INDEXING AND RETRIEVAL OF LOW QUALITY HANDWRITTEN DOCUMENTS

by

HUAIGU CAO

September 2008

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

Department of Computer Science and Engineering

INDEXING AND RETRIEVAL OF LOW QUALITY HANDWRITTEN DOCUMENTS

A Dissertation Presented

by

HUAIGU CAO

Approved as to style and content by:

Dr. Venu Govindaraju , Chair

Dr. Peter Scott, Member

Dr. Vipin Chaudhary, Member

Dr. Jan Chomicki, Director of Graduate Study
Department of Computer Science and Engineering

ABSTRACT

Decades of the development in document analysis and recognition techniques has made it possible to convert large amount of documents into electronic formats and store them into computers. In recent years, the achievement in information retrieval has provided a powerful tool for prompt access to the information that lies in the documents. Inspired by the success of applications in the above two areas, in this thesis, we investigate methods that aim at improving the performance of retrieving handwritten document images. Unlike the retrieval of machine-printed documents from which we will anticipate very high OCR accuracy, the retrieval of handwritten document images is more challenging due to document analysis and recognition errors.

In existing methods to retrieve handwritten document images, usually the index is built on the text collected from top-n ($n > 1$) candidates returned by a word recognizer. Different weights may apply to the candidates according to their ranks. Effective as these primitive methods are, with the assumptions of flawless word segmentation and isolated word recognition, these methods are vulnerable by word segmentation errors and cannot take advantage of the language model which has become a standard component in the state-of-the-art handwriting recognition systems. However, incorporation of the word segmentation scores (probabilities) and language model into any existing indexing techniques in general increases the complexity of the problem. In our indexing method, we solved this challenging problem by separating the term counts from standard IR models, estimating them on the word sequence level, and plugging them back in the IR models. A fast algorithm using dynamic programming was proposed to reduce the time complexity. In addition to the application

in document retrieval, we also used the word segmentation information in keyword retrieval.

In another major contribution of this paper, we applied the Markov random field (MRF) modeling to the binarization problem. The MRF can precisely describe the constraint of local smoothness in the image. We can also use the constraint of smoothness to remove the grid from the form image, which is a very useful application in form image preprocessing. This research work virtually addresses a general topic in the preprocessing of degraded handwritten document images. Applications in both handwriting recognition and handwritten document image retrieval can benefit from our approach.

TABLE OF CONTENTS

Page

ABSTRACT . . . . . **iv**

LIST OF TABLES . . . . . **ix**

LIST OF FIGURES . . . . . **x**

CHAPTER

1. INTRODUCTION **1**

1.1 Motivations 1
1.2 Challenges 2
1.3 Research Topics 4

2. BACKGROUND **5**

2.1 State of the Art in Off-line Handwriting Recognition 5

2.1.1 Handwriting Recognizers 5
2.1.2 Feature Selection and Extraction 7
2.1.3 Language Modeling 9

2.2 IR Techniques 12
2.3 Overview of Prior Works 12

2.3.1 Document Image Binarization 12
2.3.2 Handwritten Document Retrieval 14
2.3.3 Keyword Spotting 16

3. HANDWRITTEN IMAGE BINARIZATION BASED ON MARKOV RANDOM FIELDS **23**

3.1 Introduction 23
3.2 Related Work 25

3.2.1 Locally Adaptive Methods for Binarization 25

3.2.2 Markov Random Field Based Approach to Binarization 26
3.2.3 Form Grid Removal 26

3.3 Markov Random Field model for handwritting images 27
3.4 Inference in the MRF Using Belief Propagation 29

3.4.1 Belief Propagation 29
3.4.2 An Example showing the BP Inference in the MRF 32
3.4.3 Learning the Prior Model $\Pr(x_j)$ and $\Pr(x_k|x_j)$ 39
3.4.4 Learning the Observational Model $\Pr(y_j|x_j)$ 43
3.4.5 Form Grid Removal 47
3.4.6 Pruning the Search Space of MRF Inference 48

3.5 Experimental Results and Analysis 51

3.5.1 Test Datasets 51
3.5.2 Display of Preprocessing Results 52
3.5.3 Results of Acceleration: Speed vs. Accuracy 53
3.5.4 Comparison to Other Preprocessing Methods 53

3.6 Summary 56

4. HANDWRITTEN DOCUMENT RETRIEVAL 61

4.1 Introduction 61
4.2 Vector IR Model for Handwritten Documents 63

4.2.1 Classic Vector Model 63
4.2.2 Modified Vector Model 65
4.2.3 Estimating Raw Term Frequency $freq_{i,j}$ 67
4.2.4 Estimating Word Segmentation Probability 70
4.2.5 Estimating Word Recognition Likelihood 73
4.2.6 Search Engine Based on Modified Vector Model 76
4.2.7 Computational Issues 77

4.3 Experimental Results and Discussions 78

4.3.1 Test Corpus 78
4.3.2 Preprocessing and Recognition of PCR Form Images 79
4.3.3 Evaluation Metrics of IR Test 82
4.3.4 IR Tests 83

4.4 Summary 87

5. HANDWRITTEN KEYWORD RETRIEVAL 90

5.1 Introduction 90
5.2 Model Definition 92

5.2.1 Word Spotting Model 92
5.2.2 Estimating Word Segmentation Probability 93
5.2.3 Estimating Word Recognition Probability 95

5.3 Experimental Results 96

5.3.1 Data Collection 96
5.3.2 Preprocessing 97
5.3.3 Evaluation Metrics 98
5.3.4 Keyword Retrieval Results 99

5.4 Summary 100

6. CONCLUSION 101

6.1 Contributions of the Thesis 101
6.2 Future Works 103

BIBLIOGRAPHY 105

LIST OF TABLES

Table **Page**

3.1 Comparison of the speed and accuracy of the proposed algorithm over different values of Pr_{min} tested on the PCR carbon form image (2420 × 370) in Figure 3.9. 52

3.2 Comparison of the speed and accuracy of the proposed algorithm over different values of Pr_{min} tested on the IAM image (2124×369) in Figure 3.12. 52

3.3 Comparison of word recognition rates of Milewski algorithm, MRF based approach, Niblack and Otsu algorithms (set #1: sample word images not affected by forms lines; set #2: sample word images affected by forms lines; overall: set #1 + set #2). 55

3.4 Comparison of word recognition rates (top-one accuracies in percentage) of the MRF based method, Niblack algorithm and Otsu algorithm on images with different noise levels. 55

4.1 Approaches to handwritten document retrieval. 63

4.2 28 query phrases used in our IR tests. 83

4.3 Approaches to handwritten document retrieval. 87

LIST OF FIGURES

Figure **Page**

1.1 An example of PCR forms. (a) A entire PCR form. (b) A small local region showing obscure text and background noise array. (c) Fields of interest in the PCR form. . . . 3

2.1 Directional features of a character images. . . . 8

2.2 Letter "O" and "Q". PCA may only extract common features of the images of "O" and "Q", but LDA may extract the difference between them . . . 9

2.3 Smoothing the language model by redistribution of the probability mass (backing off) . . . 11

2.4 The feature series used in DTW word spotting. . . . 18

2.5 Sakoe-Chiba band. . . . 19

2.6 A sample from George Washington's manuscripts. . . . 20

3.1 Stroke preserving line removal. (a) A word image with an underline across the text. (b) Binarized image with the underline removed. (c) Binarized image with the underline removed and fixed. . . . 25

3.2 The topology of the Markov network. (a) the input image y and the Inferred Image x; (b) the Markov network generalized from (a). In (b) each node x_i in the field is connected to its four neighbors. Each observation node y_i is connected to node x_i. An edge indicates the statistical dependency of two nodes. . . . 28

3.3 An acyclic Markov network. . . . 32

3.4 A cyclic Markov network. . . . 32

3.5 Shared patches in binary document image. . . . 33

3.6 114 representatives of shared patches obtained from clustering. 40

3.7 Binarized images from three writers for learning the prior model. 40

3.8 The smoothed gray-scale histogram and estimated foreground and background p.d.f. using two methods. Thresholding based method did not perform well at the intersection of two density functions, whereas EM algorithm based method improved the result. 44

3.9 A sample patch cropped from a carbon image in our test set. All pixels we intend to paint in are marked in black. 47

3.10 An example of PCR forms. (a) A entire PCR form. (b) A small local region showing obscure text and background noise array. (c) Fields of interest in the PCR form. 57

3.11 The binarization and line removal result of the sample shown in figure 3.9. 58

3.12 A sample from IAM database. 58

3.13 Comparison of binarization results of the MRF based algorithm versus three other algorithms. 59

3.14 Comparison of line removal results of the Milewski algorithm and the MRF based algorithm. 60

4.1 Three feature representing a gap between two consecutive connected components. 72

4.2 Genuine matching probability/score curve estimated from training set. 76

4.3 Flowchart of the search engine. 77

4.4 TF matrices from text IR and document image IR. The TF matrix for document image IR can be approximated by a sparse matrix if we turn the shadowed elements that are below a threshold to 0. 78

4.5 An example of PCR forms. (a) A entire PCR form. (b) A small local region showing obscure text and background noise array. (c) Fields of interest in the PCR form. 80

4.6 An example of the binarization and line removal result. 81

4.7 The performance of word segmentation (recall-precision curve). 81

4.8 The MAP and R-Precision values of 7 IR tests. 86

4.9 The 11-point average precision curves of tests 1, 5 and 7. 88

5.1 Diagram of the keyword spotting system. 92

5.2 The text in a PCR form. 97

5.3 An example of the binarization and line removal result. 98

5.4 11-point average precision curves of Tests 1-2. 98

CHAPTER 1

INTRODUCTION

1.1 Motivations

Decades of the development in optical character recognition (OCR) techniques has made it possible to convert great volumes of documents into digital forms such as plan text, PDF, XML and store them in the computer. In recent years, the achievement in information retrieval has provided a powerful tool for indexing and searching large scale on-line database of documents. Although information retrieval has been successful on text edited in the computer, the IR performance on OCR'ed text will be impaired by document analysis and recognition errors. Researchers have shown that the performance of OCR text retrieval is badly affected when dealing with short or low quality documents [3, 17]. Although OCR has been successful in applications of machine-printed document recognition and handwriting recognition (HR) with small lexicon, unconstrained handwriting with large lexicon in general still has very low accuracy. According to the state of the art in word recognition technologies, the word recognition accuracy is 60-70% on handwritten documents of good quality which makes IR results acceptable, but only 20-30% on low-quality historical manuscript and carbon forms, so conventional IR algorithms perform very badly on these documents.

The objective of this thesis is to investigate approaches to improving the performance of indexing and retrieval of low quality handwritten document images. There are two popular applications in handwritten document retrieval: document retrieval and keyword retrieval (word spotting). Document retrieval approaches search for

documents within a data-set that are relevant to the given query phrase. A document retrieval system computes the doc-query similarity and ranks the documents according to their similarities. Word spotting approaches search for query words within a date-set. After preprocessing of document images and word segmentation, feature vectors are extracted from word images and stored in a database. When a user provides a query word, the similarity between the query and the word image in the database is computed, and word images are returned in the decreasing order of similarities.

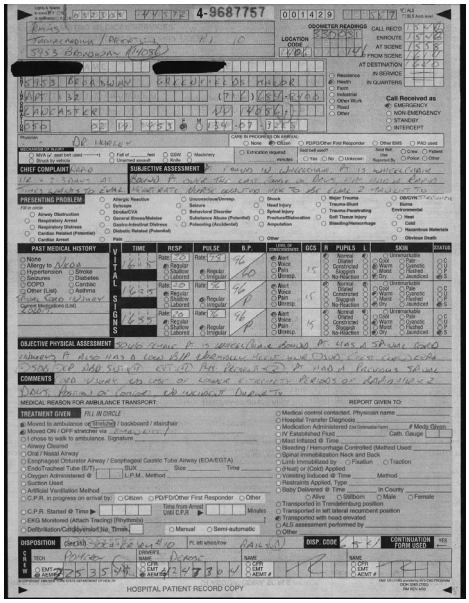
1.2 Challenges

Several challenges lie in handwritten document retrieval. Firstly, the quality of degraded document images is very bad. A typical category of the low quality image is the carbon images shown in Figure 5.2. If we binarize the carbon images in our data set with ordinary smoothing and binarization algorithms, the result can even be very hard for human beings to read. In addition to carbon images, we also use clean handwriting images with synthetic noise for the evaluation of binarization methods. Secondly, the handwriting is loosely constrained in terms of writing style and words chosen to use. Finally, the lack of an IR model appropriate for handwritten data that has large amount of OCR errors is also a challenge.

