63	0.70	0.35	0.85	1.00
top2	Tray	salt	theft	farm	produces	In	the	martial	
	0.23	0.30	0.05	0.32	0.30	0.10	0.33	0.05	
top3	They	cell	chair	forth	precincts	th	lire	marries	
	0.20	0.21	0.02	0.15	0.05	0.09	0.20	0.04	
top4	Tory	sail	than	foam	producer	ill	she	mother	
	0.05	0.09	0.02	0.13	0.01	0.09	0.07	0.04	
top5	Tiny	sell	then	force	produced	Io	tire	matter	
	0.02	0.07	0.01	0.07	0.01	0.02	0.05	0.02	

Sentence 2

	10	11	12	13	14	15	16	17	18	19
top1	This	form	ie	amidst	toe	some	as	chat	one	.
	0.90	0.33	0.40	0.30	0.35	0.75	0.95	0.66	0.91	1.00
top2	Tire	farm	is	almost	the	came	SE	blot	cue	
	0.04	0.32	0.35	0.30	0.33	0.15	0.02	0.16	0.06	
top3	Tok	forth	le	utmost	lire	same	SS	that	COB	
	0.03	0.15	0.15	0.25	0.20	0.08	0.01	0.15	0.01	
top4	True	foam	Ca	somber	she	score	CS	mat	ore	
	0.02	0.13	0.07	0.11	0.07	0.01	0.01	0.02	0.01	
top5	Tina	force	k	aimless	tire	arms	BS	boat	CDC	
	0.01	0.07	0.03	0.04	0.05	0.01	0.01	0.01	0.01	

Figure 9.1: Word candidate sets of two example sentences: "*They sell their farm products in the market. This farm is almost the same as that one.*"

Sentence 1

	1	2	3	4	5	6	7	8	9
top1	They	sell	their	farm	products	in	the	market	.
	0.99	0.97	0.76	0.80	0.89	0.95	0.99	0.87	1.00
top2		soil	then	force	produced	In		matter	
		0.03	0.05	0.15	0.06	0.04		0.09	
top3			theft	form	producer			mother	
			0.05	0.03	0.03			0.03	
top4			chair	foam	produces				
			0.05	0.02	0.02				

Sentence 2

	10	11	12	13	14	15	16	17	18	19
top1	This	form	is	almost	the	same	as	that	one	.
	0.99	0.35	0.99	0.98	0.99	0.73	0.99	1.00	0.99	1.00
top2		force		somber		came				
		0.32		0.01		0.19				
top3		farm				some				
		0.24				0.04				
top4		foam				score				
		0.09				0.03				

Figure 9.2: Word lattice of the example sentences after relaxation and removal of candidates with low confidence

Sentence 1

	1	2	3	4	5	6	7	8	9
top1	They	sell	their	farm	products	in	the	market	.
	0.99	0.97	0.76	0.52	0.89	0.95	0.99	0.87	1.00
top2		soil	then	force	produced	In		matter	
		0.03	0.05	0.24	0.06	0.04		0.09	
top3			theft	form	producer			mother	
			0.05	0.19	0.03			0.03	
top4			chair	foam	produces				
			0.05	0.05	0.02				

Sentence 2

	10	11	12	13	14	15	16	17	18	19
top1	This	farm	is	almost	the	same	as	that	one	.
	0.99	0.52	0.99	0.98	0.99	0.73	0.99	1.00	0.99	1.00
top2		force		somber		came				
		0.24		0.01		0.19				
top3		form				some				
		0.19				0.04				
top4		foam				score				
		0.05				0.03				

Figure 9.3: Word lattice of the example sentences after using visual context (assuming that images 4 and 11 are visually equivalent)

If two word images are detected as visually equivalent, they hold the equivalence relation. Given a text page, all word images can be extracted after layout analysis. The word image clustering algorithm described in Chapter 4 can be applied to identify visual equivalence relations among words in the text page.

9.2.1 Relaxation Augmented with Image Equivalence Constraints

Figure 9.4 is the description of the relaxation algorithm which integrates word collocation and visual inter-word constraints for candidate selection in text recognition. Given a sequence of word images from a text page, the algorithm outputs a decision for each image. The first step of the algorithm is word image clustering in order to find all possible visual equivalence relations. For each cluster, an image prototype is generated to represent images in the cluster. A word recognizer is then applied on the prototype of each cluster to generate a set of word candidates. Each word image inside a cluster inherits the candidate set of the cluster. For each candidate, there is a probabilistic score which indicates the confidence of the recognizer in the candidate. Because of the noise inside the image, the score is not necessarily reliable. After an iteration of relaxation, probabilistic scores of candidates of an image are upgraded based on word collocation data. The probabilistic scores of the candidates of a cluster is upgraded by summing up the probabilistic scores of the word images inside the cluster. Then the candidates of each cluster are sorted in decreasing order of confidence value. Next, each word image inherits its candidate set from the cluster it belongs to. When there is no significant change in confidence scores, the relaxation stops. The top candidate of each word image is selected as the decision.

9.2.2 Experiments and Analysis

Five articles from the Brown Corpus were selected as testing samples. They are *A06*, *G02*, *J42*, *N01* and *R07*. Each text has about 2000 words. There are a total of 11,402 words in

INPUT: A sequence of word images W[i], i = 1,2,...n, which are extracted from a text page.
OUTPUT: W[i].decision, for each i = 1,2,...n

```
/*Word Image Clustering*/
ClusterList ← ∅;
FOR i=1 to n DO
    FoundMatch ← FALSE;
    FOR each cluster C[j] in ClusterList DO
        IF (DistanceOfImage(W[i].image, C[j].prototype) < threshold)
            C[j].ImageList ← C[j].ImageList ∪ {W[i]};
            W[i].ClusterIndex ← j;
            FoundMatch ← TRUE;
    IF (FoundMatch == FALSE )
        Create a new cluster C[k];
        C[k].ImageList ← {W[i]};
        W[i].ClusterIndex ← k;
        ClusterList ← ClusterList ∪ {C[k]};

/*Isolated Word Recognition–Candidate Generation*/
FOR each cluster C[j] in ClusterList DO
    C[j].CandidateList ← WordRecognition(C[j].prototype);
    Sort candidates in C[j].CandidateList in decreasing order;

IterationCount ← 0;
REPEAT
    IterationCount ← IterationCount + 1;
    /* Generate Word Lattice */
    FOR each word image W[i] DO
        W[i].CandidateList ← C[W[i].ClusterIndex].CandidateList;

    /* Compute New Confidence Scores For Candidates Of Word Images*/
    FOR each word image W[i] DO
        FOR each word candidate w[m] in W[i].CandidateList DO
            Upgrade w[m].prob by using word collocation;

    /* Compute New Confidence Scores For Candidates Of Clusters*/
    FOR each cluster C[j] in ClusterList DO
        FOR each candidate c[n] in C[j].CandidateList DO
            c[n].prob ← 0.0;
            FOR each word image W[i] in C[j].ImageList DO
                FOR each word candidate w[m] in W[i].CandidateList DO
                    IF (c[n].string == w[m].string)
                        c[n].prob ← c[n].prob + w[m].prob;
        Sort candidates in C[j].CandidateList in decreasing order;
UNTIL probabilistic scores of word candidates become stable;
/* Generate Word Lattice */
FOR each word image W[i] DO
    W[i].CandidateList ← C[W[i].ClusterIndex].CandidateList;

/* Select Best Candidate */
FOR each word image W[i] DO
    W[i].decision ← CandidateWithHighestScore(W[i].CandidateList);
END
```

Figure 9.4: Relaxation algorithm

those testing samples (see Table 9.1). For each word in those texts, the top10 word candidate lists were generated. The dictionary size is about 70,000. We simulated performance on highly degraded text. The performance model used here has top1 correct rate of around 55%.

The word collocation data we used was trained on the Penn Treebank and the Brown Corpus after removing the five testing samples from the Brown Corpus. The Brown corpus is divided into 500 samples of approximately 2000 words each [89]. The part of the Penn Treebank used here is the collection of articles from the *Wall Street Journal* that contains three million words. We used the frequency of a word pair to measure its collocation strength. There are a total of 1,200,000 word pairs after training.

We also simulated the best performance of word image clustering so that all images with the same truth value are placed in the same cluster. Table 9.2 lists the top 20 largest clusters for each article.

Article	Number Of Words	Number Of Different Words	Number Of Words Which Occur More Than Once
A06	2213	788	245
G02	2267	787	238
J42	2269	712	257
N01	2313	628	265
R07	2340	680	252

Table 9.1: Information About Testing Samples

The result of applying the original relaxation algorithm to the noisy text images is shown in Table 9.3. The top1 correct rate of isolated word recognition is as low as 57%. Relaxation based on word collocation can improve the top1 correct rate to 83%. After integrating word collocation and visual constraints, the correct rate of the first choice can be further improved to 88%. There is a 5% improvement by introducing visual contextual constraints. Here, testing samples were not used to train word collocation data. We also

A06		G02		J42		N01		R07	
freq	word	freq	word	freq	word	freq	word	freq	word
150	the	184	the	145	the	156	.	158	.
87	.	115	of	116	,	136	,	116	,
86	,	109	,	99	of	115	the	82	the
77	of	83	.	75	.	73	he	55	of
53	a	55	to	68	to	57	and	53	a
52	to	50	in	66	and	45	to	53	and
43	in	46	and	53	was	44	a	45	to
37	and	42	a	48	law	37	of	45	was
33	he	40	is	44	in	36	in	38	she
31	for	30	that	36	a	32	was	37	that
28	is	26	it	25	as	30	her	31	had
26	said	21	for	23	that	28	Morgan	30	in
23	state	21	was	22	it	25	she	29	he
19	on	18	responsibility	20	by	24	said	24	Arlene
18	that	17	as	20	international	23	you	24	it
16	Republican	16	–	20	with	22	his	24	they
16	will	16	by	19	were	20	had	20	but
15	by	13	has	16	;	19	him	20	on
15	his	13	which	16	this	19	on	19	this
15	with	12	national	15	be	18	that	18	her

Table 9.2: Top20 Most Frequent Words In Testing Articles

ran the algorithm on the testing samples by using word collocation data which includes the samples for training. In this ideal case, relaxation based on word collocation improves the top1 correct rate to 91.08%. After integrating word collocation and visual constraints, the correct rate of the top1 is 93.69%.

	Top10 Correct Rate									
Article	1	2	3	4	5	6	7	8	9	10
	Word Recognition									
A06	55.82%	77.77%	88.75%	90.78%	92.91%	93.99%	94.89%	97.29%	98.33%	100%
G02	67.71%	83.90%	90.03%	92.19%	93.74%	95.41%	96.29%	97.79%	99.12%	100%
J42	54.52%	79.73%	87.09%	89.33%	91.19%	92.60%	93.36%	94.36%	99.03%	100%
N01	57.33%	82.53%	90.66%	91.83%	92.65%	93.17%	93.82%	95.16%	97.71%	100%
R07	52.22%	68.68%	81.20%	85.00%	91.92%	92.74%	94.66%	95.17%	96.71%	100%
Total	57.10%	78.47%	87.51%	89.79%	92.47%	93.57%	94.59%	95.93%	98.16%	100%
	Relaxation without using visual inter-word constraints(10 iterations)									
A06	83.15%	92.86%	97.06%	97.88%	98.73%	99.14%	99.46%	99.59%	99.68%	100%
G02	83.81%	92.94%	96.16%	97.40%	98.72%	99.25%	99.51%	99.78%	99.87%	100%
J42	83.65%	93.08%	96.74%	97.97%	98.55%	99.07%	99.38%	99.43%	100%	100%
N01	82.71%	93.51%	96.93%	98.36%	99.09%	99.35%	99.61%	99.87%	100%	100%
R07	82.69%	92.56%	95.51%	96.58%	97.99%	98.59%	99.15%	99.40%	99.70%	100%
Total	83.19%	92.99%	96.47%	97.63%	98.61%	99.07%	99.42%	99.61%	99.85%	100%
	Relaxation using visual inter-word constraints(10 iterations)									
A06	88.52%	95.30%	97.47%	98.55%	99.01%	99.14%	99.19%	99.32%	99.41%	100%
G02	87.87%	94.40%	96.69%	97.88%	98.59%	98.94%	99.12%	99.16%	99.21%	100%
J42	89.51%	95.37%	98.10%	98.94%	99.16%	99.38%	99.56%	99.56%	99.56%	100%
N01	87.12%	95.03%	98.10%	99.09%	99.52%	99.65%	99.70%	99.74%	99.74%	100%
R07	88.16%	94.87%	96.54%	97.69%	98.29%	98.72%	98.85%	99.02%	99.06%	100%
Total	88.22%	94.99%	97.37%	98.42%	98.91%	99.16%	99.28%	99.35%	99.39%	100%

Table 9.3: Relaxation Results

9.2.3 Discussion

Based on word candidates provided by an isolated-word recognizer for each word image, high-level knowledge sources can be applied to select a candidate for each word image. A word-collocation-based relaxation algorithm was proposed for this task. Word collocation is a local statistical constraint, which sometimes is not sufficient to distinguish among candidates. To make candidate selection more accurate, the visual equivalence constraint has been investigated. Visual inter-word relations provide a way to link word images in the text and to interpret them systematically. A modified relaxation algorithm in which word collocation and visual constraints are integrated was designed. Experimental results showed that the modified algorithm improved performance.

9.3 Word Candidate Selection Using Visual and Linguistic Constraints

In the previous section we reported experiments that used image equivalence constraints within the word-collocation-based relaxation algorithm. The results demonstrated that by integrating visual and linguistic constraints, text recognition on degraded documents can be improved more than if just linguistic constraints were used.

In this section, we will extend the approach. Different types of visual inter-word relations are considered. The focus of this work is on using these two constraints extensively and effectively.

Given a sequence of word images, it is very time-consuming to compute all possible visual inter-word relations directly by image matching. Under such an image-driven approach, every two images have to be compared to see if they have the visual inter-word relations defined in Table 4.1. A more efficient approach is to compute visual inter-word relations using the "symbolically-driven" approach.

Given a sequence of word images, after running a word recognition algorithm, a list of word candidates is generated for each image. Those word candidates are hypotheses for the interpretation of the images. By analyzing the inter-word relations between two word candidates at the string level, hypotheses about visual relations are generated. Given a visual relation hypothesis, it is then tested by image matching. In such a way, we can avoid exhaustive search at the image level but still find a set of visual inter-word relations. Those visual inter-word relations can be used to reduce the size of candidate lists or to select a candidate as the decision for those related word images.

For word images at different positions in a text page, the difficulty of choosing a decision word from the candidate list varies significantly. There are two major factors which contribute to the difficulty of candidate selection:

1. local linguistic context;

2. the competition among words in the candidate list for the position.