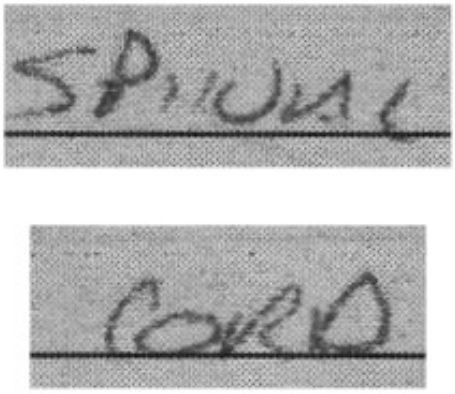
There are three possible research focuses of handwriting retrieval: preprocessing, handwriting recognition and information retrieval. Our research focuses on preprocessing and information retrieval:

1. Before we send the image to the recognizer it has to be binarized and the form grids have to be removed. If we could improve the quality of binarized images it would be possible to get a more acceptable recognition rate.

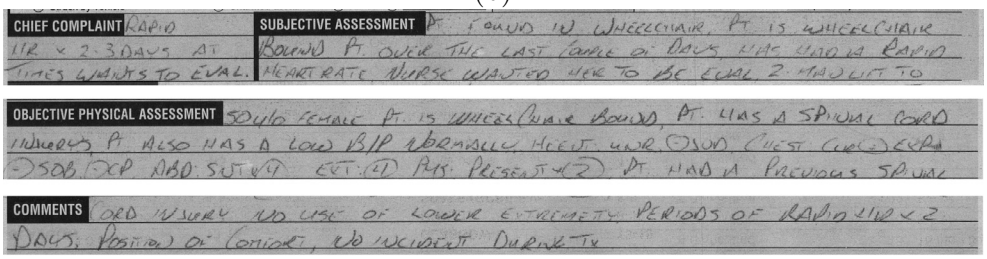
2. We also develop specific IR models for handwritten data. This is new to the information retrieval field. We will be exploring a specific IR model for handwrit-



(a)



(b)



(c)

Figure 1.1. An example of PCR forms. (a) A entire PCR form. (b) A small local region showing obscure text and background noise array. (c) Fields of interest in the PCR form.

ten data based on tight interaction of handwriting recognition and information retrieval techniques.

Handwriting recognition techniques have been developed and used for years and it's hard to improve. Thus we do not focus on this area.

1.3 Research Topics

One of our research advances is to investigate new IR models and techniques for handwritten document images. Indexing of handwritten document images are traditionally done on the OCR'ed text of handwritten document images using existing IR techniques. Due to the high error rate of handwriting recognition, the index built on the OCR'ed text loses lots of information of original documents and is far from ideal for retrieval. Several approaches based indexing of ranked OCR results [49, 24, 37] have been proposed. The purpose of using ranked OCR results is to improve recall rate. Suppose we use the text composed of top-10 word recognition candidates for retrieval, then the chance that the keyword we search for is within the text is larger than the chance when we search the text composed of only top-1 candidates. There are two directions to improve existing methods: On the one hand, we need to utilize as many as possible word image hypotheses which may not be considered in OCR tasks; on the other hand, we need to assign different weights to candidates at different ranks to maximize the precision rate.

Another research advance made by this thesis is to improve the quality of binarization result of low quality images. Although the binarization of document images is a widely defined subject, our research is based on the Bayesian approach. Most of the prior works for binarization are heuristic. Given the nature of binarization problem (which is basically low-level image processing), heuristic constraints are not always applicable and sufficient. By adopting Bayesian approaches, we will be able to generate "trainable" constraints and develop scalable algorithms.

CHAPTER 2

BACKGROUND

2.1 State of the Art in Off-line Handwriting Recognition

2.1.1 Handwriting Recognizers

There are two approaches to implementing the handwriting recognizer: *holistic* and *analytic*. The holistic approach treats a word word as a class and recognize a word image as a whole. The analytic approach segments and recognize the individual characters. The holistic approach has limited applications because of the difficulty to get enough training data when the number of classes increases. Most successful word recognition algorithms are based on analytic approach.

A typical analytic approach is based on character segmentation and searching algorithm. Usually over-segmentation is adopted and distances of different combinations of segments can be calculated. The best segmentation path with minimum distance is obtained by dynamic programming or some A*-type algorithm. For a few instances of this kind of word recognition algorithms see [6, 15, 33].

Another type of analytic algorithms is based on HMM. When the lexicon size is very large, it is not feasible to build an HMM for each class because there is not enough training data for each class. So the HMM for word recognition is usually a concatenation of character based models. In the HMM for word recognition, features are normally extracted from left to right using a sliding window, and the observation distribution is assumed to be mixture of Gaussian. Because of the linear, left-to-right direction of handwriting, a linear transition structure is often adopted (i.e. the state transition probabilities are chosen in such a way that a linear left-to-right ordering of

the states is imposed). Although a word HMM can be made by concatenating several character HMM's, there is no need to provide the character boundaries along with the transcription for training. Instead, in the training of the HMM, the character boundaries are automatically found by an EM algorithm (the Baum-Welch algorithm.) This property makes it possible to reduce the time required to prepare the transcription of large amount of training data. For a few instances of HMM based word recognition algorithms see [7, 34, 59].

Word sequence recognition [40] is to recognize a whole line or concatenation of several lines of words. Suppose f is a sequence of feature vectors, then word sequence recognition is to find a sequence of words s that maximize the probability

$$\hat{s} = \underset{s}{\operatorname{argmax}} \Pr(s|f) = \underset{s}{\operatorname{argmax}} \Pr(f|s) \cdot \Pr(s) \tag{2.1}$$

$\Pr(f|s)$ can be estimated by word recognition algorithm such as HMM, and $\Pr(s)$ can be estimated by a language model (n-grams). Similar to analytic word recognition that split a word into characters, word sequence recognition can be done by splitting a line into words and searching for the best path of line separation by dynamic programming. The advantage of word sequence recognition over single word recognition is that the use of language model reduces recognition errors. But this technique requires large amount of natural language text as training data.

The features used in word recognitions can be projection profiles, directional features, structural features (holes, ascenders, descenders, ...), and so on.

Lexicon plays an important role in cursive Latin handwriting recognition. A small lexicon reduces the difficulty of the problem greatly and this led to successful applications of automatic recognition of zip code, address, cheque, *etc.* Although off-line cursive handwriting recognition with large lexicon (5,000-30,000) remains an unsolved problem, the 60-80% accuracy rate on good quality, unconstrained handwriting has

been acceptable for commercial applications. However the accuracy on degraded manuscripts such as historical documents and carbon forms is still as low as 20-40%.

2.1.2 Feature Selection and Extraction

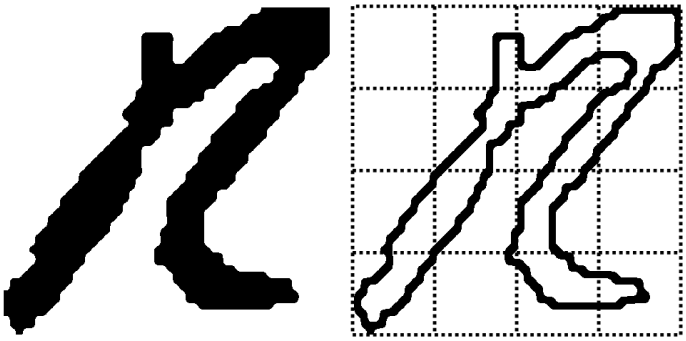
Features play an very important role in a handwriting recognition system. Various types of features can be computed from the handwritten document images:

1. the raw intensity of pixels,
2. statistics of local regions (mean, variance and other higher-order moments of the intensity),
3. features describing the connectivity of strokes including directional features and Gabor features.
4. concavity, run length and other structure features
5. numbers of ascenders and descenders,
6. intensity projection profile.

Most of the handwriting recognition systems use a combination of several types of features.

Among all of the different features mentioned above, the most powerful features are the directional features or directional element features. The directional element features, based on the idea of non-linear matching of the directions of the patterns [56], was originally proposed in [55], and has been applied in recognition systems of both machine-printed documents and handwritten documents in several languages [32, 33, 64]. The basic steps for computing the directional features from a character image are as follows:

1. Find the contour of the input image;



(a) Contour image and partition of bins.

(b) Four types of connected neighboring pixels.

Figure 2.1. Directional features of a character images.

2. divide the input image into overlap or non-overlap bins;

3. For each bin, trace the contour and count the four types of neighboring pixels shown in Figure 2.1 (horizontal, vertical, diagonal and back diagonal.) Hence, four features are computed from each bin.

When the dimensionality of the input features is too high, the handwriting recognition algorithm may either not be able to process the features or produce worse results. Thus we need to reduce the dimensionality of the feature space. Our goal is to keep relevant information for the recognition and eliminate the redundant information. This step is called feature extraction. Several methods can be applied to reduce the dimensionality, including the Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA). PCA is a linear orthogonal transform. The transform is a projection onto new coordinates so that new dimensions are not correlated and the variances of all the new dimensions are sorted in decreasing order.

PCA provides the optimal compression of energy by minimizing the mean square error of approximating the data but does not necessarily produce the best dimensions for classification. For example, if we apply PCA to the images of letters "O" and "Q"

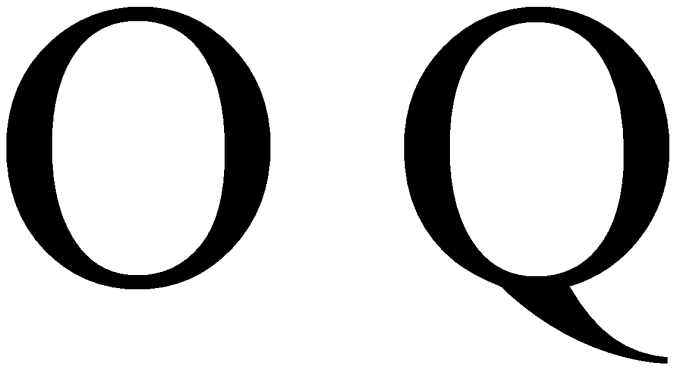


Figure 2.2. Letter "O" and "Q". PCA may only extract common features of the images of "O" and "Q", but LDA may extract the difference between them

shown in Figure 2.2, the extracted features may only retain the shape information that is common to both letters. This is because the overall shapes of the two letters look very similar, and PCA only keeps a rough shape that has most portion of energy but loses some detail such as the"tail" of letter "Q" that is useful for classification. In this case, LDA may be better than PCA. LDA is another linear transform of the feature space. Rather than finding optimal representation of the data, LDA is to find the best projection direction of features to maximize the ratio between the between-class variance and within-class variance. In this sense, LDA only selects the dimensions that show the diffence between classes but are consistent for the features of the same class. LDA is widely used in handwriting recognition systems, especially in HMM-based systems.

2.1.3 Language Modeling

A language model is the probability distribution of word sequences from the text of a corpus. The language model can be denoted by $\Pr(w_1, w_2, ...w_n)$ where $w_1, w_2, ...w_n$ are a sequence of n words. We usually assume that a language is an n-gram, *i.e.*, a $(n-1)$-th order Markov chain. For example, bi-gram

$$\Pr(w_1, w_2, ...w_n) = \Pr(w_1)\Pr(w_2|w_1)\Pr(w_3|w_2)...\Pr(w_n|w_{n-1}) \tag{2.2}$$

and tri-gram

$$\Pr(w_1, w_2, ...w_n) = \Pr(w_1)\Pr(w_2|w_1)\Pr(w_3|w_2, w_1)...\Pr(w_n|w_{n-1}, w_{n-2}). \tag{2.3}$$

The quality of describing some given text by a language model can be measured by perplexity

$$\mathcal{P}(\mathbf{w}) = [\Pr(w_1, w_2, ...w_n)]^{-\frac{1}{n}}, \tag{2.4}$$

where $\mathbf{w} = (w_1, w_2, ...w_n)$ is the text composed of a sequence of words $w_1, w_2, ...w_n$ and $\Pr(w_1, w_2, ...w_n)$ is the language model. Generally speaking, the smaller the perplexity, the better the language model describes the text.