Our previous experiments on a word-collocation-based relaxation algorithm showed that a large portion of word images (about 40% of the words in a text) can be located and the correct rate of candidate selection is very high (over 99%) for that portion, although the accuracy of the algorithm on the whole page is only about 85%. To improve the overall performance, instead of applying the same techniques uniformly on all word images we apply different techniques progressively to different word images. Such a "*divide-and-conquer*" strategy is important not only for accuracy considerations but also for efficiency. Another characteristic of such a strategy is its adaptability. For those word images for which it is easy to determine decisions with high confidence, the decisions can be used to improve results on other images.

9.3.1 A Procedure to Integrate Visual and Linguistic Constraints

Here, we propose a procedure for word candidate selection by exploiting visual and linguistic constraints. Given a sequence of word images extracted from a document page, the steps of the recognition procedure are described below (also see Figure 9.5):

- **Step 1–Word Recognition**: Apply a word recognition algorithm to generate a candidate list for each word image; The word recognition algorithm can be an OCR with dictionary-based postprocessing, or a word-shape-analysis algorithm. For each word image, a list of word candidates (for example, the top10 list) is provided. Each candidate has a confidence score. Candidates are ranked by their confidence scores inside the list.

- **Step 2–Word Image Clustering**: Cluster word images based on their visual equivalence; By image matching, word images which are almost equivalent can be grouped. Images from the same cluster hold the type one (equivalence) inter-word relations in Table 4.1.

- **Step 3–Word-Collocation-Based Relaxation**: Apply a word-collocation-based relaxation algorithm to re-evaluate confidence scores of candidates and to re-rank candidates by their new confidence scores inside a candidate list.

- **Step 4–Candidate Filtering**: Remove those candidates with very low confidence for each word image; After word-collocation-based relaxation, the confidence score of a word candidate is adjusted by considering whether it fits its local linguistic context. The summation of confidence scores of all candidates for a word image is equal to 1.0. If the confidence score of a candidate becomes very low, it usually means that the candidate does not make sense in its local linguistic context and there exists one or more other competitive candidates which have high confidence. In this case, the candidate with low confidence can be deleted from the candidate list without taking too much risk.

- **Step 5–Candidate Evaluation with Image Equivalence Constraints**: Adjust confidence scores of word candidates using image equivalence constraints; For those word images in the same cluster, the configurations of candidate lists should be consistent because they are visually equivalent images. Visually equivalent images are instances of the same word which occur in different parts of a text. However, after word-collocation-based relaxation, configurations of candidate lists of visually equivalent images usually become different. For example, given two images which are visually equivalent, the most preferred candidate for one word image may be quite different from that of another word image because their linguistic contexts are different. This kind of inconsistency can be detected and alleviated by using image equivalence constraints.

- **Step 6–Visual Relation Hypothesis Generation**: Generate hypotheses of visual inter-word relations by analyzing possible symbolic relations between word candidates for different word images (see Figure 9.6 for an example).

- **Step 7–Visual Relation Hypothesis Test**: Test hypotheses of visual inter-word relations by image matching; Given two word images and a hypothesis about a possible inter-word relation between them, an image matching algorithm determines whether such a relation exists at the image level.

- **Step 8–Candidate Evaluation by Consistency Analysis**: Adjust confidence scores of word candidates by consistency analysis between the image and the symbolic levels; This step is similar to step 5. If two word images hold an inter-word relation at the image level, their decision words usually keep the same relation at the symbolic level. In other words, if two candidates from two visually related images keep the same relation at the string level, they are more likely to be selected as the decisions for those images than any two candidates which can not keep the relation at the string level. The confidence score of a candidate can be increased or decreased depending on whether relation consistency can be found or not. After this, candidates will be re-ranked by their new confidence scores.

- **Step 9–Candidate Filtering**: Same as step 4.

- **Step 10–Candidate Selection**: Select the candidate with highest confidence as the decision for each image.

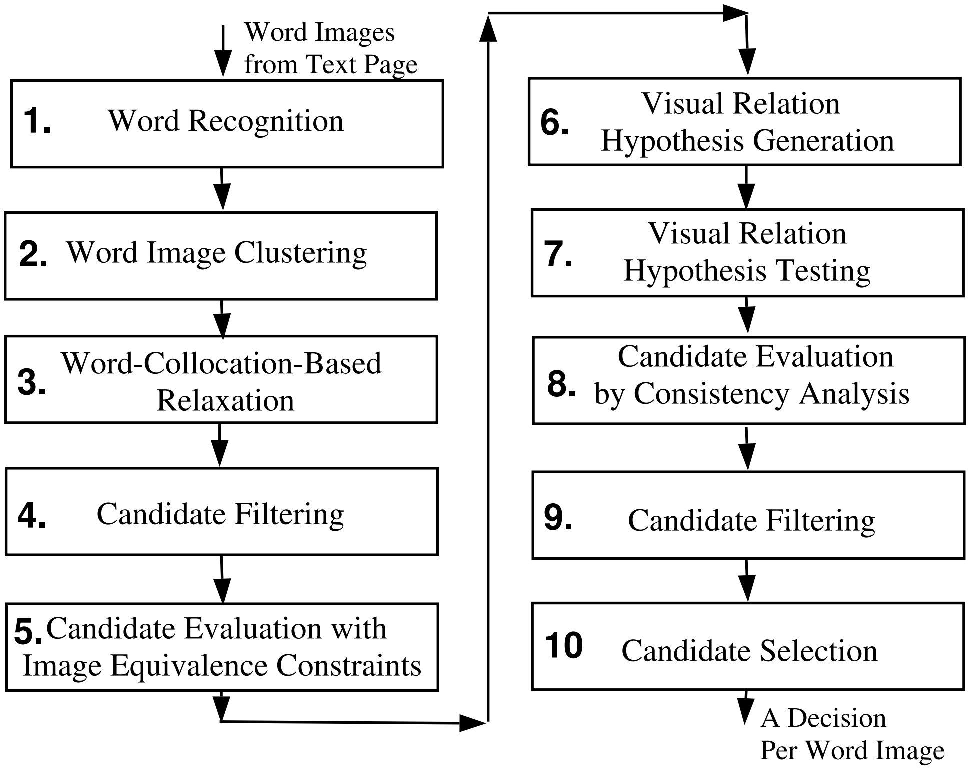


Figure 9.5: Outline of the proposed procedure

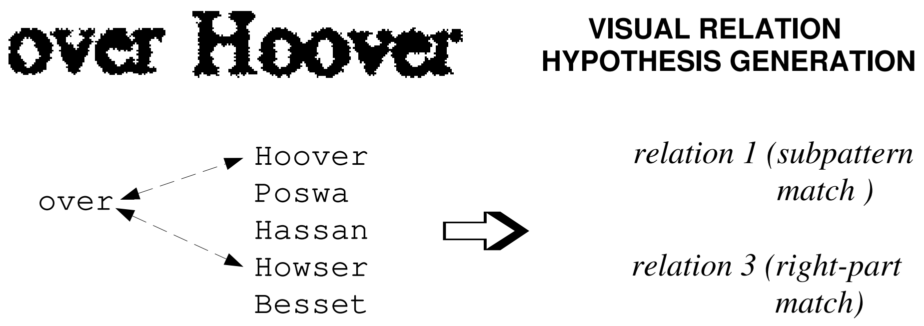


Figure 9.6: Visual relation hypothesis generation

9.3.2 Experimental Results

One article from the Brown Corpus was used as the test sample. The article is "*G02*," which has 2,272 words. Each punctuation mark is counted as an individual word. The article was printed in 12pt *times-roman* font by using *ditroff* on five text pages. Those text pages were scanned as binary images at the resolution of 300ppi. Coordinates of word bounding boxes were provided by Caere's AnyFont OCR Toolkit on these scanned images. To simulate degraded document pages, the University of Washington's Document Degradation Model (DDM) was applied to those scanned images. The parameters of the DDM are set at (820, 0.000, 1.0, 1.0, 1.0, 1.0, 3).

Figure 9.7 shows a small portion of a text page after adding noise using the DDM model. The performance of Caere's OCR package on those degraded pages is poor (see Figure 9.8), although its performance on the original scanned pages is almost perfect. These degraded text pages were used in this experiment.

After extracting word images from those degraded text pages by using the coordinates of their word bounding-boxes, a word recognition algorithm based on the word-shape-analysis method is applied to each word image to generate its top10 word candidates. The dictionary used here has more than 90,000 different entries. In this experiment, if the top10 does not include the correct choice, the last candidate will be replaced by the correct choice. We assume that each punctuation mark has been recognized correctly so that their candidate lists have a single candidate. The total number of such punctuation marks is 241. For each of the other words, there is a candidate list with 10 word candidates ranked by their confidence scores, which are the probabilities of being correct. There are several repeated entries in the dictionary we used. Therefore, in the word recognition result, for some word images there are less than 10 different word candidates in their top10 lists. That is, there are 54 word images with 9 word candidates. For the 2,272 word images, the average size of a candidate list is 9.0. Table 9.4 shows the correct rates of the top 1 to 10 and accumulated correct rates of the top1 to top10 after word recognition.

In the past , the duties of the state , as Sir Henry Maine noted long ago ,
were only two in number : internal order and external security . By prevailing
over other claimants for the loyalties of men , the nation-state maintained
an adequate measure of certainty and order within its territorial borders .
Outside those limits it asserted , as against other states , a position
of sovereign equality , and , as against the inferior peoples of the non-Western
world , a position of dominance . It became the sole subject of international
law (a term which , it is pertinent to remember , was coined by Bentham) ,
a body of legal principle which by and large was made up of what Western
nations could do in the world arena . (that corpus of law was a reflection
of the power system in existence during the eighteenth and nineteenth centuries .

Figure 9.7: First 12 text lines extracted from the first page

---

```
En the past, the duties of the sage, as Sir Henry Maine noted long ago .
true ouly two in number: internal Stir and external seax y . By prevailing
over otherer eldmants for the loyalties of tnen, the nation-stllte maintained
an adequate nxeasse of Key and aver within its territorial bards .
Outside Bose limits it a, as against other states, s position
of sovereign equality, and, as against the inferior peoples of the non-Western
world, a position of dominance . It became the sole subject of international
law ( a teem which, it is pertinent to retnembs, was Cued by BenXn ),
a body of legal principle which by and large was made up of what Western
nations could do in the world arena . ( that corpus of law was a rcflecsia
of the power sysbtn in exiswe dwing the eighteenth and nineteentht centuries
```

Figure 9.8: A commercial OCR's output for the example text block

---

After the step of word image clustering, there are a total of 1,074 clusters. Among them, 282 clusters have more than one image instance. There are 1,480 word images which are grouped in those large clusters. The largest cluster has 169 image instances. The distribution of the sizes of clusters is shown in Table 9.5.

The next step is to apply the word-collocation-based relaxation algorithm to word candidate lists. The number of iterations in relaxation is 10. The correct rate of the first choice

	top1	top2	top3	top4	top5	top6	top7	top8	top9	top10
correct %	60.1%	9.7%	5.4%	4.4%	3.7%	3.4%	4.0%	3.3%	2.6%	3.5%
acc. correct %	60.1%	69.9%	75.3%	79.6%	83.4%	86.7%	90.7%	94.0%	96.5%	100.0%

Table 9.4: Accuracy of word recognition results (Number of words = 2272; Average neighborhood size = 9.0)

Frequency	Cluster Size	Frequency	Cluster Size	Frequency	Cluster Size	Frequency	Cluster Size
1	169	1	21	5	10	28	4
1	95	1	17	4	9	53	3
1	50	2	14	4	8	132	2
1	45	1	13	8	7	792	1
1	44	1	12	11	6		
1	29	6	11	18	5		

Table 9.5: Distribution of the size of image clusters

is 85.2% after this step (see Table 9.6).

	top1	top2	top3	top4	top5	top6	top7	top8	top9	top10
correct %	85.2%	6.5%	2.3%	1.2%	1.4%	1.0%	0.8%	0.7%	0.5%	0.5%
acc. correct %	85.2%	91.7%	94.0%	95.2%	96.6%	97.6%	98.4%	99.0%	99.5%	100.0%

Table 9.6: Accuracy of relaxation (Number of words = 2272; Average neighborhood size = 9.0)

After word-collocation-based relaxation, the step of candidate filtering was conducted to remove those candidates with low confidence scores. In this experiment, the same method described in Chapter 7 was used. The candidates at the bottom of a candidate list were removed if their cumulative confidence is less than 0.05. After this step, the average size of the candidate lists was reduced to 3.7 from the original 9.0. About 45% of the word images have only one candidate left. The accuracy of the remaining choice is 99.7%. Tables 9.7 and 9.8 show the accuracy of candidate lists and the distribution of neighborhood size.

	top1	top2	top3	top4	top5	top6	top7	top8	top9	top10
correct %	85.2%	6.5%	2.1%	1.1%	1.3%	0.8%	0.6%	0.5%	0.4%	0.4%
acc. correct %	85.2%	91.7%	93.8%	94.9%	96.2%	96.9%	97.5%	98.0%	98.4%	98.8%

Table 9.7: Accuracy after candidate filtering (Number of words = 2272; Average neighborhood size = 3.7)

The next step is to use visual equivalence constraints to adjust confidence scores of word candidates. For those word images which were in the same cluster, the decisions have to agree with each other. The weighted voting procedure described in the previous experiment was used. For each cluster with multiple images, a candidate list is created by taking the

Neighborhood Size	Percentage		Accuracy
1	45.5%	(1030/2272)	99.7%
2	12.9%	(286/2272)	98.0%
3	7.0%	(151/2272)	95.0%
4	3.3%	(73/2272)	97.3%
5	4.7%	(101/2272)	95.3%
6	1.6%	(35/2272)	97.2%
7	0.9%	(19/2272)	95.0%
8	1.4%	(32/2272)	100.0%
9	11.6%	(263/2272)	99.6%
10	11.3%	(256/2272)	100.0%

Table 9.8: Distribution of neighborhood size after candidate filtering (Average neighborhood size = 3.7)

union of candidates of images in the cluster. The confidence score of each candidate for the cluster is the summation of confidences of the same candidate from candidate lists of different words in the cluster. After computing the candidate list for each cluster, each image in the cluster inherits this candidate list as its own. The accuracy after the step is shown in Table 9.9.

	top1	top2	top3	top4	top5	top6	top7	top8	top9	top10
correct %	87.8%	5.0%	1.8%	1.1%	0.9%	0.6%	0.5%	0.4%	0.3%	0.4%
acc. correct %	87.8%	92.8%	94.6%	95.7%	96.7%	97.3%	97.8%	98.2%	98.5%	98.8%

Table 9.9: Accuracy after using image equivalence constraints (Number of words = 2272; Average neighborhood size = 3.7)

In the step of visual inter-word relation hypothesis generation, symbolic inter-word relations between candidates from different word images are calculated. If a symbolic inter-word relation between candidates from two word images is found, a hypothesis about the same type of visual inter-word relation for those two images will be generated. In this experiment, we did not try to generate all possible hypotheses between every word image pair because there would be too many candidate pairs to be checked. To be efficient, we only generated these relation hypotheses for image pairs in which one image has just one candidate. And for each image cluster, only one of the images was selected to represent the images in the cluster for such analysis. There were 16,826 relation hypotheses generated that involved 694 image clusters. Table 9.10 lists the results of the step of hypothesis generation.

Relation type		Number of hypotheses generated	Number of hypotheses tested	Time on SPARCStation (seconds)
0	(equivalence)	243	83	47
1	(subpattern)	7009	2647	1,738
2	(same-left-part)	2162	220	1,103
3	(same-right-part)	4817	453	1,902
4	(head-to-tail-match)	2595	184	2,694
total		16826	3587	7,484

Table 9.10: Number of relation hypotheses generated by the symbolic level analysis

After generating visual relation hypotheses by symbolic analysis, those hypotheses were tested by image matching. Among 16,826 hypotheses, 3,587 hypotheses were confirmed and the rest were rejected(see Figure 9.10). Those confirmed hypotheses are visual inter-word relations which are detected without exhaustive comparison at the image level. Each visual relation has a confidence score which is the similarity measurement from image matching.

The next step is to re-evaluate the confidence scores of the word candidates based on consistency analysis at the image and string level. When two word images hold a visual inter-word relation, the confidence scores of those candidates which keep the same type of relation at the string level will be increased; the confidence scores of those candidates which do not keep the same type of relation at the string level will be decreased. After this step, the correct rate of the first choice was improved to 90.2% (see Table 9.10).

	top1	top2	top3	top4	top5	top6	top7	top8	top9	top10
correct %	90.2%	4.2%	1.1%	1.0%	0.8%	0.5%	0.2%	0.3%	0.3%	0.3%
acc. correct %	90.2%	94.4%	95.5%	96.5%	97.2%	97.8%	97.9%	98.2%	98.6%	98.8%

Table 9.11: Accuracy after consistency analysis (Number of words = 2272; Average neighborhood size = 3.7)

After further removal of those low confidence word candidates, the average size of the candidate lists drops to 2.8 from 3.7. There are now 50.7% of words with only one candidate, and the correct rate of those candidates is as high as 99.6% (see Tables 9.12 and 9.13).

	top1	top2	top3	top4	top5	top6	top7	top8	top9	top10
correct %	90.2%	4.2%	1.1%	1.0%	0.7%	0.5%	0.2%	0.3%	0.3%	0.3%
acc. correct %	90.2%	94.4%	95.4%	96.4%	97.1%	97.6%	97.8%	98.1%	98.4%	98.6%

Table 9.12: Accuracy after further candidate filtering (Number of words = 2272; Average neighborhood size = 2.8)

In summary, by applying the procedure, the accuracy of the first choice is improved from 60% to about 90% and the average size of the candidate lists is reduced from 9.0 to 2.8. For about 50.7% of word images, the size of the candidate list is reduced to a single word and its accuracy is 99.6%.

Neighborhood Size	Percentage		Accuaracy
1	50.7%	(1152/2272)	99.6%
2	15.9%	(360/2272)	97.5%
3	7.3%	(165/2272)	93.3%
4	5.4%	(123/2272)	98.3%
5	3.3%	(75/2272)	97.3%
6	2.6%	(60/2272)	98.3%
7	3.9%	(89/2272)	98.8%
8	4.4%	(100/2272)	100.0%
9	3.7%	(84/2272)	100.0%
10	2.8%	(64/2272)	100.0%

Table 9.13: Distribution of neighborhood size after further candidate reduction (Average neighborhood size = 3.7)

9.3.3 Discussion

In this experiment, we proposed a multiple-step procedure to integrate visual and linguistic constraints for word candidate selection. Word collocation constraints are effective but their usefulness differs depending on the position of the word in the text and the configuration of its candidate list. For many word images, decision candidates can be determined with very high accuracy by the word-collocation-based relaxation algorithm and confidence-based candidate reduction. Visual inter-word relations provide a way to link word images inside a text page and across text pages of the same article. They can be used as global constraints for candidate selection. For those word images for which the decision candidates are difficult to determine by word collocation, candidate selection can be made much easier by exploiting visual inter-word relations.

In this experiment, we also explored an efficient approach to computing visual inter-word relations: the *symbolically-driven* approach. Since it is costly to compute all possible visual inter-word relations by taking the straightforward *image-driven* approach, in which image matching operations are conducted to exhaustively test whether or not any two word images hold a certain relation, a *hypothesis-generation-and-test* method was proposed here. A hypothesis about a visual inter-word relation was generated by string comparison between the word candidates from two images, then it was tested by image matching. In this method, visual analysis, symbolic analysis and linguistic analysis are interleaved.

9.4 Conclusions

Integration of image processing and natural language processing is a new area of interest in document analysis. Word candidate selection is a problem we are faced with in degraded text recognition. Statistical language models and lattice parsers were proposed for this problem. Visual inter-word constraints in a text page can be used with linguistic knowledge sources to facilitate candidate selection. Preliminary experimental results show that the performance of candidate selection is improved significantly by combining the use of visual and linguistic constraints.

Chapter 10

Chinese/Japanese Recognition Using Visual and Linguistic Context

In this thesis, a framework was proposed to exploit visual and linguistic contextual information for document recognition. Although we concentrated our discussion on English document recognition, the methodology can be adapted to other alphabetic languages such as French, German and Spanish.

For ideographic languages, such as Chinese and Japanese, the applicability of the methodology is not obvious because of the intrinsic difference between alphabetic and ideographic language systems. There are a large number of ideographs– Chinese characters (which are called as "*Hanzi*" in Chinese and "*Kanji*" in Japanese). In this chapter, we discuss briefly the uses of visual and symbolic/linguistic context for Chinese/Japanese OCR postprocessing.

10.1 Visual Similarity Analysis of Chinese Characters

The Chinese and Japanese character sets are large since both utilize thousands of Chinese characters [96]. Traditionally, a Chinese or Japanese OCR has to represent each character category individually as one or more prototypes, or as a structural description which is a composition of manually derived components such as radicals, so that an input character image can be recognized by classifying it into a class which corresponds to a character category.

A Chinese character is usually complex in structure. It can be composed of several elements, such as radicals and other simple components, based on lexicographical principles. Although there are more than ten thousand Chinese characters in use, lexicographical researchers estimate that there are only about 800 character elements and no more than 100 such elements are frequently used [168].

Many Chinese characters share a similar character element at the lexicographical level. If they are printed in the same font, their images will be partially similar. Figure 10.1 shows three sets of characters which have the same left, right or bottom part respectively. Figure 10.2 is an image fragment extracted from a scanned Japanese document page. There are eight character images. Among them, characters *1, 3* and *4* have a similar left part; characters *3* and *7* have a similar right part. Generally, similarities at the lexicographical level and the visual level are highly consistent. As we observed, if character categories share the same lexicographical element in a region, their image instances are usually similar visually in that region.

There are previous studies on Chinese character recognition using radical-based partial matching [1, 168]. Although radicals are well-defined lexicographically, it is time-consuming to manually locate all possible character radicals and describe the structure of each Chinese character according to the radical set. The goal of our work is to conduct a visual similarity analysis among Chinese characters at the image level. Given a set of Chinese character images, we try to automatically determine which character images have some components in common and what those components are. If the images are font samples, lexicographical

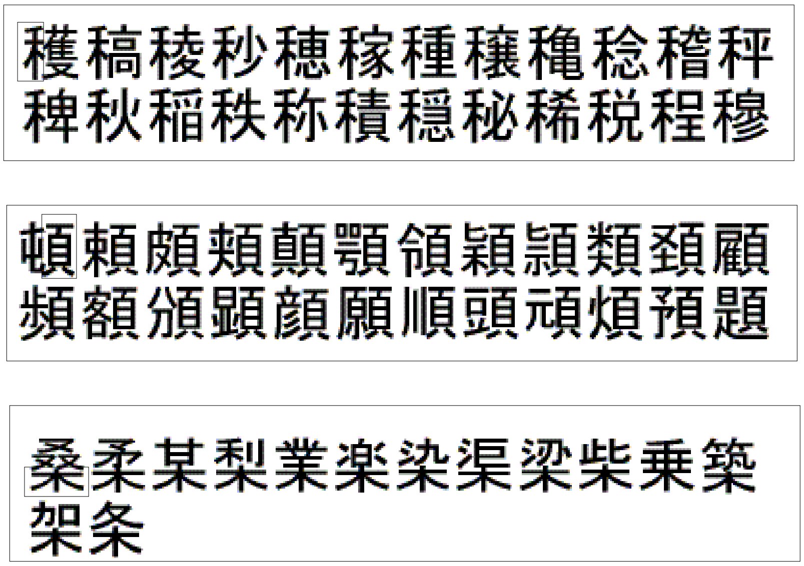


Figure 10.1: Chinese characters sharing visual similarity

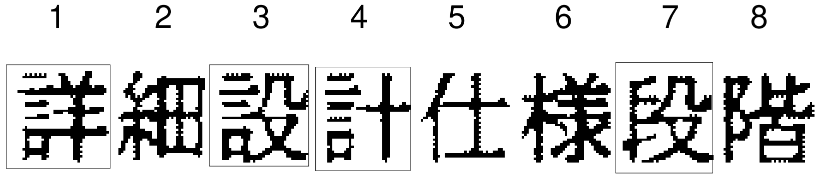


Figure 10.2: Characters sharing visual similarity in a phrase from a Japanese document page

similarities among categories can be further derived based on visual similarities among samples which represent different categories. If images are from a document page, visual similarity between two character images can be used as a constraint in the stage of OCR postprocessing to interpret those related images consistently.

In the rest of this section, we will first describe how to conduct visual similarity analysis, then discuss how to utilize inter-character relations in OCR postprocessing.

10.1.1 Methodology

Although visual similarity analysis can be accomplished with pixel-to-pixel matching, we choose feature-based matching to conduct the analysis because it is more efficient than pixel-to-pixel matching.

To represent the stroke structure of a Chinese character, its feature vector usually has many dimensions. For some types of structural features such as *local stroke direction* (*LSD*) [2, 133], geometric information is retained as in the original image. We use *LSD* features for our experiments. When given a character image, its LSD feature vector can be computed as follows (see Figure 10.3 for an example in detail): first, for each black pixel, compute its directional run-length for each of four directions and normalize it as a ratio to the total run-length in all directions; second, partition the image into $n \times n$ areas and compute the directional run-length of each area as an average of the pixels in the area. Here, we choose n equal to 8. Therefore, the size of the LSD feature vector is $4 \times 8 \times 8 = 256$. In Figure 10.3 (c), the values in the LSD feature vector are scaled to integers at the range of 0 and 255 so that they can be visualized in a gray-scale image.

The difference between two feature vectors can be measured simply by their city-block distance. To make the distance independent of the size of the feature vector, it can be normalized, dividing by the size of the feature vector. If the total distance is less than

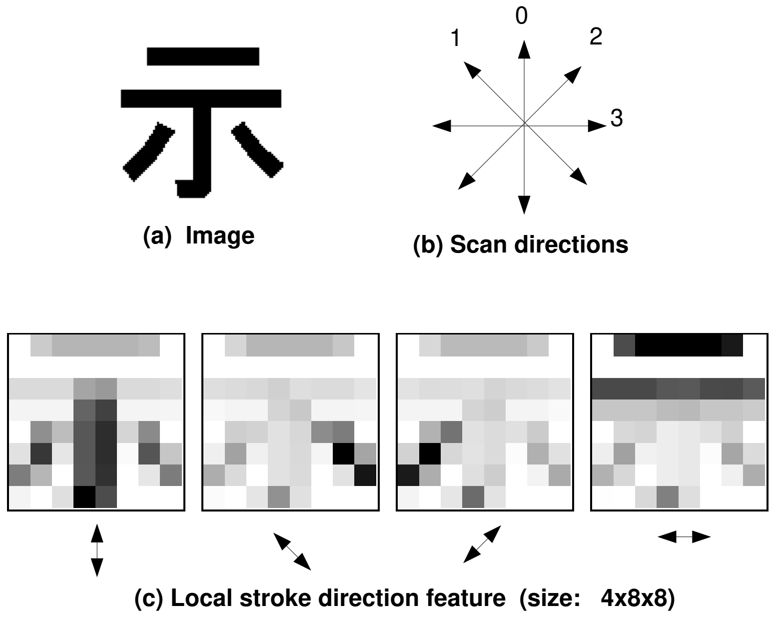
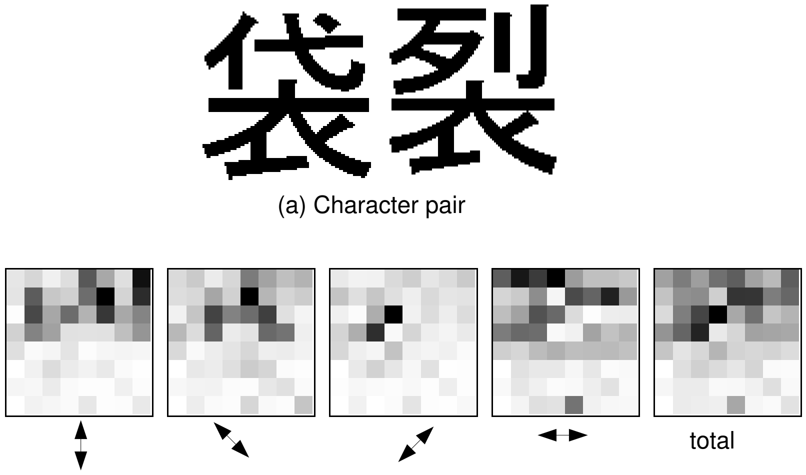


Figure 10.3: A Chinese character image and its *local stroke direction* (LSD) feature vector

a given threshold, we can say they are similar globally. If the total distance is large, by analyzing the distribution of distance at each dimension, we can determine whether they are partially similar. For example, in Figure 10.4 (a), two Chinese characters are shown. Figure 10.4 (b) shows the distance distribution of their difference. We can determine that distance scores from the bottom part are significantly smaller than that from the top part. The result suggests to us that the bottom part of those two images are very similar.

The similar region of two LSD features can be approximately represented by a mask. A mask is defined as a $n \times n$ matrix with binary values, where "1" means "*Set-On*" and "0" means "*Set-Off*." The mask in Figure 10.4 (c), with "1"s in the last 4 rows, is derived from Figure 10.4 (b) to record that the two images are similar at their bottom part.

As described above, given two feature vectors, a mask can be derived to indicate the region in which they match well. Similarly, given two feature vectors and a mask, we can check whether the features can match well in the region represented by the mask. A pre-



(b) Difference in LSD feature

```
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1
```

(c) Mask obtained

Figure 10.4: Two Chinese images and their difference in LSD

defined mask X can be treated as a relation. $X(f_1, f_2)$ denotes that f_1 and f_2 are similar in the "*Set-On*" region of the mask X.

In our preliminary experiments, twenty-nine types of masks are considered to represent different regions (see their definitions and denotations in Figure 10.5). Each mask is a possible relation between images. More complex masks can be defined similarly.