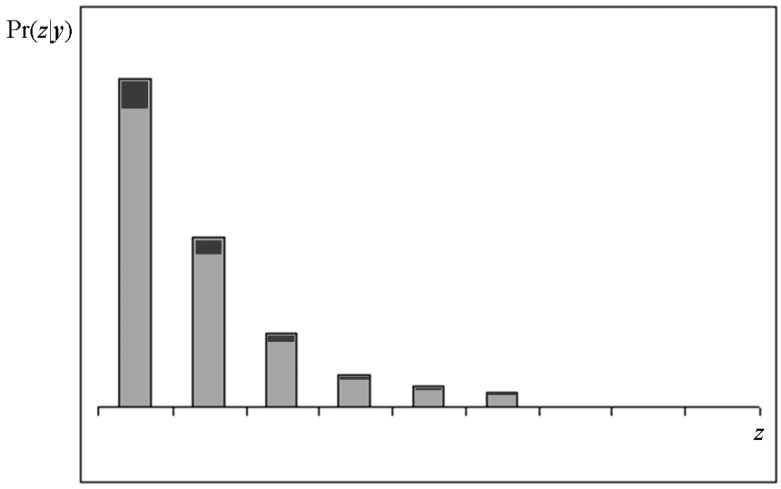
The language model can be obtained by counting the frequency of words from text. For tri-gram,

$$\Pr(w_k|w_{k-1}, w_{k-2}) = count(w_{k-2}w_{k-1}w_k)/count(w_{k-2}w_{k-1}) \tag{2.5}$$

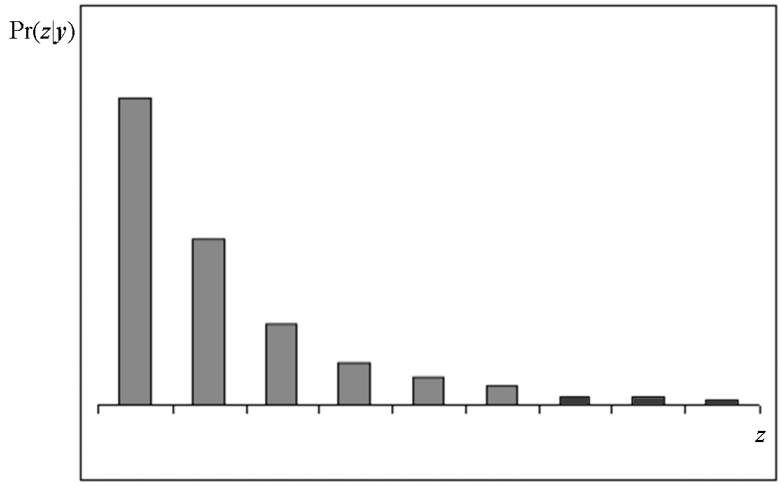
Or more generally, for any n-gram,

$$\Pr(z|\mathbf{y}) = \frac{count(\mathbf{y}z)}{\sum_w count(\mathbf{y}w)} = \frac{count(\mathbf{y}z)}{count(\mathbf{y})} \tag{2.6}$$

When the number of occurrences of $\mathbf{y}z$ is zero, the estimated probability density $\Pr(z|\mathbf{y})$ is also zero. This can lead to bad performance of the recognizer. We often use smoothing techniques to make sure probability density of the language model is non-zero throughout the term space of the n-gram. A simple but effective smoothing method is called "Backing Off" [52]. The basic idea is to reduce (discount) the amount of non-zero probabilities within the distribution of n-gram, and redistribute the discounted probability mass to those zero probabilities.



(a) Estimated n-gram (no smoothing).



(b) Smoothed n-gram.

Figure 2.3. Smoothing the language model by redistribution of the probability mass (backing off)

2.2 IR Techniques

Information retrieval (IR) is the science of searching for information in documents, searching for documents themselves, searching for metadata which describe documents, or searching relational or other databases. Document retrieval (searching for relevant documents) is most related to our work. We will introduce document retrieval techniques in this section.

Document retrieval is to search a collection of documents for those relevant to a certain query phrase. Three classic document retrieval algorithms or IR models: the Boolean model, the vector model, and the probabilistic model [1], although proposed decades before, are still very effective means of document retrieval. In the Boolean model, retrieval is based on whether or not the documents contain the query terms, whereas in both vector model and probabilistic model the relevance of a document is measured by a similarity between the document and the query and a rank is assigned to each document according to the degree of relevance. The most important things in all of the above classic IR models are the existence and number of occurrences of each query term in the document. In recent years, new methods for measuring the relevance of a document, such as the PageRank [36] were proposed.

2.3 Overview of Prior Works

2.3.1 Document Image Binarization

Recognition of low quality handwritten documents such as carbon forms is commonly considered as a very hard, or even impossible problem. This is largely due to the extremely low image quality. Usually the quality of a document image is affected by varying illumination and noise such as Gaussian noise, artifacts, smearing, and so on.

By assuming that the background changes slowly, the problem of varying illumination has been solved by several adaptive binarization algorithms. The algorithms for deciding either global or local thresholds ofr binarization were proposed in [47, 45, 53].

Although noise can be depressed by smoothing, the resulting blurring will also affect the OCR rate. Approaches based on heuristics, to name a few, Kamel/Zhao [30], Yang/Yan [60], and Milewski [42], solve the problem to some extent by heuristic search of stroke locations.The Kamel/Zhao algorithm is a local algorithm which finds stroke locations and then removes the noise in the non-stroke area using an interpolation and thresholding step. A parameter of stroke width is needed. The Yang/Yan algorithm is a variant of the method by Kamel/Zhao which is meant to handle varying intensity, illumination, and smearing. The Milewski algorithm is also a heuristic based method. It detects strokes from local statistics in different directions.

In recent years, inspired by the success of Markov Random Field (MRF) in the area of image restoration [18, 19, 20], some attempts were made to apply MRF to the preprocessing of textual region of degraded images [22, 23, 58]. The advantage of the MRF model over heuristics is that it can describe the probabilistic dependency of neighboring pixels or image patches, i.e., the prior probability, and learn it from training data. In other words, the spatial constraints between neighboring pixels are learned from training set of images instead of conceived heuristically.

In order to use MRF, one need to pick forms of prior and observation models. Usually this is done in ad hoc way. The forms of MRF's taken by all the existing approaches dealing with textual image are not very appropriate for handwritten document. The MRF based approach proposed by Wolf *et al.* [58] defined the prior model on a 4×4 clique and is appropriate for textual images in low resolution video. However, for 300 dpi high resolution handwritten document images, it is not feasible to learn the prior probability or energy potentials if we simply define a much larger neighborhood.

Gupta *et al.* [22, 23] proposed an algorithm for restoration and binarization of blurred images of license plate digits. Different from Wolf *et al.* [58], the vertices in statistical dependency graph of MRF represent image patches rather than pixels. The advantage of the patch based approach is the clique size is reduced and is much faster. They adopted the factorized form of MRF, *ie.*, the product of compatibility functions [18, 19, 20]. They defined compatibility functions as mixtures of multivariate normal distributions calculated over samples of their training set, and incorporated recognition into the MRF to reduce the number of samples involved in the calculation of compatibility functions. However this scheme can hardly be applied to unconstrained handwriting image because of the larger number of classes and the low performance of existing handwriting recognition algorithm.

Although MRF enlightened us to apply probabilistic neighborhood constraints to binarization, the computation is the biggest issue in all of the existing works. None of them solves the problem of high resolution handwritten document binarization. In Chapter 3, we will propose an MRF based binarization algorithm for handwritten document of resolution as high as 300dpi. In addition to binarization, with only a little mend, we apply our algorithm to form grid removal, a very important step of handwritten form analysis.

2.3.2 Handwritten Document Retrieval

Several works have been done to improve the IR performance of OCR'ed text. Researchers [17, 3] have shown that the performance of OCR text retrieval is badly affected when dealing with short or low quality documents. In [44, 46, 28] different approaches modeling typical recognition errors were proposed. In [44] a probabilistic model for misrecognition was proposed and this model was used to design the term-weighting scheme of information retrieval. The approach that generates candidate terms for each "true" search term and adds the retrieval results of candidate terms

into the final result was studied in [46]. In [28], a language model that took common recognition errors into account was built. This language model can then be used to approximate an "uncorrupted" version of a particular document, and it can be used for retrieval in a language modeling approach.

The problem of indexing and retrieving handwritten documents has recently been addressed by researchers. Due to low recognition accuracies, It is difficult to use probabilistic modeling of OCR'ed text for indexing and retrieving handwritten documents. A new trend in this research area is to index every word image with word recognition probabilities of all term candidates. Rath and Manmatha[49] proposed an OCR-free approach to historical manuscript retrieval that learned the joint probability of the query word and features of the word image. Assuming the independency of all terms in query q, the query-relevance probability

$$\Pr(q|d_j) \sim \prod_{t_i \in q} \Pr(t_i|d_j), \tag{2.7}$$

where q is the query, d_j is a document, and t_i's are terms. Let the term frequency $tf_{i,j}$ be the term-dependence probability, *i.e.*, $\Pr(t_i|d_j) = tf_{i,j}$. The term frequency is estimated by word recognition probabilities:

$$tf_{i,j} = \frac{1}{|d_j|} \sum_{o=1}^{|d_j|} \Pr(t_i|fv_o) \tag{2.8}$$

where fv_o runs over all feature vectors of word images in document d_j, $\Pr(t_i|fv_o)$ is the probability that the o-th word image is term t_i and is estimated from a labelled training set of word image features.

Two problems arise in the above probabilistic model. Firstly, probability estimate cannot be accurate when the dimensionality of feature vector increases. Secondly, the probabilistic model assumes that the term-relevance probability $\Pr(t_i|d_j)$ equals the terms frequency $tf_{i,j}$ which is not really accurate. Under this assumption, once any

term from the query phase occurs rarely in a document, *i.e.*, the term frequency is very small, the probability $\Pr(q|d_j)$ in factorized form will be close to zero.

A few works have attempted to solve this problem of probability estimate using OCR ranks [37, 24, 8]. Lee *et al.*[37] implemented retrieval on text composed of top-k candidates of character recognition results of Hangul document images. Howe *et al* used the same probabilistic model as in [49] except that the term-dependence probability is assumed to be inversely proportional to the word recognition rank of the term, namely,

$$Pr(t_i|fv_o) = \frac{Const}{rank(t_i)}. \tag{2.9}$$

In our previous work [8], we used the following formula to estimate the word recognition probability

$$Pr(t_i|fv_o) = \text{Top\_}R\text{\_Word\_Recognition\_Rate} - \text{Top\_}R\text{-1\_Word\_Recognition\_Rate}, \tag{2.10}$$

where $R = rank(t_i)$. The above rank-based probability estimate algorithm are still too simple to get the optimal results.

2.3.3 Keyword Spotting

Besides approaches to handwritten document retrieval, keyword retrieval, referred to as keyword spotting, as an alternative approach of indexing and retrieving handwritten documents has been proposed in [31, 38, 63]. The idea is to search the document for a certain keyword by feature matching instead of recognition. All existing methods [31, 38, 63] perform matching on feature space and require manual indexing of template images for query words.

- DTW based keyword spotting

In the Dynamic Time Warping (DTW) based method [31, 38], the following preprocessing steps are commonly performed.

1. Word segmentation is performed and the background of every word image is cleaned by removing irrelevant connected components from other words that reach into the word's bounding box.

2. Inter-word variations such as skew and slant angle are detected and eliminated.

3. The bounding box of any word image is cropped so that it tightly encloses the word.

4. The baseline of word images are normalized to a fixed position by padding extra rows to the images.

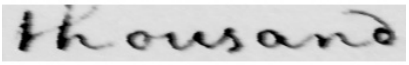
A normalized word image is represented by a multivariate time series composed of features from each column of the word image. These features include projection profile, upper/lower word profile, and number of background-to-foreground transitions.

1. Projection Profile. The projection profile of a word image is composed of the sums of foreground pixels in each columns.

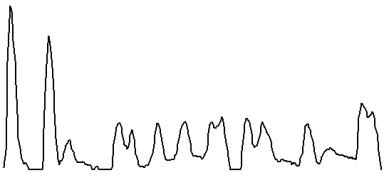
2. Upper/Lower Profiles. The upper profile of a word image is made of the distances from the upper boundary to the nearest foreground pixels in each column.

3. Background-to-Foreground Transitions. The number of background pixels whose right neighboring pixels are foreground pixels is taken as the number of background-to-foreground transitions of the column.

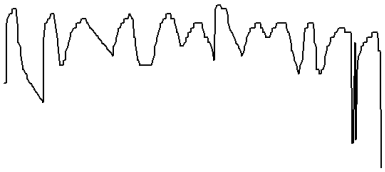
Figure 2.4 shows the four feature series of a word image from the handwriting data set of George Washington's manuscripts (CIIR, University of Massachusetts [31]).



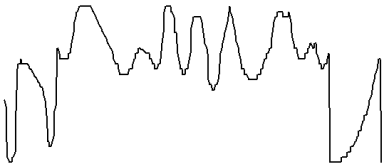
(a) A word image from George Washington's manuscripts.



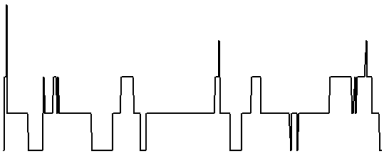
(b) Projection profile.



(c) Lower profile.



(d) Upper profile.



(e) Background-to-foreground transitions.

Figure 2.4. The feature series used in DTW word spotting.

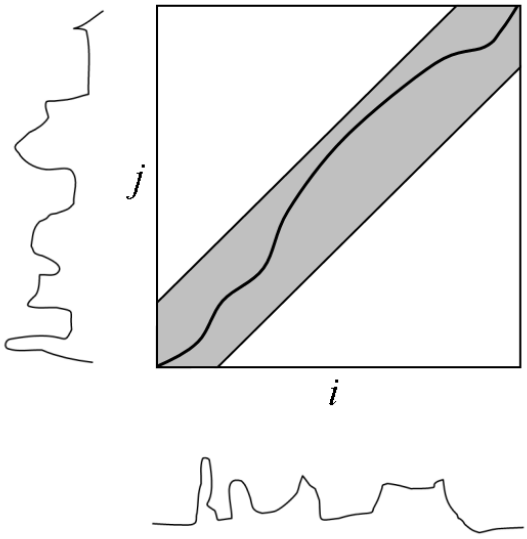


Figure 2.5. Sakoe-Chiba band.