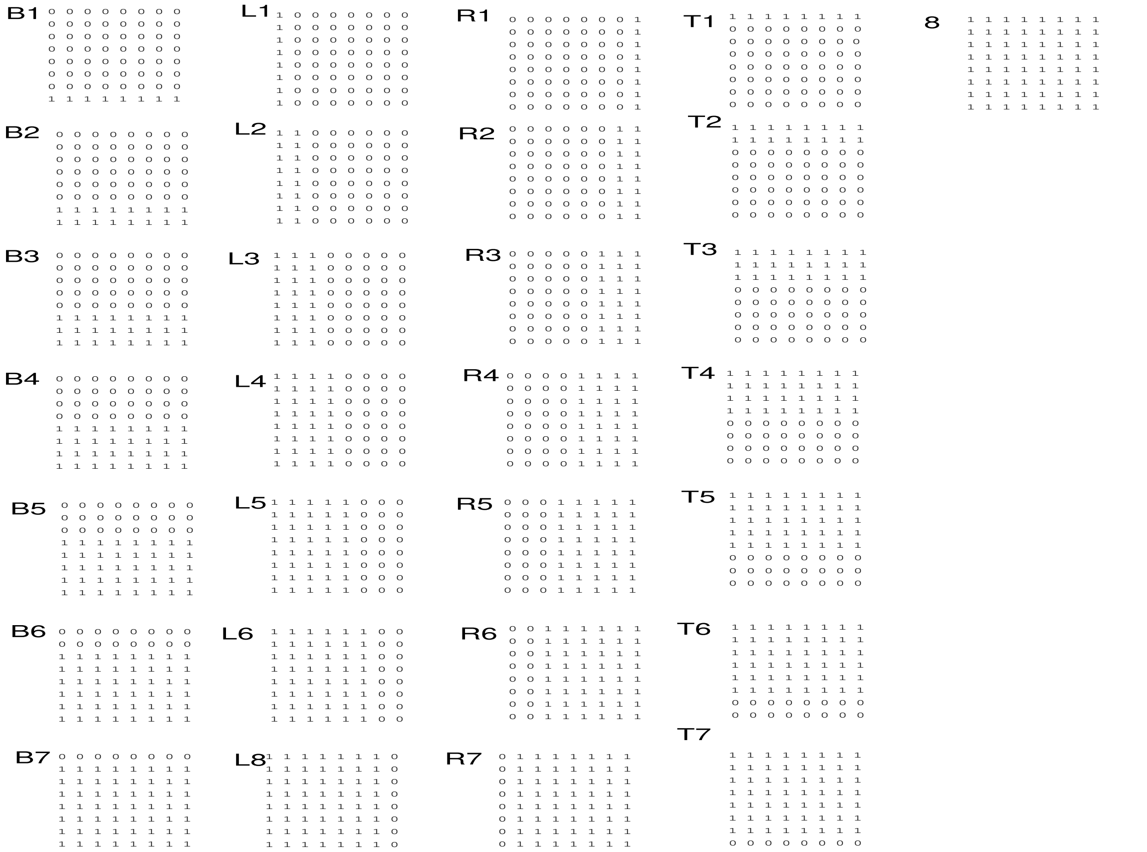


Figure 10.5: Twenty nine types of masks

By applying feature-based visual similarity analysis to character images from a specific typeface, we can compute a similarity matrix which records all possible visual inter-category relations. There are 2,965 Kanji characters in the first level of the JIS code. Using a 96x96 *gothic* font, 50,456 inter-category relations were computed. 25 of the 29 masks defined in

Figure 10.5 are used as masks (the 4 masks not in use are $B1$, $L1$, $R1$ and $T1$, because their "*Set-On*" regions are too small). Figure 10.6 shows a portion of the similarity matrix created according to those relations. Each row represents a Kanji category; so does each column. If there is a relation entry $X(i,j)$ at (i,j) in the matrix, it means that categories i and j hold the relation X. The matrix allows several different relations between two categories. For example, there are $B5$, $L4$ and $R2$ for categories 3028 and $3B4F$ in JIS code. It is obvious that the matrix is symmetrical. The matrix is sparse because most of its elements are empty. By carefully analyzing the relations in the matrix, a list of radicals can be compiled.

JIS Code		3021	3022	3023	3024	3025	3026	3027	3028	324F	332C	357C	3F6A	4F50	4F51	4F52	4F53
	Kanji	亜	唖	娃	阿	哀	愛	挨	姶	河	階	亭	衰	椀	湾	碗	腕
3021	亜																
3022	唖																
3023	娃								L3								
3024	阿									R3	L3						
3025	哀											T3	B2				
3026	愛																
3027	挨																
3028	姶			L3													
3076	咽		L2														
325E	嘩		L2														
333F	蛙			R4													
373E	珪			R5													
374B	桂			R6													
3B4F	始								B5 L4 R2								
3D58	舜						T3										
436F	衷					8											
483A	蛤								R6								
4929	菱						B3										
4F50	椀															R5	R5
4F51	湾																
4F52	碗													R5			R5
4F53	腕													R5		R5	

Figure 10.6: Similarity matrix built on *gothic* font

In principle, relations among categories are transitive: if $X(i,j)$ and $X(j,k)$ are found in the similarity matrix, $X(i,k)$ should also be found the the matrix. However, the matrix may not have this property because the relation entries are computed by feature matching under a specific threshold. Sometimes, the difference between images A and C can be larger than a threshold although the differences between images A and B and between images B and C are smaller than the threshold. To alleviate the effect of thresholding, the matrix can be augmented to be transitive: when $X(i,j)$ and $X(j,k)$ are found in the similarity matrix, $X(i,k)$ will be added to the matrix if $X(i,k)$ has not been found.

Based on the augmented similarity matrix, categories can be further clustered according to relations. For a relation $X(i,j)$, a cluster C can be computed. By initializing C as $\{i,j\}$, C will be expanded iteratively. If category n is not in C, but there exists an m in C so that $X(m,n)$ is a relation, then n will be added into C. The process terminates when C can not be expanded any further. For the cluster C, the label is the mask X, which indicates the similar part shared by the categories in the cluster. There are 2,407 clusters created based on the matrix from the *gothic* font and the average number of categories in a cluster is 5.2. Twelve such clusters are listed in Figure 10.7. For most clusters, the common region corresponds to a lexicographical element (see in Figure 10.7 (a)-(f) and (h)-(i)). For some clusters, the common region has no meaning at the lexicographical level (see Figure 10.7 (j)-(k)). There are also some clusters in which the regions indicated by their masks contain different radicals because those radicals are visually so similar that the LSD feature is not sensitive to the difference among them under the given threshold (Figure 10.7 (l) is an example of such clusters).

10.1.2 OCR Postprocessing Using Visual Similarity Analysis

If a Chinese or Japanese text page is degraded, it might be difficult for an OCR to recognize each character image accurately. Instead, the OCR may generate a candidate list for each Chinese character (see Figure 10.8 for the top5 candidate lists of three Chinese characters) [65]. The candidates in a candidate list are ranked by their confidence scores

a. mask=L3 謂詠謁課該諌記誼議詰許謹訓計詣訣謙諺誇語
誤護講詐讃詞詩試誌諮識謝諸訟証詔詳譲診訊
諏誠請設説詮訴詑託諾誰誕談註調諜訂諦討読
謎認誹謬評譜訪謀訳諭誘謡諒論話詫

b. mask=R2 穎頴額顎顔願傾頚顕顧順須頗碩頂顛頭嚬頓煩
頒瀕頻頬預頼領類

c. mask=L3 鯵鮎鰯鰻鰍鰹鯨鯉鮭鯖鮫鮮鯛鱈鰭鮒鮪鱒鱗鰐

d. mask=R4 鵜鴎鴨鶏鴻鴿鴫鶴鴇鳩鵬鵡

e. mask=T4 雲霞雫需震雪霜電雰霧雷零霊露

f. mask=R4 劾勤功効劫助勅動勃幼励肋

g. mask=B2 哀袈裟襲衰製装袋表褒蓑裏裂

h. mask=B4 袈裟裳装袋裏裂

i. mask=R6 儒濡

j. mask=B2 旭危池氾泡厄

k. mask=T4 吉去圭市志支寺赤走夫末未

l. mask=L2 挨穐握梓扱按椅移稲榎掩援押横桶穏科稼拐械
概柿拡攪格核穫掛樫橿梶括椛樺株栢換柑桓棺
揮机棋機稀技掬桔橘杵朽拒挟橋極桐櫛掘掲携
桂稽桁捲検権枯梧檎抗拘控杭校梗構稿拷根梱
挫採材榊搾柵桜摎撮擦札捌桟指枝私持捨杓種

Figure 10.7: Examples of category clusters

which were provided by the classifier to indicate how likely the choice is to be correct. Due to the presence of noise, on occasion the correct choice does not earn the highest confidence score, although it usually appears inside the candidate list. In this case, a postprocessing stage is required to select a decision character from the candidate list for each character image.

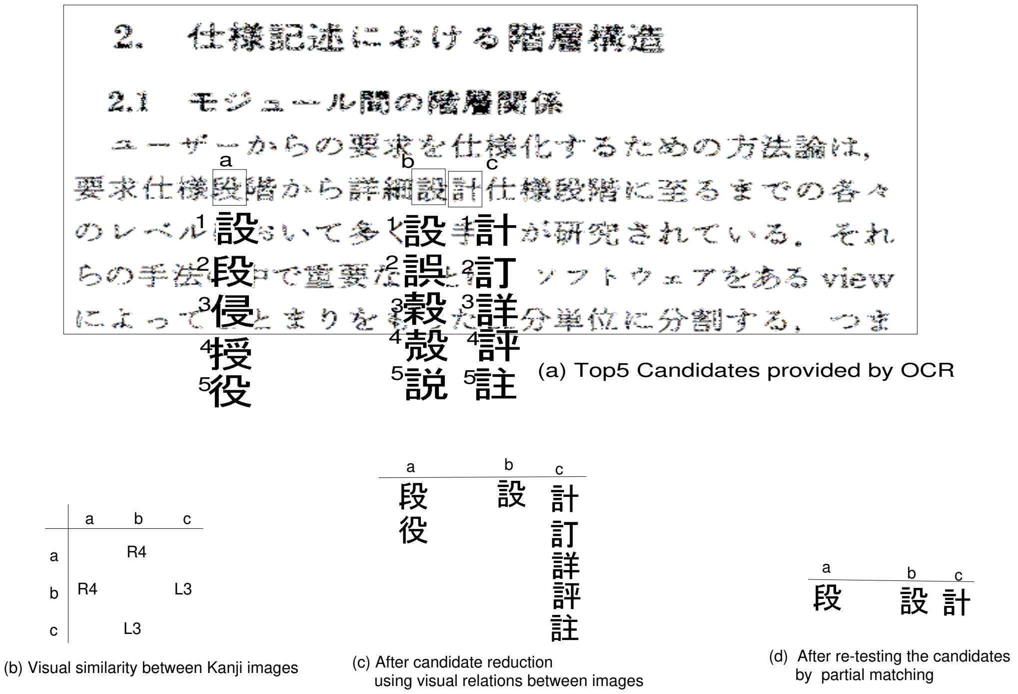


Figure 10.8: Kanji candidate selection using visual similarity

For a character image with several candidates, visual similarity between character images and categories can provide useful information to select a proper choice. Two methods are proposed here.

The first method is to use visual relations between character images to reduce the size of a candidate list if possible. Given a text page, character images can be extracted after the stage of character segmentation. Visual inter-character relations between character images can be computed. For example, for images *a*, *b* and *c* in Figure 10.8 (a), two relations,

$R4(a,b)$ and $L3(b,c)$, are derived (see Figure 10.8 (b)). Here, $R4(a,b)$ means that images a and b have a similar right part; $L3(b,c)$ means images b and c have a similar left part. Correct choices for images a, b and c should keep those relations in the similarity matrix. Let's consider the candidates of image b to see how the candidate list can be reduced by visual similarity analysis. Because image b is similar to image c in its left part, its candidates *3* and *4* can be removed because neither of them can keep the relation with any candidates of image c. Similarly, by considering the relation between images a and b, candidates *2* and *5* of image b can be removed and there will be only one candidate remaining for image b. In the same way, several candidates of image a can be rejected. It has to be pointed out that candidate *1* of image a was deleted because of the fact that images a and b are different on left part. The result of this analysis is shown in Figure 10.8 (c).

The second method is to utilize possible visual relations among candidates for a character image and to re-test them by partial matching. Consider the candidates of image c in Figure 10.8 (c). By checking the augmented similarity matrix, we know that all candidates have the same left part. To distinguish those candidates, we have to focus on the information from their right parts. By matching the right part of the image with right parts of prototypes for categories that those candidates belong to, those candidates can be re-ranked by their new confidence scores which are based on the information from the right part. The first candidate will be selected as the decision for the image. For the example above, the final result is shown in Figure 10.8 (d).

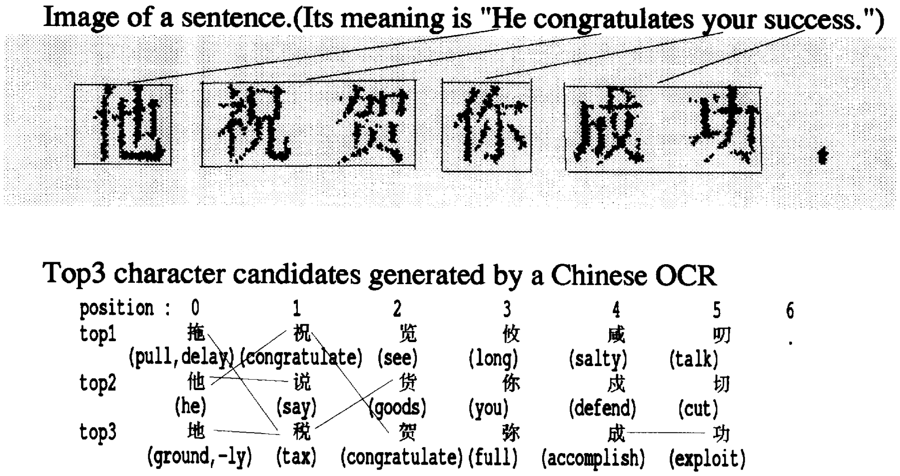
10.2 Integration of Linguistic and Visual Contextual Analysis for Chinese OCR Postprocessing

A Chinese text is comprised of a sequence of Chinese characters (i.e., Hanzi). The number of frequently used Hanzi characters is more than three thousand. A Chinese word may consist of one or several Chinese characters. However, there exist no visible word boundaries inside Chinese texts.

Linguistic context can be applied to choose decisions for those character images which have competitive candidates generated by a Chinese character classifier. Like that for English text recognition, linguistic knowledge sources useful here include word dictionaries, statistical language models, language syntax and semantics[24, 43, 78].

Character recognition results for a Chinese sentence are shown in Figure 10.9 (a). The translation of the sentence is "*He congratulates your success*" There are six Chinese characters in the sentence. For each character, three character candidates are provided by a classifier. The candidates for a character image are visually so similar that they are easy to be confused by a character classifier.

The relaxation algorithm and lattice parser described in Chapters 7 and 8 have been adapted for Chinese text recognition. A context-free grammar for Mandarin Chinese has been written. It has more than 400 rules. A large Chinese lexicon has been edited. It has about 34,000 words which are assigned POS tags. Statistical data have been collected from two large Chinese corpora, the PH corpus and the HXWZ corpus. The PH corpus includes about 8000 news articles with about 4 million characters [45]. The HXWZ corpus is collected from an online Chinese journal "Hua Xia Wen Zhai" (Chinese News Digest) which publishes weekly, since April 1991. Each issue of HXWZ has about 20,000 characters. Statistical data include: character frequency, word frequency, character collocation and word collocation. With those knowledge sources, the relaxation algorithm and lattice parser can work for both OCR postprocessing and PinYin-to-text translation. For the sample sentence in Figure 10.9 (a), character collocation data can be applied to reduce the size of candidate lists (see Figure 10.9 (b)). The best parse tree generated by the lattice parser is shown in Figure 10.9 (c). The candidates finally selected are shown in Figure 10.9 (d).



(a) Top3 character candidates generated by a Chinese OCR for a sentence

position	0	1	2	3	4	5	6
top1		祝		攸			.
top2	他	说	货	你	戍	切	
top3	地		贺		成	功	

(b) Reduced candidate list after applying character-collocation constraints

```
( S
  ( S-BODY
    ( NP
      ( PPP  他  ) )
    ( VP
      ( VB  祝  贺  )
      ( NP
        ( S-BODY
          ( NP
            ( PPP  你  ) )
          ( VP
            ( VB  成  功  ) ) ) ) ) ) )
  ( END_MARK  .  ) )
```

(c) The most preferred parse tree generated by a lattice parser

position	0	1	2	3	4	5
top1		祝				
top2	他			你		
top3			贺		成	功

(d) The character candidates selected after parsing

Figure 10.9: Chinese recognition using linguistic context. In (a), the lines between character candidates indicate the strong collocation between characters.

As demonstrated for English text recognition, visual inter-character relations can be used as global constraints, together with linguistic constraints, for Chinese recognition. Figure 10.10 gives an example of the integration of visual and linguistic context for Chinese OCR postprocessing. For character images 1 and 7, three character candidates are provided by the OCR. By using linguistic knowledge, we know that there are two possible interpretations for the sentence (see Figure 10.10 (b)). According to those interpretations at the sentence level, for image 1, the second candidate can be selected with high confidence; for image 7, no decision can be made because the first and the second candidates both make sense in context. But, if we consider the visual equivalence between images 1 and 7, we can make a decision for image 7. Because both images have to be recognized as the same character, the second candidate will be selected for image 7 just as for image 1 (see Figure 10.10 (c). In the same way, the other inter-character relations discussed in the first section of this chapter can be used together with linguistic constraints.

10.3 Conclusions

In the chapter, we extended the approach of contextual analysis pursued for English text recognition to Chinese and Japanese text recognition. The extension is not trivial because of the intrinsic difference between alphabetic language systems and ideographic language systems. We presented an approach to automatically analyze visual similarity among Chinese characters based on a feature description. By training on font images, visual relations between categories can be compiled. Similarly, visual relations between character images from a text page can be computed. OCR postprocessing methods were proposed to choose decision characters from candidate lists by using visual relations between images together with a similarity matrix which was pre-compiled to record possible visual similarity among categories. After discussing the use of linguistic context for Chinese OCR postprocessing, we further demonstrated that visual and linguistic context can be exploited together to

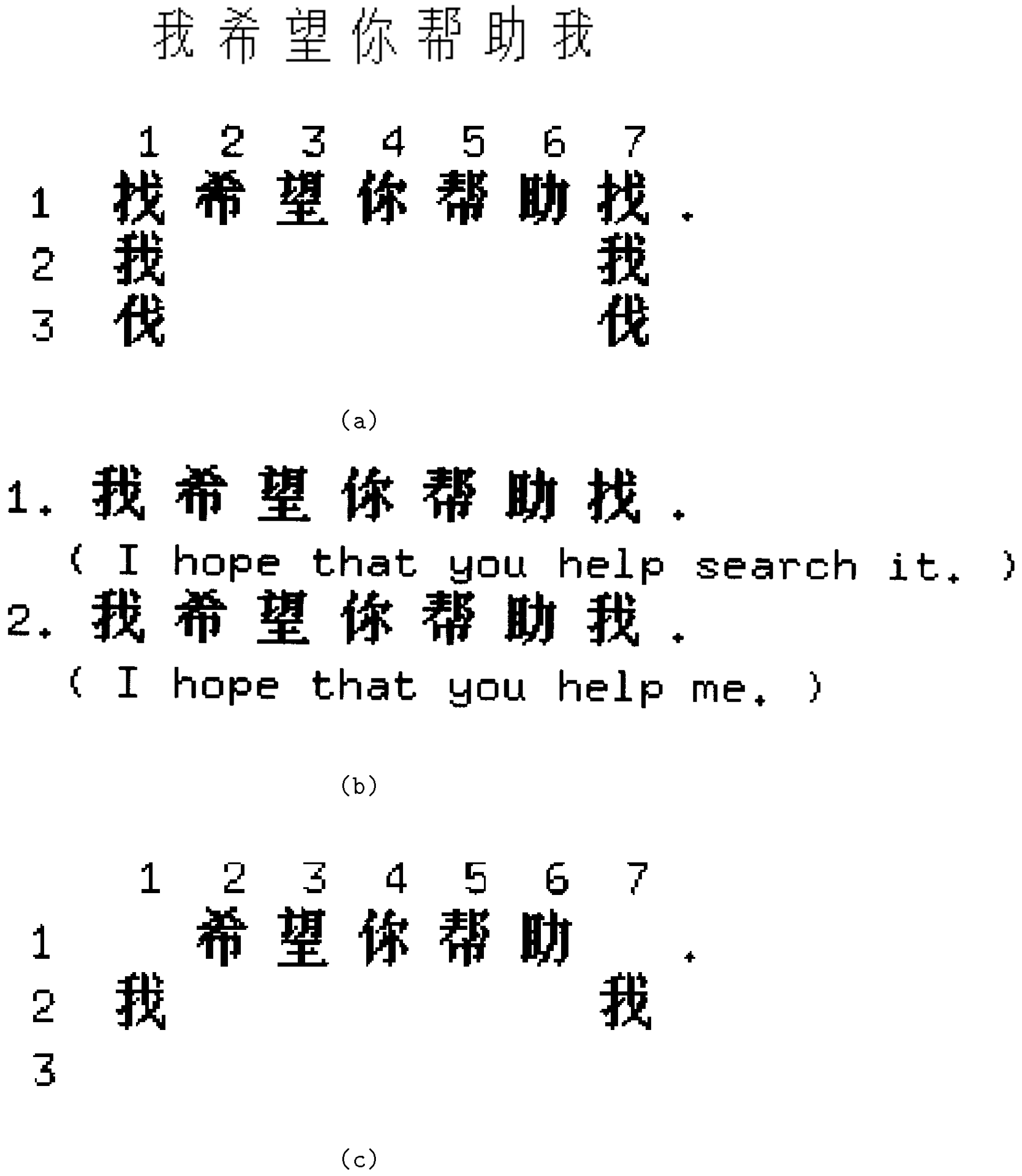


Figure 10.10: Chinese recognition using linguistic and visual context. Character recognition results are shown in (a). There are seven characters in the sentence. for characters 1 and 7, there are three alternatives. By image matching, we know that character images 1 and 7 are visually equivalent. Using linguistic knowledge, two possible interpretations for the sentence are listed in (b). By integrating visual and linguistic constraints, (c) shows that the second candidate for characters 1 and 7 can be selected correctly.

achieve better performance on OCR postprocessing.

Chapter 11

Conclusions

In this work, we explored the use of contextual information for degraded text recognition. A computational framework was proposed to exploit visual and linguistic context for passage level postprocessing. Experiments on components of the framework have been conducted. The following are the main contributions of this research:

1. Techniques for OCR postprocessing are classified according to the level of contextual constraints exploited in those methods. In addition to traditional word-level postprocessing, passage-level postprocessing is identified as a necessary stage to improve recognition performance for degraded text. Visual context, as well as symbolic and linguistic context, can be used in passage-level postprocessing.

2. A new set of visual inter-word relations was defined [60]. Those visual relations reflect typographical constraints inside a document page and are tolerant to uniform noise. In a document image written in normal English, many word images have visual inter-word relationships to each other [61].

3. The principle of consistency between visual inter-word relations and symbolic inter-word relations was described. Therefore, visual inter-word relations can be used as contextual constraints to interpret related word images systematically [58].

4. A new algorithm was designed to exploit visual inter-word relations for character segmentation of degraded text pages [61].

5. New techniques for OCR postprocessing were proposed [59, 60]. It is observed that current commercial OCR systems often recognize almost equivalent image patterns as different string patterns. In the techniques proposed, consistency analysis of inter-word relations at the image level and the symbolic level is applied to detect and correct potential OCR errors.

6. Different linguistic constraints were explored to solve the problem of word candidate selection for degraded text recognition [62, 63]. A new relaxation algorithm that uses word collocation data collected from large text corpora was proposed and tested. A lattice parser was proposed to take advantage of syntactic constraints for word candidate selection. They were integrated to achieve better performance.

7. A computational framework in which visual contextual constraints and different linguistic constraints can be exploited for passage-level postprocessing was proposed [58, 64]. Unlike the traditional approach of contextual analysis, which is limited to symbolic level processing and uses only an intra-word linguistic context, visual and linguistic inter-word constraints are integrated. The interaction between language-level analysis and image data distinguishes this from work in other areas such as speech recognition. More importantly, the use of visual inter-word relations gathered from an entire page or document is different from previous approaches that typically process only a single phrase or a sentence in isolation.

8. The methodology was generalized to other alphabetic languages and ideographic languages, such as Chinese and Japanese [65, 66].

Chapter 12

Future Directions

Some interesting further studies can be conducted. We conclude this thesis by briefly discussing several possible future directions.

12.1 Word Candidate Selection

In this work, we formalized the problem of passage level postprocessing as a problem of word candidate selection by assuming that a word candidate list can be provided by a word recognizer. Word candidate lists used in the experiments for word candidate selection were generated by a word shape analysis algorithm, which may not be practical for unrestricted English document recognition. Thus, further research should be done on word recognition algorithms that provide high quality candidate lists so that the techniques for word candidate selection can be used.

In the experiments for word candidate selection, the assumption is that the correct candidate is always included in the candidate list. This assumption should be relaxed to handle the case in which the correct candidate is not in the initial candidate list. New postprocessing techniques have to be developed to propose new candidates. Although we have already developed methods to use visual constraints to generate new word candidates, the exploitation of linguistic context for the problem is still an interesting problem for study.

12.2 Integration of Visual and Syntactic Constraints in Parsing

More work on the integration of visual and linguistic constraints can be carried out. In this thesis, experiments on this topic were concentrated on the integration of statistical language models and visual inter-word constraints. New methods can be developed that integrate visual inter-word constraints with language syntax inside lattice parsing.

12.3 Visual Inter-word Relation Computing

Visual inter-word relations calculated by image processing carry rich information about the symbolic content of document and typesetting characteristics. As we have demonstrated, many applications, such as character segmentation, font learning, word recognition and postprocessing, can make use of this information. In this thesis, we presented methods to calculate these relations. Visual inter-word relations are slow to compute through image matching. To improve efficiency, in this thesis, we proposed to use feature-matching and OCR results to guide image matching. Other information, such as connected component information and image profiles, can also be used to speed up image matching Similar to algorithms for string matching [30], fast algorithms for image matching can also be proposed.

The visual similarity measure discussed in this thesis works well for document images which are degraded by random noise or other uniformly distributed noise. For images with artifacts, such as character skew and perspective distortion [7, 81], more complex similarity measures can be developed.

12.4 A Unified Approach toward Text Recognition

Character segmentation, character recognition (OCR) and postprocessing are three major components of a text recognition system. Traditionally, they are treated as very different tasks and organized as a cascade (see Figure 12.1.(a)). For example, character segmentation is an image processing task, which tries to segment each word image into a sequence of character images; character recognition is a pattern classification task, which aims at mapping each character image onto its symbolic identification; and postprocessing is a symbolic processing task, which utilizes contextual information, such as lexical knowledge, to detect and correct potential character recognition errors.

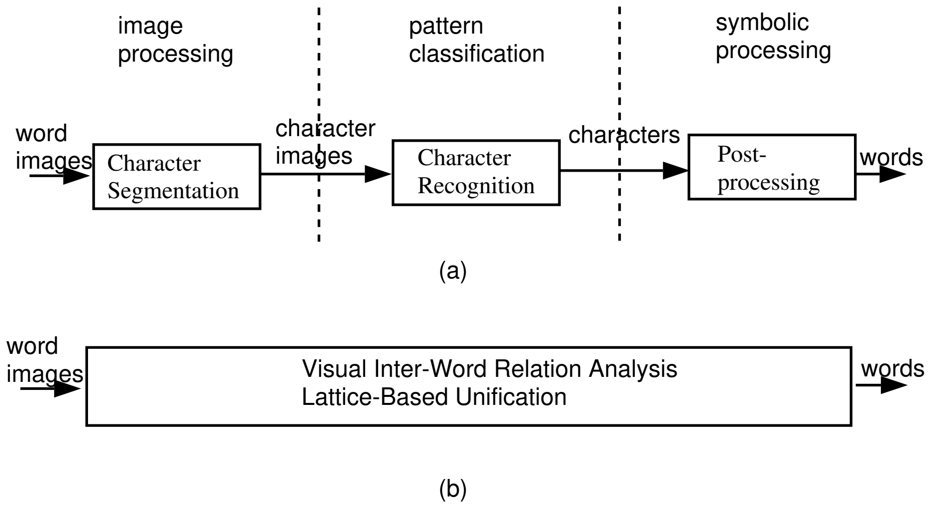


Figure 12.1: Models of text recognition. (a) the traditional approach; (b) the new approach.

We can extend our current work and propose a unified approach for text recognition by exploiting visual inter-word context. Under this approach, different stages of text recognition can be accomplished by the same set of operations– *inter-word relation analysis* and *lattice-based unification* (see Figure 12.1.(b)).

Given a text image, we assume that word segmentation has been done. Each word image can be represented by the coordinates of its bounding box in the text image. For each word image, a lattice can be defined as a data structure which keeps a record of the current result for the word image. The result can be derived from character segmentation, OCR, or visual contextual analysis. An entry in a lattice is a sub-image associated with its label which reflects the recognition result for a piece of image. A label can be a character, a string or a pointer to another image. At the beginning, the lattice for each word image is empty. New entries can be created and added to a lattice through a unification algorithm.