Suppose two word images w_A and w_B are represented by $\{f_A(1), f_A(2), ..., f_A(l_A)\}$ and $\{f_B(1), f_B(2), ..., f_B(l_B)\}$, respectively, where $f_A(i)$ is the feature vector of the i-th column of image w_A, $f_B(j)$ is the feature vector of the j-th column of image w_B, and l_A and l_B are the lengths of w_A, w_B, respectively. Then the DTW matching of w_A and w_B is given by the recurrence equation

$$DTW(i,j) = \min \left\{ \begin{array}{l} DTW(i-1,j) \\ DTW(i-1,j-1) \\ DTW(i,j-1) \end{array} \right\} + d(i,j) \tag{2.11}$$

where $d(i,j)$ is the square of the Euclidean distance between $f_A(i)$ and $f_B(j)$.

The time complexity of the DTW algorithm is in $O(l_A \cdot l_B)$. In order to speed up the computation, a global path constraint like the Sakoe-Chiba band (Figure 2.3.3 can be applied to force the paths to stay close to the diagonal of the DTW matrix. Another advantage of the path constraint is to prevent pathological warping.

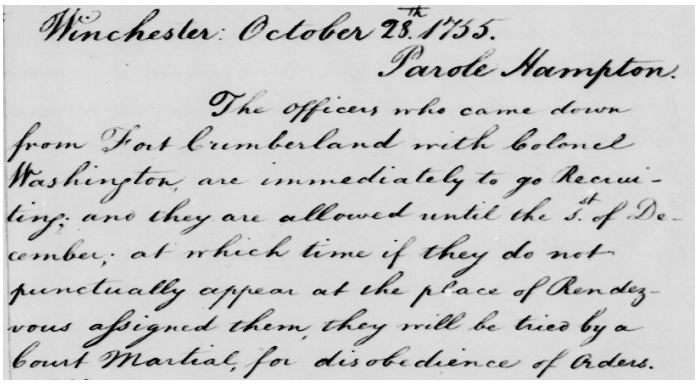


Figure 2.6. A sample from George Washington's manuscripts.

The matching error of $f_A(i)$ and $f_B(j)$ is given by $\frac{1}{l}DTW(l_A, l_B)$ where l is length of the warping path recovered by DTW. The word images are ranked in the increasing order of the matching errors to the template image.

The DTW based method has been tested on George Washington's manuscripts (Figure 2.3.3). The performance of keyword spotting was evaluated using the Mean Average Precision measure [1]:

1. For each query, check the returned word images starting from rank1. Whenever a relevant word image is found, record the precision of the word images from the one with rank 1 to the current one. The average value of the recorded precisions for the query is taken as the Average Precision of the query.

2. The mean value of the Average Precisions of all of the queries is the Mean Average Precision of the test.

A Mean Average Precision of 40.98% on 2372 word images of good quality and a Mean Average Precision of 16.50% on 3262 word images of poor quality were reported [38].

- GSC feature based keyword spotting

In the GSC feature based method [63], a word image is represented by GSC features that consist of 512 bits corresponding to gradient (192 bits), structural (192 bits) and concavity (128 bits) features. A word image is divided into 32 regions (8×4) and 16 binary GSC features are extracted from each region. The gradient features are obtained by thresholding the results of Sobel edge detection in the 12 directions. The structural features consist of the presence of corners, diagonal lines, and vertical and horizontal lines in the gradient image, as determined by the 12 rules. The concavity features include direction of bays, presence of holes, and large vertical and horizontal strokes.

The similarity of two word images is measured by the bitwise matching of the respective GSC feature vectors of the two images. The dissimilarity of two GSC feature vectors X and Y is defined as

$$D(X,Y) = \frac{1}{2} - \frac{S_{11}S_{00} - S_{10}S_{01}}{2\sqrt{(S_{10}+S_{11})(S_{01}+S_{00})(S_{11}+S_{01})(S_{00}+S_{10})}} \tag{2.12}$$

where S_{00}, S_{01}, S_{10}, and S_{11} are the numbers of 0-to-0, 0-to-1, 1-to-0, and 1-to-1 matches from X to Y, respectively. For example, the numbers of 0-to-0, 0-to-1, 1-to-0, and 1-to-1 matches between "0110110" and "0101001" are 1, 2, 3, and 1, respectively.

The GSC method has been tested on 9312 word images of 4 words ("been", "Cohen", "Medical", and "referred") written by 776 individuals. Each word was written three times by each individual. One of the three word images for every word written by any person is taken as a query template, and the remaining are taken for test. The performance of keyword spotting is evaluated by the recall and precision at different number of top matches. When the number of top matches of a query equals the number of relevant images, the recall value equals the precision value and is referred to as R-Precision.

The reults of both the GSC based method and the DTW based method are reported in [63]. The R-precision values of the above four queries using the GSC based method are 45.45%, 56.59%, 54.11%, and 62.04%, respectively. The R-precision values of the above four queries using the DTW based method are 35.53%, 38.65%, 44.39%, and 55.23%, respectively. Although the above results are obtained from a data set of multiple writers, the size of the lexicon is very small (containing only 4 words) and therefore the data set is not truly unconstrained.

The keyword spotting algorithms mentioned have at least three problems:

1. Matching based algorithms take a template image as input so manual indexing of a small portion of handwritings is required.

2. Features may vary a lot between writers even for the same word.

3. Existing keyword spotting algorithms assume no segmentation error. However this is not the case in real applications.

CHAPTER 3

HANDWRITTEN IMAGE BINARIZATION BASED ON MARKOV RANDOM FIELDS

3.1 Introduction

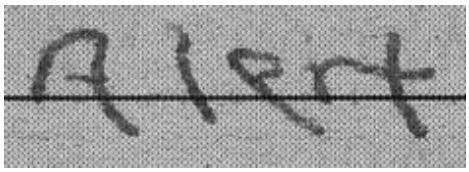
The goal of this chapter is preprocessing of degraded handwritten document images, such as carbon forms, for subsequent recognition and retrieval. Carbon form recognition is generally considered to be very hard, or even an impossible problem. This is largely due to the extremely low image quality. Although the background variation is not very intense, the handwriting is often occluded by extreme noise from two sources: (i) the extra carbon powder imprinted on the form because of accidental pressure and (ii) the inconsistent force of writing. For example, people tend to write lightly at the turns of strokes. This is not a serious problem for writing on regular paper. However, when writing on carbon paper, the light writing causes notches along the stroke. Furthermore, most carbon forms have a colored background which results in very low contrast and signal-to-noise ratio. Thus, the image quality of carbon copies is generally poorer than that of non-carbon copy degraded documents. Therefore the task of binarizing the carbon copy documents with handwritten data is very challenging.

Traditional document image binarization algorithms [47] [45] [53] [30][60] separate the foreground from the background by histogram thresholding and analysis of the connectivity of strokes. These algorithms, although effective, rely on heuristic rules of spatial constraints which are not scalable across applications. Recent research [22] [23] [58] has applied Markov Random Field (MRF) based methods to

document image binarization. Although these algorithms make various assumptions applicable only to low resolution document images, we take advantage of the ability of the MRF to model spatial constraints in the case of high resolution handwritten documents.

We present a method that uses a collection of standard patches to represent each patch of the binarized image from the test set. These representatives are obtained by clustering patches of binarized images in the training set. The use of representatives reduces the domain of the prior model to a manageable size. Since our objective is not image restoration (from linear or non-linear degradation), we do not need an image/scene pair for learning the observation model. We can learn the observation model on-the-fly from the local histogram of the test image. Therefore our algorithm achieves performance similar to adaptive thresholding algorithms [45, 53] even without using the prior model. Of course the result improves with the inclusion of spatial constraints added by the prior model. In addition to binarization, we also apply our MRF based algorithm to the removal of form lines by modeling the way the probabilistic density of the observation model is computed.

One significant improvement made in [10] since our prior works [11] is the use of a more reliable method of estimating the observational model. It is based on mathematical morphological operations to obtain the background and Gaussian Mixture Modeling to estimate the foreground and background probability densities. Another improvement is the use of more efficient pruning methods to reduce the search space of the MRF effectively by identifying the patches that are surrounded by background patches. We present experimental results on the PCR (Pre-Hospital Care Report) dataset of handwritten carbon forms [43] and provide quantitative comparison of word recognition rates on forms binarized by our method versus other heuristic approaches.



(a)

(b)

(c)

Figure 3.1. Stroke preserving line removal. (a) A word image with an underline across the text. (b) Binarized image with the underline removed. (c) Binarized image with the underline removed and fixed.

3.2 Related Work

3.2.1 Locally Adaptive Methods for Binarization

Usually the quality of a document image is affected by the varying illumination and noise. By assuming that the background changes slowly, the problem of varying illumination is solved by adaptive binarization algorithms such as Niblack [45] and Sauvola [53]. The idea is to determine the threshold locally, using histogram analysis, statistical measures (mean, variance, *etc.*), or the lightness of the extracted background. Although noise can be reduced by smoothing, the resulting blurring affects the OCR rate. Approaches based on heuristic analysis of local connectivity, such as Kamel/Zhao [30], Yang/Yan [60], and Milewski [43], solve the problem to some extent by searching for stroke locations and targeting only the non-stroke area. The Kamel/Zhao algorithm locates strokes using stroke width and then removes the noise in the non-stroke area using an interpolation and thresholding step. The Yang/Yan algorithm is just a variant of the same method. The Milewski algorithm examines

neighboring blocks in orientations to search for non-stroke area. However, all these approaches are heuristic whereas our objective is to develop a non-heuristic method.

3.2.2 Markov Random Field Based Approach to Binarization

In recent years, inspired by the success of the Markov Random Field (MRF) based approach in the area of image restoration [18], [19], [20], attempts have been made to apply MRF to preprocessing of degraded document images [22], [23], [58]. The advantage of the MRF model over heuristic methods is that it allow us to describe the dependency of neighboring pixels as the prior probability, and learn it from training data. Wolf *et al.* [58] defined the prior model on a 4×4 clique which is appropriate for textual images in low resolution video. However, for 300 dpi high resolution handwritten document images, it is not computationally feasible to learn the potentials if we simply define a much larger neighborhood. Gupta *et al.* [22], [23] studied restoration and binarization of blurred images of license plate digits. They adopted the factorized style of MRF using the product of compatibility functions [18], [19], [20] which are defined as mixtures of multivariate normal distributions computed over samples of the training set. They incorporated recognition into the MRF to reduce the number of samples involved in the calculation of the compatibility functions. However this scheme also can not be directly applied to unconstrained handwriting because of the larger number of classes and the low performance of existing handwriting recognition algorithms. We will describe a MRF adapted for handling handwritten documents that will overcome the challenges of computational complexity caused by high resolution data and low accuracy rates of current handwriting recognizers.

3.2.3 Form Grid Removal

The process of removing pre-printed form grids while preserving the overlapping textual matter is referred to as image in-painting (Figure 3.1) and is performed by inferring the removed overlapping portion of images from spatial constraints. MRF is

ideally suited for this task and has been used successfully on natural scene images ([5], [61]). Our task on document images is similar but more difficult: both of them use spatial constraints to paint in the missing pixels but the missing portions in document images often contain strokes with high frequency components and details. Previously reported work on line removal in document images are heuristic [2] [43] [62]. Bai *et al.* [2] remove the underline in machine-printed documents by estimating its width. It works on machine-printed documents because the number of possible situations in which strokes and underlines intersect is limited. Milewski *et al.* [43] proposed to restore the strokes of handwritten forms using a simple interpolation of neighboring pixels. Yoo *et al.* [62] describe a sophisticated method which classifies the missing parts of the strokes into different categories such as horizontal, vertical, and diagonal, and connects them with runs (of black pixels) in the corresponding directions. It relies on many heuristic rules and is not accurate when strokes are lightly (tangentially) touching the grid.

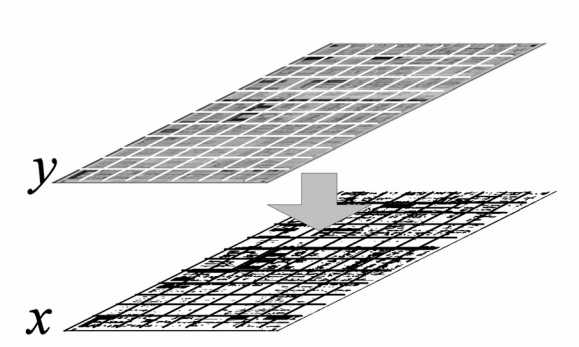
3.3 Markov Random Field model for handwritting images

We use a MRF model (Figure 5.3) with the same topology as the one described in [19]. A binarized image x is divided into non-overlapping square patches $x_1, x_2, ..., x_N$, and the input image, or the observation y is also divided into patches $y_1, y_2, ..., y_N$ so that x_i corresponds to y_i for any $1 \leq i \leq N$. Each binarized patch solely depends on its four neighboring binarized patches in both horizontal and vertical directions, and each observed patch solely depends on its corresponding binarized patch. Thus,

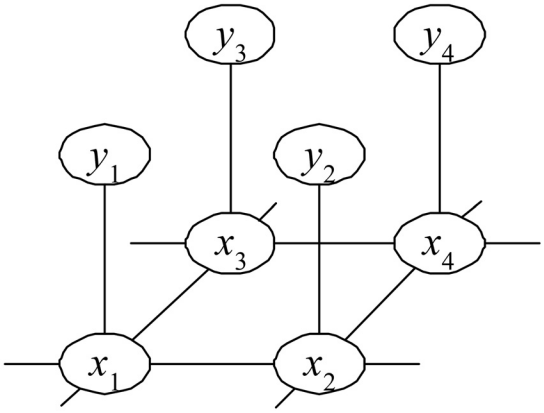
$$\Pr(x_i|x_1, ..., x_{i-1}, x_{i+1}, ..., x_N, y_1, ...y_N) = \Pr(x_i|x_{n_1,i}, x_{n_2,i}, x_{n_3,i}, x_{n_4,i}), 1 \leq i \leq N, \tag{3.1}$$

where $x_{n_1,i}$-$x_{n_4,i}$ are the four neighboring vertices of x_i, and

$$\Pr(y_i|x_1, ..., x_N, y_1, ..., y_{i-1}, y_{i+1}, ..., y_N) = \Pr(y_i|x_i), 1 \leq i \leq N \tag{3.2}$$



(a) The Input Image y and the Inferred Image x.



(b) Markov Network.

Figure 3.2. The topology of the Markov network. (a) the input image y and the Inferred Image x; (b) the Markov network generalized from (a). In (b) each node x_i in the field is connected to its four neighbors. Each observation node y_i is connected to node x_i. An edge indicates the statistical dependency of two nodes.

An edge in the graph represents the dependency of two vertices. The advantage of such a patch based structure is that relatively large areas of the local image are statistically dependent. Our objective is to estimate the binarized image x from the posterior probability $\Pr(x|y) = \dfrac{\Pr(x,y)}{\Pr(y)}$. Since $\Pr(y)$ is a constant over x, we only need to estimate x from the joint probability $\Pr(x,y) = \Pr(x_1, ..., x_N, y_1, ..., y_N)$. This can be done by either the MMSE or MAP approaches [18, 19]. In the MMSE approach, the estimation of each x_j is obtained by computing the marginal probability,

$$\hat{x}_{jMMSE} = \sum_{x_j} x_j \times \sum_{x_1..x_{j-1}x_{j+1}...x_N} \Pr(x_1, ..., x_N, y_1, ..., y_N) \tag{3.3}$$

In the MAP approach, the estimation of each x_j is obtained by taking the maximum of the probability,

$$\hat{x}_{jMAP} = \underset{x_j}{\operatorname{argmax}} \max_{x_1...x_{j-1}x_{j+1}...x_N} \Pr(x_1, ..., x_N, y_1, ..., y_N) \tag{3.4}$$

Estimation of the hidden vertices $\{x_j\}$ using Equation (3.3) or (3.4) is referred to as inference. It is impossible to compute either Equation (3.3) or (3.4) directly for large graphs because the computation grows exponentially as the number of vertices increases. We can use the belief propagation algorithm (BP) [48] to approximately compute the MMSE or MAP estimation in linear time (in the number of vertices in the graph).

3.4 Inference in the MRF Using Belief Propagation

3.4.1 Belief Propagation

In the Belief Propagation algorithm, the joint probability of the hidden image x and the observed image y from a Markov Random Field is approximated by the following factorized form [20, 19]

$$\Pr(x_1, ..., x_N, y_1, ..., y_N) = \prod_{(i,j)} \psi(x_i, x_j) \prod_k \phi(x_k, y_k) \tag{3.5}$$

where (i,j) are neighboring hidden nodes and ψ and ϕ are pairwise compatibility functions between neighboring nodes, learned from the training data. The MMSE and MAP objective functions can be rewritten as:

$$\hat{x}_{jMMSE} = \sum_{x_j} x_j \times \sum_{x_1..x_{j-1}x_{j+1}...x_N} \prod_{(i,j)} \psi(x_i, x_j) \prod_k \phi(x_k, y_k) \tag{3.6}$$

$$\hat{x}_{jMAP} = \operatorname*{argmax}_{x_j} \max_{x_1...x_{j-1}x_{j+1}...x_N} \prod_{(i,j)} \psi(x_i, x_j) \prod_k \phi(x_k, y_k) \tag{3.7}$$

The Belief propagation algorithm provides an approximate estimation of $\hat{x}_{jMMSE}$ or $\hat{x}_{jMAP}$ in Equations (3.6) and (3.7) by iterative steps. An iteration only involves local computation between the neighboring vertices. In the BP algorithm for MMSE, Equation (3.6) is approximately computed by two iterative equations:

$$\hat{x}_{jMMSE} = \sum_{x_j} x_j \phi(x_j, y_j) \prod_k M_j^k \tag{3.8}$$

$$M_j^k = \sum_{x_k} \psi(x_j, x_k) \phi(x_k, y_k) \prod_{l \neq j} \tilde{M}_k^l \tag{3.9}$$

In Equation (3.8), k runs over any of the four neighboring hidden vertices of x_j. M_j^k is the message passed from j to k and is calculated from Equation (3.9). $\tilde{M}_k^l$ is M_k^l from the previous iteration. The expression of M_j^k only involves the compatibility functions related to vertices j and k so M_j^k can be thought of as a message passed from vertex j to vertex k. Note that M_j^k is actually a function of x_j. Initially $M_j^k(x_j) = 1$ for any j and any value of x_j.

The formulas of the belief propagation algorithm for the MAP estimation are similar to Equations (3.8) and (3.9) except that $\sum_{x_j} x_j$ and $\sum_{x_k}$ are replaced with $\underset{x_j}{\text{argmax}}$ and $\max_{x_k}$, respectively:

$$\hat{x}_{jMAP} = \underset{x_j}{\text{argmax}} \phi(x_j, y_j) \prod_k M_j^k \tag{3.10}$$

$$M_j^k = \max_{x_k} \psi(x_j, x_k) \phi(x_k, y_k) \prod_{l \neq j} \tilde{M}_k^l \tag{3.11}$$

In our experiments, we use MAP estimation. The form of pairwise compatibility functions ψ and ϕ is usually heuristically selected as functions with the distance between two patches as the variable. We found that a simple form is not suitable for binarized images because the distance can only take a few values. Another way to select the form of ψ and ϕ is to use pairwise joint probabilities [18, 19]:

$$\psi(x_j, x_k) = \frac{\Pr(x_j, x_k)}{\Pr(x_j) \Pr(x_k)} \tag{3.12}$$

$$\phi(x_k, y_k) = \Pr(x_k, y_k) \tag{3.13}$$

Replacing the ψ and ϕ functions in Equations (3.10) and (3.11) with the definitions in Equations (3.12) and (3.13), we obtain

$$\hat{x}_{j\ \text{MAP}} = \underset{x_j}{\text{argmax}} \Pr(x_j) \Pr(y_j | x_j) \prod_k M_j^k, \tag{3.14}$$

and

$$M_j^k = \max_{x_k} \Pr(x_k | x_j) \Pr(y_k | x_k) \prod_{l \neq j} \tilde{M}_k^l, \tag{3.15}$$

In order to avoid overflow, we instead calculate the log values of the factors in Equations (3.14) and (3.15).

$$L_j^k = \max_{x_k} \left(\log \Pr(x_k | x_j) + \log \Pr(y_k | x_k) + \sum_{l \neq j} \tilde{L}_k^l \right), \tag{3.16}$$

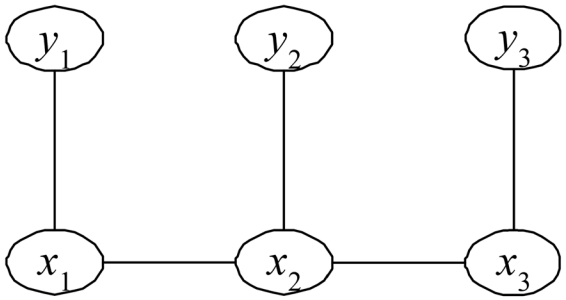


Figure 3.3. An acyclic Markov network.

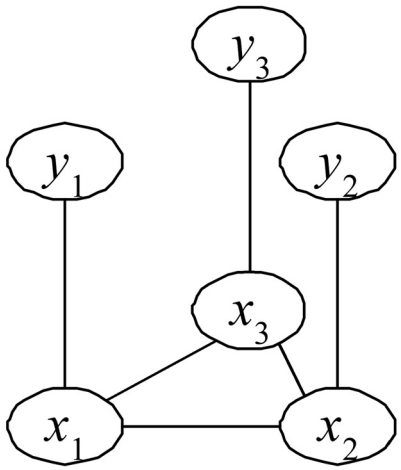


Figure 3.4. A cyclic Markov network.

$$\hat{x}_{j\,\mathrm{MAP}} = \underset{x_j}{\operatorname{argmax}} \left(\log \Pr(x_j) + \log \Pr(y_j|x_j) + \sum_k L_j^k \right), \tag{3.17}$$

where $L_j^k = \log M_j^k$, $\tilde{L}_k^l = \log \tilde{M}_k^l$, and the initial values of $\tilde{L}_j^k$'s are set to 0's.

3.4.2 An Example showing the BP Inference in the MRF

For a better understanding of the BP algorithm in Equations (3.14) and (3.15), let's consider the inference in the toy model in Figure 3.3. Here, suppose we use the MAP criteria. The MAP estimation of x_1 is given be the following equation:

$$\hat{x}_{1\,\mathrm{MAP}} = \underset{x_1}{\operatorname{argmax}} \max_{x_2,x_3} \Pr(x_1, x_2, x_3, y_1, y_2, y_3) \tag{3.18}$$

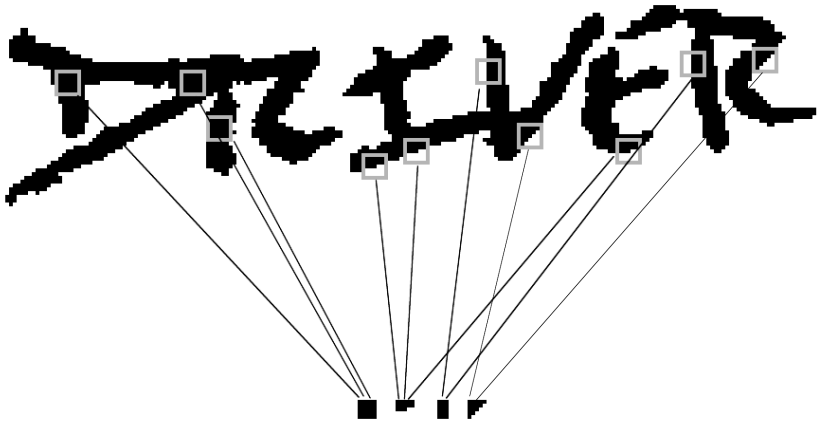


Figure 3.5. Shared patches in binary document image.