Given a text image, an image corresponding to a word or a string can be represented by its bounding box, (s_x, s_y, e_x, e_y), where (s_x, s_y) is the coordinate of the left-top pixel and (e_x, e_y) the one of the right-bottom pixel of the image. For an image $I(s_x, s_y, e_x, e_y)$, its lattice can be defined as a set,

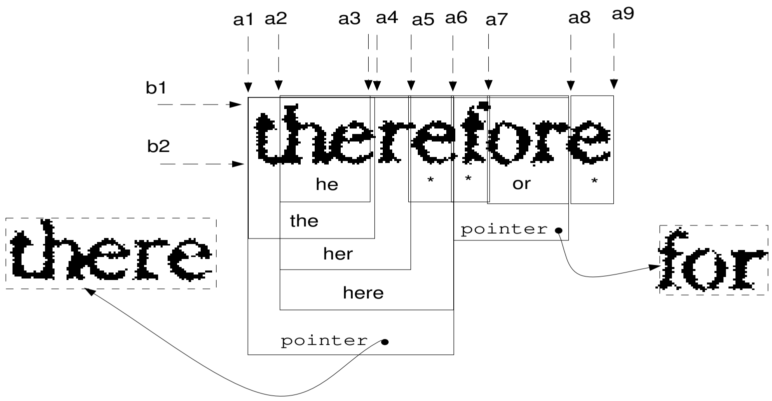
$$\begin{aligned} & L(s_x, s_y, e_x, e_y) \\ & \quad = \{(u, s_y, v, e_y, label, conf) \mid s_x \le u < v \le e_x\} \end{aligned}$$

where $(u, s_y, v, e_y, label, conf)$ is an entry for a fragment of the image $I(s_x, s_y, e_x, e_y)$: (u, s_y, v, e_y) is its bounding-box; *label* is its label; and *conf* is the confidence for the label. A label can be defined as

$$label = \ * \ \mid character \mid string \mid pointer$$

where "*" is a special label which indicates no recognition result yet for the segment of the image, and *pointer* is a pointer to the bounding box of another image fragment which is visually equivalent to the segment. Figure 12.2 is a word image and its lattice. The lattice stores results of segmentation, recognition and visual contextual constraints for the word image.

Based on the visual similarity measurement and inter-word relation calculation, discussed in this thesis, we can define $I(s1_x, s1_y, e1_x, e1_y) \approx I(s2_x, s2_y, e2_x, e2_y)$ if the two images can match each other very well.



$$
\begin{aligned}
& L(a1, b1, a9, b2) \\
& = \{(a1, b1, a4, b2, \text{“the”}, 0.99), \\
& \quad (a1, b1, a6, b2, pointer, 0.90), \\
& \quad (a2, b1, a3, b2, \text{“he”}, 0.90), \\
& \quad (a2, b1, a5, b2, \text{“her”}, 0.90), \\
& \quad (a2, b1, a6, b2, \text{“here”}, 0.95), \\
& \quad (a5, b1, a6, b2, \text{“*”}, 0.90), \\
& \quad (a6, b1, a7, b2, \text{“*”}, 0.95), \\
& \quad (a6, b1, a8, b2, pointer, 0.90), \\
& \quad (a7, b1, a8, b2, \text{“or”}, 0.88), \\
& \quad (a8, b1, a9, b2, \text{“*”}, 0.90)\}
\end{aligned}
$$

Figure 12.2: The lattice for a word image

If $I(s1_x, s1_y, e1_x, e1_y) \approx I(s2_x, s2_y, e2_x, e2_y)$, the lattice $L(s1_x, s1_y, e1_x, e1_y)$ can be upgraded to a new lattice, $L(s1_x, s1_y, e1_x, e1_y) \uplus L(s2_x, s2_y, e2_x, e2_y)$, which is the unification of the lattice $L(s1_x, s1_y, e1_x, e1_y)$ and the lattice $L(s2_x, s2_y, e2_x, e2_y)$. Formally,

$$\begin{aligned}
&L(s1_x, s1_y, e1_x, e1_y) \uplus L(s2_x, s2_y, e2_x, e2_y) \\
&\quad = L(s1_x, s1_y, e1_x, e1_y) \ \cup \\
&\qquad \{(u, s1_y, v, e1_y, label, conf) \mid s1_x \le u < v \le e1_x \text{ and} \\
&\qquad\quad (s2_x + u - s1_x, s2_y, s2_x + v - s1_x, e2_y, label, conf) \in L(s2_x, s2_y, e2_x, e2_y)\}
\end{aligned}$$

The left two examples in Figure 12.3 show the effect of the unification operation on two visually equivalent word images.

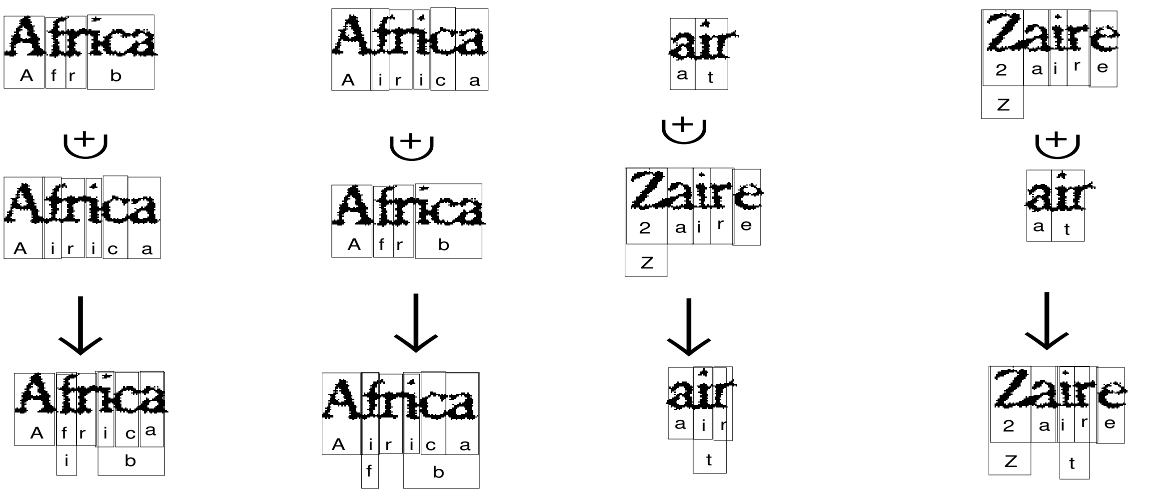


Figure 12.3: The unification of two visually equivalent images

Similarly, if the image $I(s1_x, s1_y, e1_x, e1_y)$ matches a part of the image $I(s2_x, s2_y, e2_x, e2_y)$, $I(s1_x, s1_y, e1_x, e1_y) \approx I(m1_x, s2_y, m2_x, e2_y)$ where $s2_x \le m1_x < m2_x \le e2_x$, we can define $L(s1_x, s1_y, e1_x, e1_y) \uplus L(s2_x, s2_y, e2_x, e2_y)$. The effect of the unification operation defined above is illustrated in the right two examples of Figure 12.3. In the general case, if two images can partially match, the unification operations between their lattices can also be defined.

The unification algorithm can be used for segmentation, recognition and postprocessing. Figure 12.4 shows an example that how the word image "`dates`" can be segmented into character level through two operations, $\uplus$ and $\odot$, where $\odot$ is an operation to reorganize a lattice so that information can be represented explicitly.

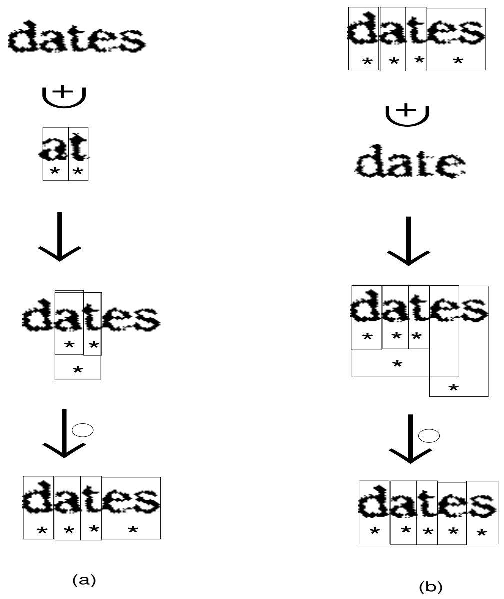


Figure 12.4: Character segmentation by unification

For an unlabeled entry in a lattice, an OCR can be applied to see if the image fragment can be recognized as a character with high confidence. If so, the character recognition result will replace the "*" in the entry. Traditional segmentation algorithm can also be use to provide segmentation points for a lattice.

Given a lattice, $L(s_x, s_y, e_x, e_y)$, for a word image, its recognition result can be defined as a set of complete paths from s_x to e_x. Figure 6.6 shows how word candidates are generated by a lattice-based search algorithm which uses a dictionary.

12.5 Handwritten Text Recognition

Handwritten text recognition is a similar, but more difficult task. Techniques designed here for machine-printed text recognition can be adapted to word candidate selection for this task. Significant research has been conducted on using linguistic contextual information to improve recognition performance [37, 156, 141, 159]. Because word segmentation sometimes is not an easy task, many possible word boundaries can be generated and the structure of a word lattice may become more complex: positions of word candidates may overlap (see the example inFigure 12.5). The word lattice parser can be generalized to handle these cases. Visual contextual information is also applicable. Figure 12.6 is a portion of a handwritten text page [138]. Because people write with a consistent style, visual similarities between word images can be observed (see Figure 12.7 for examples). Like what we demonstrated for machine-printed text recognition, visual inter-word relations can be useful to improve handwriting recognition performance. Given online/offline handwritten word scripts, methods can be designed to quantitatively measure visual similarity among words and to further analyze the writing style of the author.

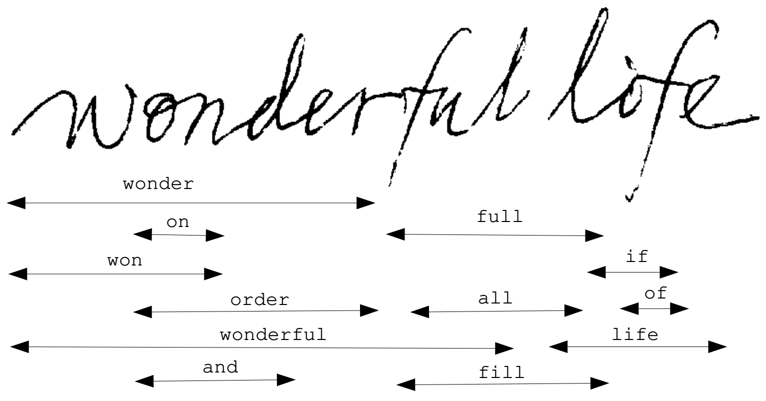
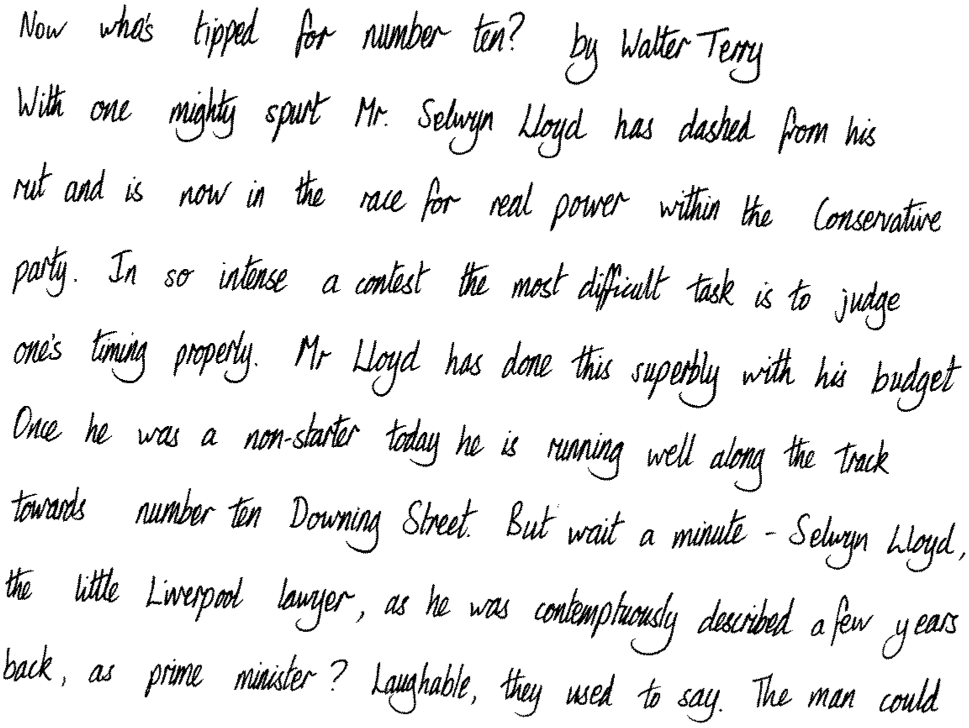
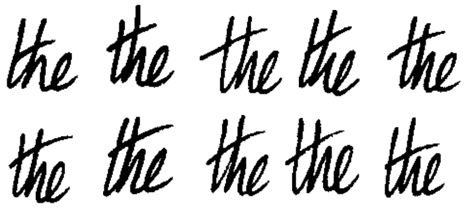


Figure 12.5: An example of word lattice for handwritten text recognition

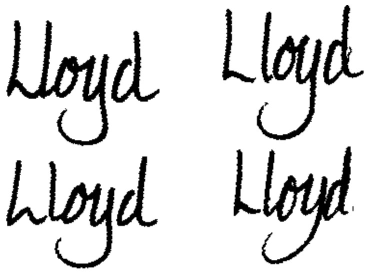


Now who's tipped for number ten? by Walter Terry
With one mighty spurt Mr. Selwyn Lloyd has dashed from his
rut and is now in the race for real power within the Conservative
party. In so intense a contest the most difficult task is to judge
one's timing properly. Mr Lloyd has done this superbly with his budget
Once he was a non-starter today he is running well along the track
towards number ten Downing Street. But wait a minute - Selwyn Lloyd,
the little Liverpool lawyer, as he was contemptuously described a few years
back, as prime minister? Laughable, they used to say. The man could

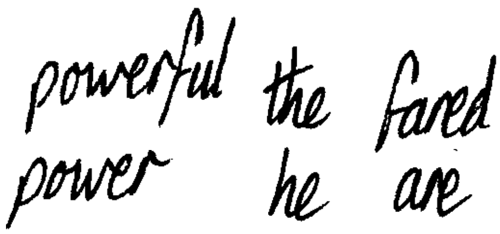
Figure 12.6: Handwritten text (A. W. Senior 1994)



(a)



(b)



(c)

Figure 12.7: Examples of visual similarity among handwritten words. (a) ten items of the functional word "*the*" which are visually similar (at least intuitively); (b) four items of the word "*Lloyd*" which are visually similar (character segmentation and recognition for those words will be easier if we consider that they have to be interpreted as the same word; and (c) examples of partial similarties which can be found between word images.

12.6 Summary

Image-based document retrieval [21, 71, 145] and text editing [5] are two interesting applications in which visual and linguistic information can also be utilized.