Using the Markov assumptions defined in the graph, we can get

$$\begin{aligned}\hat{x}_{1\,\mathrm{MAP}} =& \underset{x_1}{\operatorname{argmax}} \;\; \Pr(x_1)\Pr(y_1|x_1) \;\; \max_{x_2,x_3} \Pr(x_2|x_1)\Pr(x_3|x_2)\Pr(y_2|x_2)\Pr(y_3|x_3) \\ =& \underset{x_1}{\operatorname{argmax}} \;\; \Pr(x_1)\Pr(y_1|x_1) \;\; \max_{x_2} \Pr(x_2|x_1)\Pr(y_2|x_2) \;\; \max_{x_3} \Pr(x_3|x_2)\Pr(y_3|x_3)\end{aligned} \tag{3.19}$$

Similarly, we can also get the MAP estimation of x_2 and x_3:

$$\begin{aligned}\hat{x}_{2\,\mathrm{MAP}} =& \underset{x_2}{\operatorname{argmax}} \;\; \max_{x_1,x_3} \Pr(x_1,x_2,x_3,y_1,y_2,y_3) \\ =& \underset{x_2}{\operatorname{argmax}} \;\; \Pr(x_2)\Pr(y_2|x_2) \;\; \max_{x_1,x_3} \Pr(x_3|x_2)\Pr(x_1|x_2)\Pr(y_1|x_1)\Pr(y_3|x_3) \\ =& \underset{x_2}{\operatorname{argmax}} \;\; \Pr(x_2)\Pr(y_2|x_2) \;\; \max_{x_1} \Pr(x_1|x_2)\Pr(y_1|x_1) \;\; \max_{x_3} \Pr(x_3|x_2)\Pr(y_3|x_3)\end{aligned} \tag{3.20}$$

and

$$
\begin{aligned}
\hat{x}_{3\ \mathrm{MAP}} =& \underset{x_3}{\operatorname{argmax}} \ \max_{x_1,x_2} \Pr(x_1, x_2, x_3, y_1, y_2, y_3) \\
=& \underset{x_3}{\operatorname{argmax}} \ \Pr(x_3) \Pr(y_3|x_3) \ \max_{x_1,x_2} \Pr(x_2|x_3) \Pr(x_1|x_2) \Pr(y_1|x_1) \Pr(y_2|x_2) \\
=& \underset{x_3}{\operatorname{argmax}} \ \Pr(x_3) \Pr(y_3|x_3) \ \max_{x_2} \Pr(x_2|x_3) \Pr(y_2|x_2) \ \max_{x_1} \Pr(x_1|x_2) \Pr(y_1|x_1)
\end{aligned}
\tag{3.21}
$$

If we use Equations (3.14) and (3.15) for the inference, initially all the M_j^k's equal to 1. After the 1st iteration,

$$
\begin{aligned}
\hat{x}_{1\ \mathrm{MAP}} &= \underset{x_1}{\operatorname{argmax}} \Pr(x_1) \Pr(y_1|x_1) \\
\hat{x}_{2\ \mathrm{MAP}} &= \underset{x_2}{\operatorname{argmax}} \Pr(x_2) \Pr(y_2|x_2) \\
\hat{x}_{3\ \mathrm{MAP}} &= \underset{x_3}{\operatorname{argmax}} \Pr(x_3) \Pr(y_3|x_3) \\
M_1^2 &= \max_{x_2} \Pr(x_2|x_1) \Pr(y_2|x_2) \\
M_2^1 &= \max_{x_1} \Pr(x_1|x_2) \Pr(y_1|x_1) \\
M_2^3 &= \max_{x_3} \Pr(x_3|x_2) \Pr(y_3|x_3) \\
M_3^2 &= \max_{x_2} \Pr(x_2|x_3) \Pr(y_2|x_2)
\end{aligned}
\tag{3.22}
$$

After the 2nd iteration,

$$
\begin{aligned}
\hat{x}_{1\,\mathrm{MAP}} &= \underset{x_1}{\operatorname{argmax}} \Pr(x_1)\Pr(y_1|x_1)\max_{x_2}\Pr(x_2|x_1)\Pr(y_2|x_2) \\
\hat{x}_{2\,\mathrm{MAP}} &= \underset{x_2}{\operatorname{argmax}} \Pr(x_2)\Pr(y_2|x_2)\max_{x_1}\Pr(x_1|x_2)\Pr(y_1|x_1)\max_{x_3}\Pr(x_3|x_2)\Pr(y_3|x_3) \\
\hat{x}_{3\,\mathrm{MAP}} &= \underset{x_3}{\operatorname{argmax}} \Pr(x_3)\Pr(y_3|x_3)\max_{x_2}\Pr(x_2|x_3)\Pr(y_2|x_2) \\
M_1^2 &= \max_{x_2}\Pr(x_2|x_1)\Pr(y_2|x_2)\max_{x_3}\Pr(x_3|x_2)\Pr(y_3|x_3) \\
M_2^1 &= \max_{x_1}\Pr(x_1|x_2)\Pr(y_1|x_1) \\
M_2^3 &= \max_{x_3}\Pr(x_3|x_2)\Pr(y_3|x_3) \\
M_3^2 &= \max_{x_2}\Pr(x_2|x_3)\Pr(y_2|x_2)\max_{x_1}\Pr(x_1|x_2)\Pr(y_1|x_1)
\end{aligned}
\tag{3.23}
$$

After the 3rd iteration,

$$
\begin{aligned}
\hat{x}_{1\,\mathrm{MAP}} &= \underset{x_1}{\operatorname{argmax}} \Pr(x_1)\Pr(y_1|x_1)\max_{x_2}\Pr(x_2|x_1)\Pr(y_2|x_2)\max_{x_3}\Pr(x_3|x_2)\Pr(y_3|x_3) \\
\hat{x}_{2\,\mathrm{MAP}} &= \underset{x_2}{\operatorname{argmax}} \Pr(x_2)\Pr(y_2|x_2)\max_{x_1}\Pr(x_1|x_2)\Pr(y_1|x_1)\max_{x_3}\Pr(x_3|x_2)\Pr(y_3|x_3) \\
\hat{x}_{3\,\mathrm{MAP}} &= \underset{x_3}{\operatorname{argmax}} \Pr(x_3)\Pr(y_3|x_3)\max_{x_2}\Pr(x_2|x_3)\Pr(y_2|x_2)\max_{x_1}\Pr(x_1|x_2)\Pr(y_1|x_1)) \\
M_1^2 &= \max_{x_2}\Pr(x_2|x_1)\Pr(y_2|x_2)\max_{x_3}\Pr(x_3|x_2)\Pr(y_3|x_3) \\
M_2^1 &= \max_{x_1}\Pr(x_1|x_2)\Pr(y_1|x_1) \\
M_2^3 &= \max_{x_3}\Pr(x_3|x_2)\Pr(y_3|x_3) \\
M_3^2 &= \max_{x_2}\Pr(x_2|x_3)\Pr(y_2|x_2)\max_{x_1}\Pr(x_1|x_2)\Pr(y_1|x_1)
\end{aligned}
\tag{3.24}
$$

The BP algorithm converges after the 3rd iteration because the message obtained at the end of the 3rd iteration is the same of that obtained at the end of the 2nd iteration. The estimated values $\hat{x}_{1\,\mathrm{MAP}}$-$\hat{x}_{3\,\mathrm{MAP}}$ in Equation (3.27) are exactly the MAP estimation given by Equations (3.19)- (3.21).

It can be proved that, for any Markov network in a tree-like structure, BP can converge to the MAP (or MMSE) estimation in N iterations, where N is the number of vertices in the MRF. However, the above assertion is not true for Markov networks with loop(s). For example, we can verify the output of BP inference in three iterations on the cyclic Markov network shown in Figure 3.4.2 is NOT the MAP estimation of x_1-x_3 in the graph. If we use Equations (3.14) and (3.15) for the graph in Figure 3.4, initially all the M_j^k's equal to 1. After the 1st iteration,

$$\begin{aligned}
\hat{x}_{1\ \mathrm{MAP}} &= \underset{x_1}{\operatorname{argmax}} \Pr(x_1) \Pr(y_1|x_1) \\
\hat{x}_{2\ \mathrm{MAP}} &= \underset{x_2}{\operatorname{argmax}} \Pr(x_2) \Pr(y_2|x_2) \\
\hat{x}_{3\ \mathrm{MAP}} &= \underset{x_3}{\operatorname{argmax}} \Pr(x_3) \Pr(y_3|x_3) \\
M_1^2 &= \max_{x_2} \Pr(x_2|x_1) \Pr(y_2|x_2) \\
M_2^1 &= \max_{x_1} \Pr(x_1|x_2) \Pr(y_1|x_1) \\
M_1^3 &= \max_{x_3} \Pr(x_3|x_1) \Pr(y_3|x_3) \\
M_3^1 &= \max_{x_1} \Pr(x_1|x_3) \Pr(y_1|x_1) \\
M_2^3 &= \max_{x_3} \Pr(x_3|x_2) \Pr(y_3|x_3) \\
M_3^2 &= \max_{x_2} \Pr(x_2|x_3) \Pr(y_2|x_2)
\end{aligned} \tag{3.25}$$

After the 2nd iteration,

$$
\begin{aligned}
\hat{x}_{1\ \mathrm{MAP}} &= \operatorname*{argmax}_{x_1} \Pr(x_1)\Pr(y_1|x_1) \max_{x_2} \Pr(x_2|x_1)\Pr(y_2|x_2) \max_{x_3} \Pr(x_3|x_1)\Pr(y_3|x_3) \\
\hat{x}_{2\ \mathrm{MAP}} &= \operatorname*{argmax}_{x_2} \Pr(x_2)\Pr(y_2|x_2) \max_{x_1} \Pr(x_1|x_2)\Pr(y_1|x_1) \max_{x_3} \Pr(x_3|x_2)\Pr(y_3|x_3) \\
\hat{x}_{3\ \mathrm{MAP}} &= \operatorname*{argmax}_{x_3} \Pr(x_3)\Pr(y_3|x_3) \max_{x_2} \Pr(x_2|x_3)\Pr(y_2|x_2) \max_{x_1} \Pr(x_1|x_3)\Pr(y_1|x_1) \\
M_1^2 &= \max_{x_2} \Pr(x_2|x_1)\Pr(y_2|x_2) \max_{x_3} \Pr(x_3|x_2)\Pr(y_3|x_3) \\
M_2^1 &= \max_{x_1} \Pr(x_1|x_2)\Pr(y_1|x_1) \max_{x_3} \Pr(x_3|x_1)\Pr(y_3|x_3) \\
M_1^3 &= \max_{x_3} \Pr(x_3|x_1)\Pr(y_3|x_3) \max_{x_2} \Pr(x_2|x_3)\Pr(y_2|x_2) \\
M_3^1 &= \max_{x_1} \Pr(x_1|x_3)\Pr(y_1|x_1) \max_{x_2} \Pr(x_2|x_1)\Pr(y_2|x_2) \\
M_2^3 &= \max_{x_3} \Pr(x_3|x_2)\Pr(y_3|x_3) \max_{x_1} \Pr(x_1|x_3)\Pr(y_1|x_1) \\
M_3^2 &= \max_{x_2} \Pr(x_2|x_3)\Pr(y_2|x_2) \max_{x_1} \Pr(x_1|x_2)\Pr(y_1|x_1)
\end{aligned}
\tag{3.26}
$$

After the 3rd iteration,

$$
\begin{aligned}
\hat{x}_{1\,\mathrm{MAP}} =& \operatorname*{argmax}_{x_1} \Pr(x_1)\Pr(y_1|x_1)\cdot[\max_{x_2}\Pr(x_2|x_1)\Pr(y_2|x_2)\max_{x_3}\Pr(x_3|x_2)\Pr(y_3|x_3)]\cdot \\
& [\max_{x_3}\Pr(x_3|x_1)\Pr(y_3|x_3)\max_{x_2}\Pr(x_2|x_3)\Pr(y_2|x_2)] \\
\hat{x}_{2\,\mathrm{MAP}} =& \operatorname*{argmax}_{x_2} \Pr(x_2)\Pr(y_2|x_2)\cdot[\max_{x_1}\Pr(x_1|x_2)\Pr(y_1|x_1)\max_{x_3}\Pr(x_3|x_1)\Pr(y_3|x_3)]\cdot \\
& [\max_{x_3}\Pr(x_3|x_2)\Pr(y_3|x_3)\max_{x_1}\Pr(x_1|x_3)\Pr(y_1|x_1)] \\
\hat{x}_{3\,\mathrm{MAP}} =& \operatorname*{argmax}_{x_3} \Pr(x_3)\Pr(y_3|x_3)\cdot[\max_{x_1}\Pr(x_1|x_3)\Pr(y_1|x_1)\max_{x_2}\Pr(x_2|x_1)\Pr(y_2|x_2)]\cdot \\
& [\max_{x_2}\Pr(x_2|x_3)\Pr(y_2|x_2)\max_{x_1}\Pr(x_1|x_2)\Pr(y_1|x_1)] \\
M_1^2 =& \max_{x_2}\Pr(x_2|x_1)\Pr(y_2|x_2)\max_{x_3}\Pr(x_3|x_2)\Pr(y_3|x_3)\max_{x_1}\Pr(x_1|x_3)\Pr(y_1|x_1) \\
M_2^1 =& \max_{x_1}\Pr(x_1|x_2)\Pr(y_1|x_1)\max_{x_3}\Pr(x_3|x_1)\Pr(y_3|x_3)\max_{x_2}\Pr(x_2|x_3)\Pr(y_2|x_2) \\
M_1^3 =& \max_{x_3}\Pr(x_3|x_1)\Pr(y_3|x_3)\max_{x_2}\Pr(x_2|x_3)\Pr(y_2|x_2)\max_{x_1}\Pr(x_1|x_2)\Pr(y_1|x_1) \\
M_3^1 =& \max_{x_1}\Pr(x_1|x_3)\Pr(y_1|x_1)\max_{x_2}\Pr(x_2|x_1)\Pr(y_2|x_2)\max_{x_3}\Pr(x_3|x_2)\Pr(y_3|x_3) \\
M_2^3 =& \max_{x_3}\Pr(x_3|x_2)\Pr(y_3|x_3)\max_{x_1}\Pr(x_1|x_3)\Pr(y_1|x_1)\max_{x_2}\Pr(x_2|x_1)\Pr(y_2|x_2) \\
M_3^2 =& \max_{x_2}\Pr(x_2|x_3)\Pr(y_2|x_2)\max_{x_1}\Pr(x_1|x_2)\Pr(y_1|x_1)\max_{x_3}\Pr(x_3|x_1)\Pr(y_3|x_3)
\end{aligned}
\tag{3.27}
$$

The BP algorithm does not converge because the updated message at the end of the 2nd iteration is not equal to the one after the 3rd iteration. Even if we proceed with a few more iterations, we can verify that the BP still does not converge. In order to explain the inherent different between an acyclic graph (Figure 3.3) and a loopy graph (Figure 3.4), we may consider the MAP estimation of x_1 (see Equation (3.18)), for example. For a tree-like topology in Figure 3.3, the joint distribution of the MRF can be factorized using the Markov assumption as follows:

$$
\Pr(x_1, x_2, x_3) = \Pr(x_1)\Pr(x_2|x_1)\Pr(x_3|x2) \tag{3.28}
$$

Thus, the MAP estimation can also be factorized (repeating Equation (3.19)):

$$\begin{aligned}\hat{x}_{1\,\mathrm{MAP}} =& \underset{x_1}{\operatorname{argmax}} \;\; \Pr(x_1)\Pr(y_1|x_1) \;\; \max_{x_2,x_3} \Pr(x_2|x_1)\Pr(x_3|x_2)\Pr(y_2|x_2)\Pr(y_3|x_3) \\ =& \underset{x_1}{\operatorname{argmax}} \;\; \Pr(x_1)\Pr(y_1|x_1) \;\; \max_{x_2} \Pr(x_2|x_1)\Pr(y_2|x_2) \;\; \max_{x_3} \Pr(x_3|x_2)\Pr(y_3|x_3)\end{aligned} \tag{3.29}$$

Under the above factorization, $\max_{x_3} \Pr(x_3|x_2)\Pr(y_3|x_3)$ is a (single-variant!) function of x_2, and $\max_{x_2} \Pr(x_2|x_1)\Pr(y_2|x_2)\max_{x_3} \Pr(x_3|x_2)\Pr(y_3|x_3)$ is also a single-variant function, *i.e.*, a function of x_1. This ensures the time complexity of computing the maximum does not increase when we proceed to an outer-level. The BP algorithm is simply a faster algorithm of the above factorization that computes duplicated components only once for all the vertices. As we know, BP $\in O(N)$ but the above factorization $\in O(N^2)$.

For a graph with loop(s), unfortunately, the above factorization does not exist since the dependencies between vertices starting from a vertex can propagate back to itself through the loop(s). Although we can run the same BP algorithm on the loopy graph, it will not converge to the true MAP (or MMSE) estimation.

Although the BP algorithm is not exact on loopy Markov networks, in several applications of image restoration, it has been proved empirically to produce excellent estimations [18], [19]. We will use the sub-optimal results given by the BP algorithm, and rely on the experimental results.

3.4.3 Learning the Prior Model $\Pr(x_j)$ and $\Pr(x_k|x_j)$

To use Equations (3.14) and (3.15), the probabilities $\Pr(x_j)$, $\Pr(x_k|x_j)$ (prior model) and the observational probabilistic densities $\Pr(y_j|x_j)$, $\Pr(y_k|x_k)$ (observational model) have to be estimated. The prior probabilities $\Pr(x_j)$ and $\Pr(x_k|x_j)$ are learned from a training set of clean handwriting images. The training set contains three high quality binarized handwriting images from different writers. We can extract about two million patch images from these samples. Some samples from the

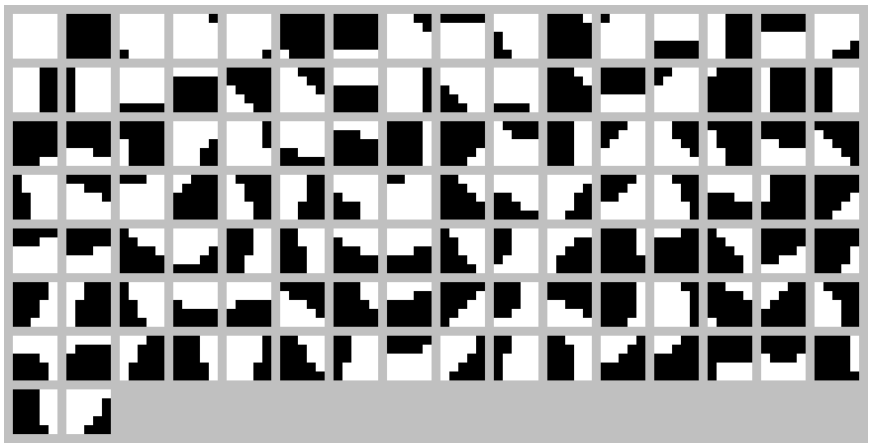


Figure 3.6. 114 representatives of shared patches obtained from clustering.

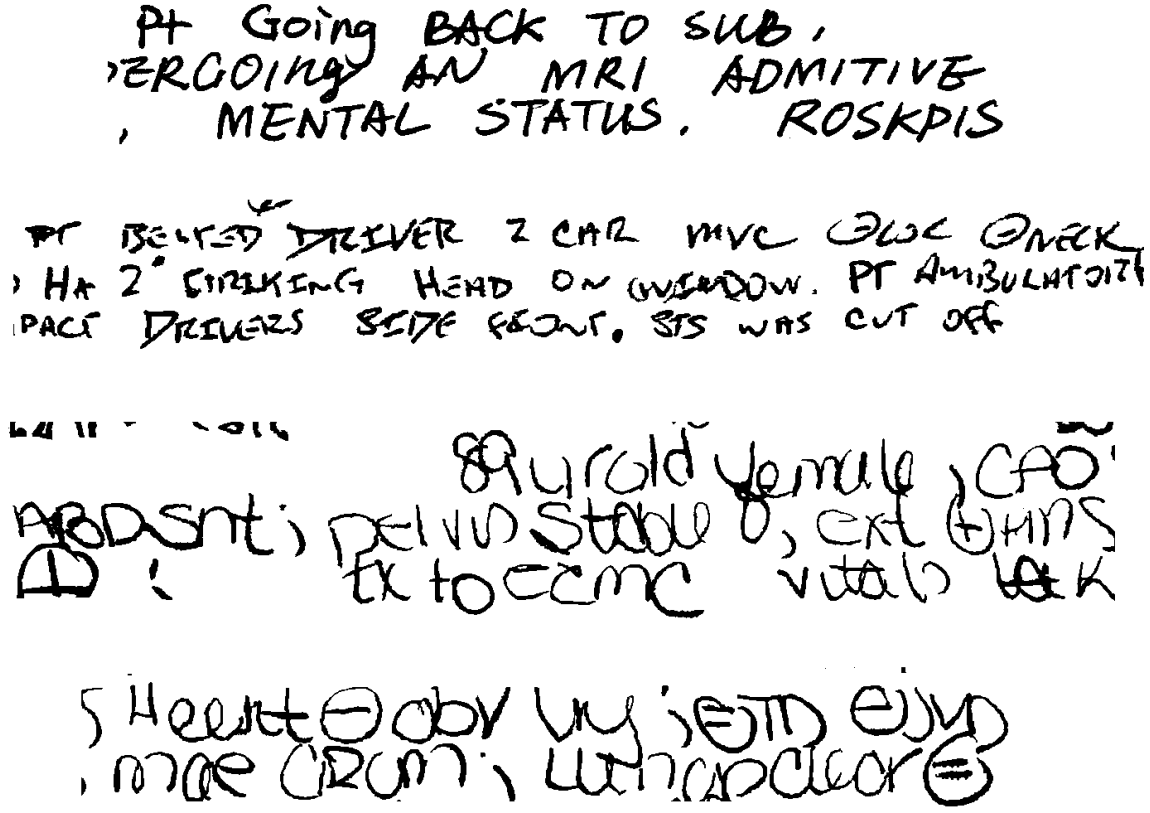


Figure 3.7. Binarized images from three writers for learning the prior model.

training set are shown in Figure 3.7. For training we use clean samples because unlike the observed image, the hidden image should have good quality.

Assuming the size of a patch is $B \times B$, the number of states of a binarized patch x_j is 2^{B^2}. If B=5, for example, there will be about 34 million states. This makes the computation of searching for the maximum in Equation (3.15) intractable. In order to solve this problem, we convert the original set of states to a much smaller set, and then estimate the probabilities over the smaller set of states. Usually this can be done by dimension reduction using transforms like PCA. But it is hard to apply such a transform to binarized images. Therefore we use a number of standard patches to represent all of the 2^{B^2} states. This is similar to vector quantization (VQ) used in data compression. The set of representatives is referred to as the codebook of VQ. Our method is inspired by the idea that images of similar objects can be represented by a very small number of the shared patches in the spatial domain. Recently, Jolic *et al.* [29] explored this possibility of representing an image by shared patches. Similarly the binarized document images with handwriting of fixed pen-width under the same resolution can also be decomposed into patches that appear frequently (Figure 3.5). The representatives are learned by clustering all the patches in our training set. We use the following approach. After every iteration of k-means clustering, we round all the dimensions of each cluster center to 0 or 1. Given a training set of $B \times B$ binary patches, represented by $\{p_i\}$, we run the k-means clustering with initial number of clusters = 1024, and remove the duplicated clusters and clusters containing less than 1000 samples. The remaining cluster centers are taken as the representatives.

If the codebook is denoted by $\widetilde{C}$=$\{C_1, C_2, ..., C_M\}$ where $C_1, ..., C_M$ are M representatives, the error of vector quantization is given by the following equation

$$\epsilon_{vq} = \frac{\sum_i [d(p_i, \widetilde{C})]^2}{\#\{p_i\} \cdot B^2} \tag{3.30}$$

where $d(p_i, \widetilde{C})$ denotes the Euclidian distance from p_i to its nearest neighbor(s) in $\widetilde{C}$, and $\#\{p_i\}$ denotes the number of elements in $\{p_i\}$. ϵ_{vq} is the square error normalized by the total number of pixels in the training set.

We can use the quantization error ϵ_{vq} to determine the parameter B. A larger patch size provides stronger local dependency but it is non-trivial to represent very large patches because of the variety of writing styles exhibited by different writers. We tried different values of B ranging between 5 and 8 which coincide with the range of stroke width in 300dpi handwriting images, and chose the largest value of B that led to an ϵ_{vq} that is below 0.01. Thus, we determined the patch size $B = 5$. Then the representation error $\epsilon_{vq} = 0.0079$ and 114 representatives are generated (Figure 3.6). The size of the search space of a binarized patch is reduced from 2^{5^2} (about 34 million) to 114.

Now we can estimate the prior probability $\Pr(x_j)$ over codebook $\widetilde{C}$.

$$\sum_{l=1}^{M} \Pr(x_j = C_l) = 1 \tag{3.31}$$

so that the prior probabilities $\Pr(x_j)$ over the reduced search space must add up to 1. We estimate $\Pr(x_j)$ from the relative size of the cluster centered at C_l. A patch p_i from the training set is a member of cluster C_l ($1 \leq l \leq M$) if C_l is a nearest neighbor of p_i among all of $C_1, ..., C_M$, and is denoted by $p_i \in C_l$. Note that a patch p_i from the training set may have multiple nearest neighbors among $C_1, ..., C_M$. The number of nearest neighbors of p_i in $\widetilde{C}$ is denoted by $n_{\widetilde{C}}(p_i)$. Thus the probability $\Pr(x_j)$ is estimated by

$$\hat{\Pr}(x_j = C_l) = \frac{\sum_{p_i \in C_l} \frac{1}{n_{\widetilde{C}}(p_i)}}{\#\{p_i\}}, \quad l = 1, 2, ..., M \tag{3.32}$$

where $\#\{p_i\}$ is the number of patches in $\{p_i\}$. $\hat{\Pr}(x_j = C_l)$ in Equation (3.32) is estimated by the size of cluster C_l normalized by the total number of training patches. It is easy to verify that the probabilities in Equation (3.32) add up to 1.

$\Pr(x_j, x_k)$ are estimated in the horizontal and vertical directions, respectively. Similar to Equation (3.32), $\Pr(x_j, x_k)$ $(x_j, x_k \in \widetilde{C})$ in horizontal direction is estimated by

$$\hat{\Pr}(x_j = C_{l_1}, x_k = C_{l_2}) = \frac{\sum_{(p_{i_1}, p_{i_2}), p_{i_1} \in C_{l_1}, p_{i_2} \in C_{l_2}} \frac{1}{n_{\widetilde{C}}(p_{i_1}) \cdot n_{\widetilde{C}}(p_{i_2})}}{\#\{(p_{i_1}, p_{i_2})\}}, l_1 = 1, 2, ..., M; \quad l_2 = 1, 2, ..., M \tag{3.33}$$

where (p_{i_1}, p_{i_2}) runs for all pairs of patches in the training set $\{p_i\}$ such that p_{i_1} is the left neighbor of p_{i_2} and $\#\{(p_{i_1}, p_{i_2})\}$ is the number of pairs of left-and-right neighboring patches in $\{p_i\}$.