The objective of visual text recognition is to correctly transform an arbitrary image of text into its symbolic equivalent. When the text image is degraded, such a transformation cannot be finished in one step because the initial recognition results provided by pattern recognition methods often contain many errors. To improve the recognition results, visual and symbolic/linguistic contexts and the interaction between them are required. We believe that, by incorporating visual and linguistic contextual analysis, more effective OCR post-processing techniques can be designed allowing commercial document recognition systems to achieve high performance on degraded documents.

Bibliography

[1] J. S. Huang A. B. Wang and K. C. Fan. Optical recognition of handwriting chinese characters by partial matching. In *Proceedings of the Second International Conference on Document Analysis (ICDAR-93)*, pages 822–826, 1993.

[2] T. Akiyama and N. Hagita. Automated entry system for printed documents. *Pattern Recognition*, Vol. 23, No. 11:1141–1154, 1990.

[3] J. Allen. *Natural Language Understanding, second edition.* Benjamin/Cummings Publishing Company, Inc., 1995.

[4] H. Alshawi. Qualitative and quantitative methods of speech translation. In *Proceedings of the Workshop: The Balancing Act – Combining Symbolic and Statistical Approaches to Language*, July 1 1994.

[5] S.C. Bagley and G.E. Kopec. Editing images of text. *CACM*, 1994.

[6] H.S. Baird. Anatomy of a versatile page reader. *Proceedings of the IEEE*, 80(7):1059–1065, 1992.

[7] H.S. Baird. Document image defect models. In H. S. Baird, H. Bunke, and K. Yamamoto, editors, *Structural Document Image Analysis*, pages 546–556. Springer Verlag, N.Y., 1992.

[8] H.S. Baird. Document image defect models and their uses. In *Proceedings of the Second International Conference on Document Analysis and Recognition ICDAR-93*, pages 62–67, 1993.

[9] H.S. Baird and G. Nagy. A self-correcting 100-font classifier. In *Proceedings of the Conference on Document Recognition of 1994 S&T/SPIE Symposium*, February 6-10 1994.

[10] H.S. Baird and K. Thompson. Reading chess. *IEEE Transactions on pattern analysis and machine intelligence*, 12(6):552–559, 1990.

[11] M. Bokser. Omnidocument technologies. *Proceedings of the IEEE*, 80(7):1066–1078, 1992.

[12] R.D. Borsley. *Syntactic Theory: a Unified Approach.* Edward Arnold, 1991.

[13] T. Briscoe and J. Carroll. Generalized probabilistic LR parsing of natural language (corpora) with unification-based grammars. *Computational Linguistics*, 19(1):25–60, March 1993.

[14] P. Brown, J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, J. Lafferty, and R. Mercer. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, 1990.

[15] Thomas H. Carr. Perceiving visual language. In L. Kaufman and J. Thomas, editors, *Handbook of Perception and Human Performance*, page Chapter 29. New York, Wiley, 1992.

[16] R.G. Casey. Text OCR by solving a cryptogram. In *IEEE Proceedings*, pages 349–351, 1986.

[17] R.G. Casey. Character segmentation in document OCR: Progress and hope. In *Symposium on Document Analysis and Information Retrievals*, pages 13–40, April 24-26 1995.

[18] R.G. Casey and G. Nagy. Recursive segmentation and classification of composite character patterns. In *Proc. 6th ICPR*, pages 1023–1026, 1982.

[19] R.G. Casey and K.Y. Wong. Document-analysis systems and techniques. In R. Kasturi and M. Trivedi, editors, *Image Analysis and Applications*, pages 1–35. 1990.

[20] J.S. Chang, Y.F. Luo, and K.Y. Su. Gpsm: A generalised probabilistic semantic model for ambiguity resolution. In *Proceedings of the 30nd Annual Meeting of the Association for Computational Linguistics*, June 28 - July 2 1992.

[21] F.R. Chen, L.D. Wilcox, and D.S. Bloomberg. Detecting and locating partially specified keywords in scanned images using hidden markov models. In *Proceedings of the Second International Conference on Document Analysis and Recognition ICDAR-93*, pages 133–138, 1993.

[22] S. Chen, S. Subramaniam, R.M. Haralick, and I.T. Phillips. Performance evaluation of two OCR systems. In *Symposium on Document Analysis and Information Retrievals*, pages 299–317, 1993.

[23] L.F. Chien, K.J. Chen, and L.S. Lee. An Augmented Chart Data Structure with Efficient Word Lattice Parsing Scheme in Speech Recognition Application. In *COLING 1990*, 1990.

[24] B.H. Chou and J.S. Chang. Language models for chinese character recognition. In *Proceedings of ROCLING V*, pages 259–286, 1986.

[25] K.W. Church. A Stochastic Part Program and Noun Phrase Parser for Unrestricted Text. In *Second Conference on Appled Natural Language Processing*, 1988.

[26] K.W. Church, W. Gale, P. Hank, and D. Hindle. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29, 1990.

[27] K.W. Church, W. Gale, P. Hank, and D. Hindle. Using statistics in lexical analysis. In Uri Zernik, editor, *Lexical Acquisition: Exploitung On-Line Resources to Build a Lexicon*, pages 115–164. Lawrence Erlbaum Associates, Publishers, 1991.

[28] K.W. Church and R.L. Mercer. Introduction to the special issue on computational linguistics using large corpora. *Computational Linguistics*, 19(1):1–24, March 1993.

[29] P.R. Cohen and E.A. Feigenbaum, editors. *The Handbook of Artificial Intelligence, Volume 3*. William Kaufmann, INC., 1982.

[30] T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms*. The MIT Press, 1990.

[31] D.A. Dahl, L.M. Norton, and S.L. Taylor. Improving OCR accuracy with linguistic knowledge. In *Proceedings of the Second Annual Symposium on Document Anaysis and Information Retrieval*, pages 169–177, Las Vegas, Nevada, April 26-28 1993.

[32] B.V. Dasarathy. *Nearest Neighbor(NN) Norms: NN Pattern Classification Techniques*. IEEE Computer Society Press, 1991.

[33] G. DeSilva and J.J. Hull. Proper noun location in document images. *Pattern Recognition*, pages 311–320, February 1994.

[34] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. Addison-Wesley, New York, 1973.

[35] D.G. Elliman and I.T. Lancaster. A review of segmentation and contextual analysis techniques for text recognition. *Pattern Recognition*, 23(3/4):337–346, 1990.

[36] J. Esakov, D.P. Lopresti, J.S. Sandberg, and J. Zhou. Issues in automatic OCR error classification. In *Symposium on Document Analysis and Information Retrievals*, pages 401–412, 1993.

[37] L.J. Evett, C.J. Wells, F.G. Keenan, T. Rose, and R.J. Whitrow. Using linguistic information to aid handwriting recognition. In *Proceedings of the International Workshop on Frontiers in Handwriting Recognition*, pages 303–311, 1991.

[38] Inc. ExperVision. *TypeReader professional for Windows, user's guide.* 1995.

[39] C. Fang and J. J. Hull. A modified character level deciphering algorithm for ocr in degraded documents. In *Proceedings of the Conference on Document Recognition, 1995 SPIE Symposium(SPIE95)*, February 1995.

[40] P. Filipski and J.J. Hull. Keyword selection from word recognition results using definitional overlap. In *Third Symposium on Document Analysis and Information Retrieval*, April 11-13 1994.

[41] K.S. Fu. *Syntactic Pattern Recognition and Applications.* Preentice-Hall, Engelwood Cliffs, N.J., 1982.

[42] T. Fujisaki, F. Jelinek, J. Cooke an E. Black, and T. Nishino. A Probabilistic Parsing Method for Sentence Disambiguation. In *International Parsing Workshop '89*, 1989.

[43] H. Fujisawa and K. Marukawa. Full-text search and document recognition of japanese text. In *Symposium on Document Analysis and Information Retrievals*, pages 55–80, April 24-26 1995.

[44] A. Goshtasby and R.W. Ehrich. Contextual word recognition using probabilistic relaxation labeling. *Pattern Recognition*, 21(5):455–462, 1988.

[45] J. Guo and H.C. Liu. Ph – a Chinese corpus for pinyin-hanzi transcription. Technical report, ISS Technical Report, National University of Singapore, 1992.

[46] J. A. Hartigan. *Clustering Algorithms.* John Wiley & Sons, 1975.

[47] P. Heeman and J. Allen. Detecting and correcting speech repair. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, June 27-30 1994.

[48] C. Hemphill and J. Picone. Chart Parsing of Stochastic Spoken Language Model. In *Speech and Natural Language Workshop*, 1989.

[49] D. Hindle and M. Rooth. Structural ambiguity and lexical relations. *Computational Linguistics*, 19(1):103–120, March 1993.

[50] T.K. Ho. *A Theory of Multiple Classifier Systems And Its Application to Visual Word Recognition.* PhD thesis, Computer Science Department of SUNY at Buffalo, 1992.

[51] T.K. Ho and H.S. Baird. Perfect metrics. In *Proceedings of the Second International Conference on Document Analysis and Recognition ICDAR-93*, pages 593–597, 1993.

[52] T.K. Ho and H.S. Baird. Asymptotic accuracy of two-class discrimination. In *Symposium on Document Analysis and Information Retrievals*, pages 275–288, 1994.

[53] T.K. Ho and H.S. Baird. Evaluation of OCR accuracy using synthetic data. In *Symposium on Document Analysis and Information Retrievals*, pages 413–422, April 24-26 1995.

[54] T.K. Ho, J.J. Hull, and S.N. Srihari. Word recognition with multi-level contextual knowledge. In *Proceedings of the First International Conference on Document Analysis (ICDAR-91)*, pages 905–915, 1991.

[55] T.K. Ho, J.J. Hull, and S.N. Srihari. A word shape analysis approach to lexicon based word recognition. *Pattern Recognition letters*, 13:821–826, 1992.

[56] J.R. Hobbs, D.E. Appelt, J.Bear, and M. Tyson. Robust Processing of Real-World Natural Language Texts. In *Proceedings of 3rd Conference on Applied Natural Language Processing*, pages 186–182, 1992.

[57] J.R. Hobbs and J. Bear. Two Principles of Parse Preference. In *COLING-90*, 1990.

[58] T. Hong. Integration of visual inter-word constraints and linguistic knowledge in degraded text recognition. In *Proceedings of 32nd Annual Meeting of Association for Computational Linguistics, Student Sessions*, pages 328–330, 27-30 June 1994.

[59] T. Hong and J. J. Hull. Improving OCR performance with word image equivalence. In *Symposium on Document Analysis and Information Retrievals*, pages 177–190, April 24-26 1995.

[60] T. Hong and J. J. Hull. Visual inter-word relations and their use for OCR postprocessing. In *Proceedings of the Third International Conference on Document Analysis (ICDAR-95)*, August 14-16 1995.

[61] T. Hong and J. J. Hull. Visual inter-word relations and their use in character segmentation. In *Proceedings of the Conference on Document Recognition, 1995 SPIE Symposium(SPIE95)*, February 1995.

[62] T. Hong and J.J. Hull. Text recognition enhancement with a probabilistic lattice chart parser. In *Proceedings of the Second International Conference on Document Analysis (ICDAR-93)*, 1993.

[63] T. Hong and J.J. Hull. Degraded text recognition using word collocation. In *Proceedings of the Conference on Document Recognition of 1994 IS&T/SPIE Symposium*, pages 334–342, February 6-10 1994.

[64] T. Hong and J.J. Hull. Degraded text recognition using word collocation and visual inter-word constraints. In *Proceedings of the Forth Conference of Applied Natural Language Processing(ANLP94)*, October 1994.

[65] T. Hong, S. W. Lam, J. J. Hull, and S. N. Srihari. The design of a nearest-neighbor classifier and its use for japanese character recognition. In *Proceedings of the Third International Conference on Document Analysis (ICDAR-95)*, August 14-16 1995.

[66] T. Hong, S. W. Lam, J. J. Hull, and S. N. Srihari. Visual similarity analysis of chinese characters and its uses in japanese OCR. In *Proceedings of the Conference on Document Recognition, 1995 SPIE Symposium(SPIE95)*, February 1995.

[67] X.D. Huang, Y. Ariki, and M.A. Jack. *Hidden Markov Models for Speech Recognition.* Edinburgh University Press, 1990.

[68] J.J. Hull. Inter-word constraints in visual word recognition. In *Proceedings of the Conference of the Canadian Society for Computational Studies of Intelligence*, May 21-23 1986.

[69] J.J. Hull. *A Computational Theory of Visual Word Recognition.* PhD thesis, Computer Science Department of SUNY at Buffalo, 1987.

[70] J.J. Hull. Incorporation of a markov model of language syntax in a text recognition algorithm. In *Symposium on Document Analysis and Information Retrievals*, pages 174–185, March 16-18 1992.

[71] J.J. Hull. Document image matching and retrieval with multiple distortion-invariant descriptors. In *IAPR Workshop On Document Analysis Systems*, pages 383–400, Oct. 18-20 1994.

[72] J.J. Hull, S. Khoubyari, and T.K. Ho. Visual global context: Word image matching in a methodology for degraded text recognition. In *Symposium on Document Analysis and Information Retrievals*, pages 26–39, March 16-18 1992.

[73] J.J. Hull, S. Khoubyari, and T.K. Ho. Word image matching as a technique for degraded text recognition. In *Proceedings of 11th International Conference on Pattern Recognition*, 1992.

[74] J.J. Hull and S.N. Srihari. Experiments in text recognition with binary n-gram and viterbi algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-4(5):520–530, 1982.

[75] P. Jacobs. Making sense of lexical acquisition. In Uri Zernik, editor, *Lexical Acquisition: Exploitung On-Line Resources to Build a Lexicon*, pages 29–44. Lawrence Erlbaum Associates, Publishers, 1991.

[76] F. Jelinek. Self-organized language modeling for speech recognition. In K. F. Lee, editor, *Reading of Speech Recognition*.

[77] M.A. Jones, G.A. Story, and B.W. Ballard. Integrating multiple knowledge sources in a bayesian OCR post-processor. In *Proceedings of ICDAR-91*, pages 925–933, 91.

[78] Y. Tanabe K. Seino and K. Sakai. A linguistic post-processing based on word occurrence probability. In S. Impedovo and J.C. Simon, editors, *From Pixels to Features III: Frintiers in Handwriting Recognition*, pages 191–199. Elsevier Science Publishers, Amsterdam, 1992.

[79] S. Kahan, T. Pavlidis, and H.S. Baird. On the recognition of printed characters of any font and size. *IEEE Transactions on pattern analysis and machine intelligence*, 9(2), 1987.

[80] T. Kanungo, H.S. Baird, and R.M. Haralick. Validation and estimation of document degradation models. In *Symposium on Document Analysis and Information Retrievals*, pages 217–228, April 24-26 1995.

[81] T. Kanungo, R.M. Haralick, and I. Phillips. Global and local document degradation models. In *Proceedings of the Second International Conference on Document Analysis and Recognition ICDAR-93*, pages 730–734, 1993.

[82] M. Kay. Algorithm Schemata and Data Structures in Syntactic Processing. Technical Report 80-12, CSL, October 1980.

[83] M. Kay. Parsing in Functional Unification Grammar. In Barbara J. Grosz, Karen Sparck Jones, and Bonnie Lynn Webber, editors, *Reading in Natural Language Processing*, pages 35–70. Morgan Kaufman Publishers, Inc., 1986.

[84] S. Khoubyari and J.J. Hull. Keyword location in noisy document images. In *Proceedings of the Second Annual Symposium on Document Anaysis and Information Retrieval*, Las Vegas, Nevada, April 26-28 1993.

[85] S. Khoubyari and J.J. Hull. Font identification using visual global context. In *Proceedings of the Conference on Document Recognition of 1994 IS&T/SPIE Symposium*, San Jose, CA, February 6-10 1994.

[86] S. Khoubyari and J.J. Hull. Font and function word identification in document recognition. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 1995. to appear.

[87] G. E. Kopec and P. A. Chou. Document image decoding using markov sources. *PAMI*, 1994.

[88] R. Krovetz. Lexical acquisition and information retrieval. In Uri Zernik, editor, *Lexical Acquisition : Exploiting on-line Resources to Build a Lexicon*, pages 45–64. Lawrence Erlbaum Associates, Hillsdale,NJ, 1991.

[89] H. Kucera and W. N. Francis. *Computational Analysis of Present-day American English*. Brown University Press, 1967.

[90] K. Kukich. Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24(4):377–439, December 1992.

[91] R. Labuz. *Typography and Typesetting*. Van Nostrand Reinhold, New York, 1988.

[92] K.F. Lee. *Automatic Speech Recognition: The Development of the SPHINX System*. Kluwer Academic Publishers, Boston, 1989.

[93] S. Liang, M. Ahmadi, and M. Shridhar. Segmentation of touching characters in printed document recognition. In *Proceedings of the Second International Conference on Document Analysis and Recognition ICDAR-93*, pages 569–572, 1993.

[94] P.P. Lippman. An introduction to computing with neural nets. *IEEE ASSP Magazine*, 4:4–22, 1987.

[95] D. Lopresti and J. Zhou. Using consensus sequence voting to correct OCR errors. In *IAPR Workshop On Document Analysis Systems*, pages 191–202, Oct. 18-20 1994.

[96] K. Lunde. *Understanding Japanese Information Processing*. O'Reilly & Associates, Inc., 1993.

[97] A.K. Mackworth. Constraint Satisfaction. In S. Shapiro, editor, *Encyclopedia of Artificial Intelligence*, pages 285–293. 1992.

[98] D.M. Magerman. *Natural Language Parsing as Statistical Pattern Recognition*. PhD thesis, Stanford University, February 1994.

[99] D.M. Magerman and M.P. Marcus. Parsing a Natural Language Using Mutual Information Statistics. In *Proceeding of AAAI-90*, 1990.

[100] D.M. Magerman and M.P. Marcus. Pearl: A Probabilistic Chart Parser. In *Proceedings of 5th Conference of the European Chapter of the Association for Computational Linguistics*, 1991.

[101] M. Marcus. *A Theory of Syntactic Recognition for Natural Language.* MIT Press, 1980.

[102] D.D. McDonald. Robust Partial-Parsing Through Incremental, Multi-Algorithm Processing. In Paul S. Jacobs, editor, *Text-Based intelligent Systems.* Lawrence Erlbaum Associates, Hillsdale, NJ, 1991.

[103] M. Meteer and H. Gish. Integrating symbolic and statistical approaches in speech and natural language applications. In *Proceedings of the Workshop: The Balancing Act – Combining Symbolic and Statistical Approaches to Language*, July 1 1994.

[104] G.A. Miller and C. Fellbaum. Semantic network of english. *Cognition*, 41, 1991.

[105] B.M.E. Moret. Decision trees and diagrams. *ACM Computing Surveys*, 14:593–623, December 1982.

[106] S. Mori, C.Y. Suen, and K. Yamamoto. Historical review of OCR research and development. *Proceedings of the IEEE*, 80(7):1029–1058, 1992.

[107] G. Nagy. At the frontiers of OCR. *Proceedings of the IEEE*, 80(7):1093–1100, 1992.

[108] G. Nagy, S. Seth, and K. Einspahr. Decoding substitution ciphers by means of word matching with application to OCR. *PAMI*, (9):710–715, 1987.

[109] G. Nagy, S. Seth, and M. Viswanathan. A prototype document image analysis system for technical journals. *Computer*, 25(7):10–24, 1992.

[110] A. Newell, J. Barnett, J.W. Forgie, C. Green, D. Klatt, J.C.R Licklider, J. Munson, D.R. Reddy, and W.A. Woods. *Speech Understanding Systems – Final Report of a Study Group*. North-Holland/American Elsevier, 1973.

[111] T. Pavlidis. Problems in the recognition of poorly printed text. In *Symposium on Document Analysis and Information Retrievals*, pages 162–173, 1992.

[112] I.T. Phillips, S. Chen, J. Ha, and R.M. Haralick. English document database design and implementation methodology. In *Symposium on Document Analysis and Information Retrievals*, pages 65–104, 1993.

[113] A. Pollatsek and K. Rayner. Reading. In M. Posner, editor, *Foundation Of Cognitive Science*. MIT Press, 1989.

[114] P. Price. Combining linguistic with statistical methods in automatic speech understanding. In *Proceedings of the Workshop: The Balancing Act – Combining Symbolic and Statistical Approaches to Language*, July 1 1994.

[115] L. R. Rabiner and B. H. Juang. An introduction to hidden markov models. *IEEE ASSP Magazine*, pages 4–16, 1986.

[116] A. Radford. *Transformational Grammar: A First Course*. Cambridge University Press, 1988.

[117] Inc. RAF Technology. Programmer's guide to the dafs library, release 0.7. Technical report, December 9 1994.

[118] Inc. RAF Technology. Dafs: Document attribute format specification. Technical report, January 5 1995.

[119] Inc. RAF Technology. Illuminator user's manual, release 0.9. Technical report, May 31 1995.

[120] K. Rayner and A. Pollatsek. *Psychology of Reading.* Prentice Hall, 1989.

[121] D.R. Reddy. Speech recognition by machine: a review. In *Proceedings of ICASSP 1992*, 1992.

[122] S. Reichardson. Bootstrapping statistical processing into a rule-based natural language parser. In *Proceedings of the Workshop: The Balancing Act – Combining Symbolic and Statistical Approaches to Language*, July 1 1994.

[123] P.S. Resnik. *Selection and Information: A Class-Based Approach to Lexical Relationships.* PhD thesis, Computer and Information Science Department of UPENN, 1993.

[124] S.V. Rice, F.R. Jenkins, and T.A. Nartker. The fourth annual test of OCR accuracy. In *1995 Annual Report of ISRI, University of Nevada, Las Vegas*, pages 11–50, 1995.

[125] S.V. Rice, J. Kanai, and T.A. Nartker. An evaluation of OCR accuracy. In *1993 Annual Report of ISRI, University of Nevada, Las Vegas*, pages 9–34, 1993.

[126] S.V. Rice, J. Kanai, and T.A. Nartker. The third annual test of OCR accuracy. In *1994 Annual Report of ISRI, University of Nevada, Las Vegas*, pages 11–40, 1994.

[127] E.S. Ristad. Computational structure of gpsg models. *Linguistics and Philosophy*, 13:521–587, 1990.

[128] C.P. Rose and A. Waibel. Recoving from parser failure: A hybrid statistical/symbolic approach. In *Proceedings of the Workshop: The Balancing Act – Combining Symbolic and Statistical Approaches to Language*, July 1 1994.

[129] T.G. Rose and L.J. Evett. Text recognition using collocations and domain codes. In *Proceedings of the Workshop on Very Large Corpora:Academic and Industrial Perspectives*, pages 65–73, 1993.

[130] T.G. Rose, L.J. Evett, and R.J. Whitrow. The Use of Semantic Information as an Aid to Handwriting Recognition. In *Proceedings of the First International Conference on Document Analysis (ICDAR-91)*, pages 629–637, 1991.

[131] A. Rosenfeld, R.A. Hummel, and S.W. Zucker. Scene labeling by relaxation operations. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(6):420–433, June 1976.

[132] D.E. Rumelhart and J.L. McClelland. An interactive activation model of context effects in letter perception. *Psychological Review*, 89(1):60–94, 82.

[133] K. Yamamoto S. Mori and M. Yasuda. Research on machine recognition of handprinted characters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(4):386–405, July 1984.

[134] R. Schartz and S. Austin. Efficient, high-performance algorithms for n-best search. In *Proceedings of the DARPA Speech and Natural Language Workshop 1990*, pages 6–11, 1990.

[135] J. Schurmann, N. Bartneck, T. Bayer, J. Franke, E. Mandler, and M. Oberlander. Document analysis – from pixels to contents. *Proceedings of the IEEE*, 80(7):1101–1119, 1992.

[136] P. Sells. *Lectures on Contempary Syntactic Theories: An introduction to Government-Binding Theory, Generalized Phrase Structure Grammar and Lexical-Functional Grammar.* CSLI:Stanford, 1985.

[137] S. Seneff. Robust parsing for spoken language systems. In *Proceedings of ICASSP 1992*, 1992.

[138] A.W. Senior. *Off-line Cursive Handwritting Recognition Using Recurrent Neural Networks.* PhD thesis, Engineering Department of Cambridge University, 1994.

[139] E.V. Siegel and K.R. McKeown. Emergent linguistic rules from inducing decision trees: Disambiguation discourse clue words. In *Proceedings of AAAI*, 1994.

[140] B. Slator. Using context for sense preference. In U. Zernik, editor, *Lexical Acquisition: Exploitung On-Line Resources to Build a Lexicon*, pages 65–96. Lawrence Erlbaum Associates, Publishers, 1991.

[141] R.K. Srihari, S. Ng, M. Baltus, and J. Kud. Use of language models in on-line sentence/phrase recognition. In *Third International Workshop on Frontiers in Handwriting Recognition*, pages 284–294, May 25-27 1993.

[142] S.N. Srihari. *Computer Text Recognition and Error Correction.* IEEE Computer Society Press, 1984.

[143] S.N. Srihari. High-performance reading machines. *Proceedings of the IEEE*, 80(7):1120–1132, 1992.

[144] S.N. Srihari. From pixels to paragraphs: the use of models in text recognition. In *Symposium on Document Analysis and Information Retrievals*, pages 47–64, 1993.

[145] G.A. Story, L.O'Gorman, D. Fox, L.L. Schaper, and H.V. Jagadish. The rightpages image-based electronic library for alerting and browsing. *IEEE Computer*, pages 17–26, September 1992.

[146] T. Strzalkowski and B. Vauthey. Information retrieval using robust natural language processing. In *Proceedings of the 30nd Annual Meeting of the Association for Computational Linguistics*, June 28 - July 2 1992.

[147] Calera Recognition Systems. *WordScan Plus, user's guide.* 1994.

[148] K. Taghva, J. Borsack, and A. Condit. An expert system for automatically correcting OCR output. In *Proceedings of the Conference on Document Recognition of 1994 IS&T/SPIE Symposium*, pages 270–278, February 6-10 1994.

[149] K. Taghva, J. Borsack, A. Condit, and S. Erva. The effects of noisy data on text retrieval. In *1993 Annual Report of ISRI, University of Nevada, Las Vegas*, pages 71–82, 1993.

[150] C.C. Tappert, C.Y. Suen, and T. Wakahara. The state of the art in on-line handwriting recognition. *IEEE Transactions on pattern analysis and machine intelligence*, 12(8):787–808, 1990.

[151] M. Tomita. *Efficient Parsing For Natural Language*. Kluwer Academic Publishers, 1985.

[152] M. Tomita. An Efficient Word Lattice Parsing Algorithm for Continuous Speech Recognition. In *Proceedings of the International Conference on Acoustic, Speech and Signal Processing*, 1986.

[153] M. Tomita. Why parsing technologies? In Masaru Tomita, editor, *Current Issues in Parsing Technology*, pages 1–9. Kluwer Academic Publishers, 1991.

[154] S. Tsujimoto and H. Asada. Resolving ambiguity in segmenting touching characters. In *1st Int. Conf. on Document Analysis and Recognition*, pages 701–709, October 1991.

[155] S. Tsujimoto and H. Asada. Major components of a complete text reading system. *Proceedings of the IEEE*, 80(7):1133–1149, 1992.

[156] R.K. Srihari V. Govindaraju and S.N. Srihari. Handwritten text recognition. In *IAPR Workshop On Document Analysis Systems*, pages 157–174, Oct. 18-20 1994.

[157] T. Wasow. Grammatical theory. In M. Posner, editor, *Foundation of Cognitive Science*, pages 161–205. MIT Press, 1989.

[158] C. J. Wells, L. J. Evett, P. E. Whitby, and R. J. Whitrow. Fast dictionary look-up for contextual word recognition. *Pattern Recognition*, 23(5):501–508, 1990.

[159] C.J. Wells, L.J. Evett, and R.J. Whitrow. Word Look-Up for Script Recognition – choosing a candidate. In *Proceedings of the First International Conference on Document Analysis (ICDAR-91)*, pages 620–628, 1991.

[160] G.M. White. Natural language understanding and speech recognition. *Communications of the ACM*, 33(8):72–82, 1990.

[161] Y. Wilks, L. Guthrie, J. Guthrie, and J. Cowie. Combining weak methods in large-scale text processing. In Paul S. Jacobs, editor, *Text-Based intelligent Systems.* Lawrence Erlbaum Associates, Hillsdale, NJ, 1991.

[162] T. Winograd. *Language as a Cognitive Process, Volume I: Syntax.* Addison-Wesley Publishing Company, 1983.

[163] W.A. Woods. Tansition Network Grammars for Natural Language Analysis. In Barbara J. Grosz, Karen Sparck Jones, and Bonnie Lynn Webber, editors, *Reading in Natural Language Processing*, pages 71–88. Morgan Kaufman Publishers, Inc., 1986.

[164] L. Xu, A. Krzyzak, and C. Y. Suen. Method of combining multiple classifiers and their application to handwritten character recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-22:418–435, 1992.

[165] S.R. Young, A.G. Hauptmann, W.H. Ward, E.T. Smith, and P. Werner. High level knowledge sources in usable speech recognition systems. *Communications of the ACM*, 32(2):183–194, 1989.

[166] D.H. Younger. Recognition and Parsing of Context-Free Language in Time n^3. *Information and Control*, 10:189–208, 1967.

[167] U. Zernik. Introduction. In Uri Zernik, editor, *Lexical Acquisition: Exploitung On-Line Resources to Build a Lexicon*, pages 1–26. Lawrence Erlbaum Associates, Publishers, 1991.

[168] X. Z. Zhang. *Techniques for Chinese Character Recognition.* Qinghua University, 1992.

[169] V. Zue, J. Glass, D. Goodine, H. Leung, M. McCandless, M. Phillips, J. Polifroni, and S. Seneff. Recent progress on the voyager system. In *Speech and Natural Language Workshop '90*, pages 206–211, 1990.