$\Pr(x_j, x_k)$ $(x_j, x_k \in \widetilde{C})$ in vertical direction is estimated by an equation similar to Equation (3.33) except that p_{i1} is the top neighbor of p_{i2}.

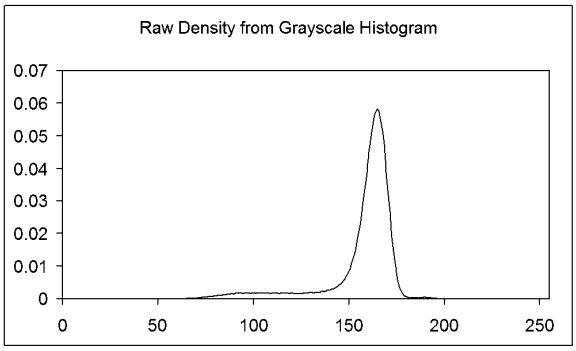
3.4.4 Learning the Observational Model $\Pr(y_j|x_j)$

For the observational model of a single pixel we can use the histogram based model generalized in [58]. For a patch based observational model, we need to map the single-pixel version to the vector space of patches. The pixels of an observed patch y_j are denoted by $y_j^{r,s}$, $1 \leq r, s \leq 5$. The pixels of a binarized patch x_j are denoted by $x_j^{r,s}$, $1 \leq r, s \leq 5$. We assume that the pixels inside an observed patch y_j and the respective binarized patch x_j obey similar dependence assumption as the patches in the patch-based topology (Equation (3.2)), *i.e.*,

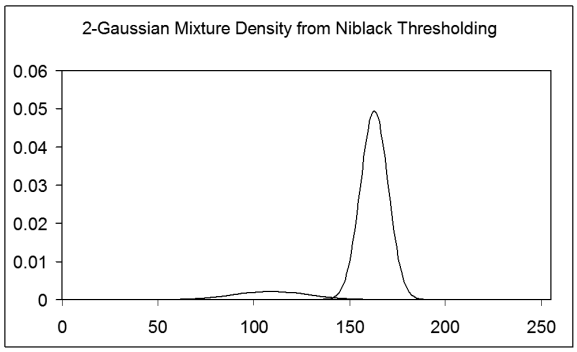
$$\Pr(y_j^{r,s}|y_j^{1,1}, ..., y_j^{r,s-1}, y_j^{r,s+1}, ..., y_j^{5,5}, x_j^{1,1}, ...x_j^{5,5}) = \Pr(y_j^{r,s}|x_j^{r,s}), \quad 1 \leq r, s \leq 5 \tag{3.34}$$

Thus it can be proved that

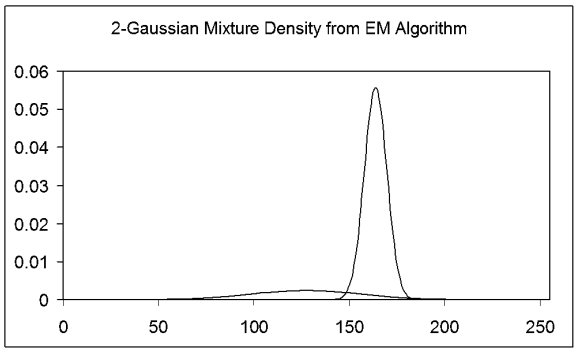
$$\Pr(y_j^{1,1}, ..., y_j^{5,5}|x_j^{1,1}, ...x_j^{5,5}) = \prod_{r=1}^{5}\prod_{s=1}^{5} \Pr(y_j^{r,s}|x_j^{r,s}) \tag{3.35}$$



(a) Raw Density by smoothing the Gray-scale Histogram of the image in Figure 3.9.



(b) 2-Gaussian Mixture Density from Thresholding.



(c) 2-Gaussian Mixture Density from EM Algorithm.

Figure 3.8. The smoothed gray-scale histogram and estimated foreground and background p.d.f. using two methods. Thresholding based method did not perform well at the intersection of two density functions, whereas EM algorithm based method improved the result.

Given the distribution of the lightness of foreground (strokes) $p_f(y_j^{r,s}) = \Pr(y_j^{r,s}|x_j^{r,s} = 0)$ and the distribution of the lightness of background $p_b(y_j^{r,s}) = \Pr(y_j^{r,s}|x_j^{r,s} = 1)$, according to Equation (3.35), the conditional p.d.f $\Pr(y_j|x_j)$ is calculated as

$$\Pr(y_j|x_j) = \prod_{1\leq r,s\leq 5, x_j^{r,s}=0} p_f(y_j^{r,s}) \prod_{1\leq r,s\leq 5, x_j^{r,s}=1} p_b(y_j^{r,s}) \tag{3.36}$$

The expression $1 \leq r, s \leq 5, x_j^{r,s} = 0$ means that the scope of the product is for any r and s such that $1 \leq r, s \leq 5$ and $x_j^{r,s} = 0$. The expression $1 \leq r, s \leq 5, x_j^{r,s} = 1$ is specified in the same way.

The probability densities p_f and p_b change over an image while the lightness of the background is changing. However, it is not a problem as we can use background regularization techniques such as the Background Surface Thresholding (BST) [54] to obtain the background and normalize the images. The background mapping technique is equivalent to adaptive thresholding algorithms such as the Niblack algorithm [45].

Learning the p.d.f. p_f and p_b is unsupervised. Assuming that p_f and p_b are two normal distributions, one way to compute p_f and p_b is as follows. First we determine a threshold T by an adaptive thresholding method such as the Niblack algorithm. Then we use all the pixels with gray-level $\leq T$ to estimate the mean and variance of p_f, and use the remaining pixels to estimate the mean and variance of p_b. This method to estimate the observational probabilistic densities is affected by the sharp truncation of "tails" in both normal distributions. Instead, we estimate the densities by modeling them as a 2-Gausian Mixture Model (2-GMM) using the Expectation-Maximization (EM) algorithm. The 2-GMM is not always reliable owing to the fact that the signals are not strictly Gaussian and that the algorithm is unsupervised of the categories (foreground and background). Our strategy is to get a reliable estimation of the p.d.f. of the background by background extraction and fix it when fitting the mixture model. Our algorithm is as follows:

1. *Background Extraction.*

   Estimate the mean, μ and variance, σ^2 of the entire input image. Binarize the image using threshold, $thr = \mu - 2\sigma$ and dilate the foreground with a 4 by 4 template. We mark the background pixels in the original image using the binarized image and estimate the mean, μ_{b_0} and variance, σ_{b_0} of density p_b from the extracted background pixels.

2. *EM Algorithm for Estimating the 2-GMM.*

   Suppose K samples of the gray-scale values of pixels from the image $z_1, z_2, ..., z_K$ are available and their distribution is $\Lambda Z_1 + (1 - \Lambda)Z_2$ where Z_1, Z_2 and Λ are three random variables, $Z_1 \sim N(\mu_f, \sigma_f^2)$, $Z_2 \sim N(\mu_b, \sigma_b^2)$, $\Lambda \in \{0, 1\}$ and $\Pr(\Lambda = 1) = \lambda$. Denote the density of a normal distribution $N(\mu, \sigma^2)$ by $n_{\mu,\sigma^2}(y)$.

   Initial values: $\hat{\mu}_f = \mu_{b_0}/2$, $\hat{\mu}_b = \mu_{b_0}$, $\hat{\sigma}_f = \hat{\sigma}_b = 10.0$, and $\hat{\lambda} = 0.5$.

   E-step: obtain the expectation of λ for every sample

$$\hat{\lambda}_i = \frac{\hat{\lambda} \cdot n_{\hat{\mu}_f,\hat{\sigma}_f^2}(z_i)}{\hat{\lambda} \cdot n_{\hat{\mu}_f,\hat{\sigma}_f^2}(z_i) + (1 - \hat{\lambda}) \cdot n_{\hat{\mu}_b,\hat{\sigma}_b^2}(z_i)}, i = 1, 2, ..., K. \tag{3.37}$$

   M-step: update the foreground mean and variance:

$$\hat{\mu}_f = \frac{\sum_{i=1}^{K} \hat{\lambda}_i \cdot z_i}{\sum_{i=1}^{K} \hat{\lambda}_i} \tag{3.38}$$

$$\hat{\sigma}_f^2 = \frac{\sum_{i=1}^{K} \hat{\lambda}_i \cdot (z_i - \hat{\mu}_f)^2}{\sum_{i=1}^{K} \hat{\lambda}_i} \tag{3.39}$$

   and the prior

$$\hat{\lambda} = \sum_{i=1}^{K} \hat{\lambda}_i / K \tag{3.40}$$

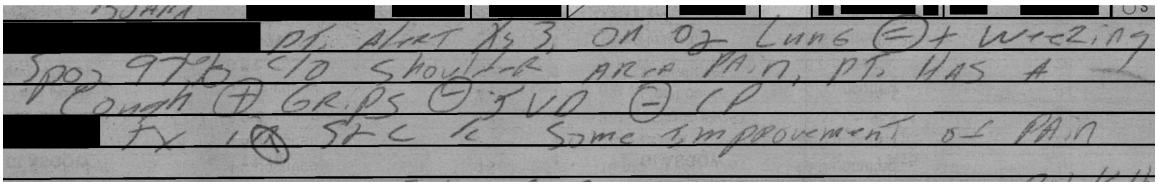


Figure 3.9. A sample patch cropped from a carbon image in our test set. All pixels we intend to paint in are marked in black.

Repeat the above E and M steps until the algorithm converges.

The comparison of the two methods for p.d.f. estimation is shown in Figure 3.8. This p.d.f. estimation algorithm (based on 2-GMM and EM) has an advantage over the thresholding based algorithms because it avoids the problem of truncation of density functions and has a smoother estimation at the intersection of two Gaussian distributions.

Note that we assume the image is bimodal. Our work focuses on document images where the bimodal assumption generally holds. If it does not hold, we can extract the region with different colors through page segmentation, and use local histogram to binarize the image.

3.4.5 Form Grid Removal

First the form grids are located by template matching - this is relatively straightforward to implement because of the fixed form layout, and is true for most types of forms in other applications as well. Therefore, we can define a boolean mask m such that

$$\begin{aligned} & m(j, r, s) = true \Longleftrightarrow \\ & \text{pixel } y_j^{r,s} \text{ is within any of the lines in the grid.} \end{aligned} \tag{3.41}$$

We only need to make a minor modification to Equation (3.36) for the form grid removal:

$$\Pr(y_j|x_j) = 1 \times \prod_{\substack{1 \le r,s \le 5, x_j^{r,s}=0, \\ \text{and } m(j,r,s)=false}} p_f(y_j^{s,t}) \times \prod_{\substack{1 \le r,s \le 5, x_j^{r,s}=1, \\ \text{and } m(j,r,s)=false}} p_b(y_j^{s,t}) \tag{3.42}$$

The probability $\Pr(y_j|x_j)$ in Equation (3.42) is 1 if $m(j,r,s)$ is always $true$ for any r and s in the j-th patch. Replace Equation (3.36) with Equation (3.42) for the compound tasks of binarization and grid removal.

3.4.6 Pruning the Search Space of MRF Inference

So far the MRF based preprocessing algorithm has been presented as a self-contained general algorithm. To make the MRF based algorithm tractable, we adopted a patch based strategy and reduce the search space of each patch using vector quantization. Initially, the size of the search space of every patch x_i in Equations (4) and (5) is 2^{25}. Thus, the domain of every variable x_i is $\{\underbrace{00...0}_{25\ 0's}, \underbrace{00...0}_{24\ 0's}1, ..., \underbrace{11...1}_{25\ 1's}\}$, and we reduce the search space to $\widetilde{C} = \{C_1, C_2, ..., C_{114}\}$ by vector quantization. Although the computation is reduced by the above strategies, the MRF based algorithm is still slower than traditional binarization algorithms. There are ways to make the algorithm faster. Next, we will describe a technique to prune the search space of each x_i. After pruning, a number of elements are removed from $C_1, C_2, ..., C_{114}$ to make an even smaller search space of x_i.

The number of possible values per patch (114) can be reduced by pruning the smaller posterior probabilities $\Pr(x_j = C_l|y)$ calculated using Equation (3.14) after each iteration, *i.e.*,

$$\Pr(x_j = C_l|y) = \frac{\Pr(x_j = C_l)\Pr(y_j|x_j = C_l)\prod_k M_j^k(C_l)}{\sum_{m=1}^{114}\left(\Pr(x_j = C_m)\Pr(y_j|x_j = C_m)\prod_k M_j^k(C_m)\right)} \quad (3.43)$$

$(l = 1, 2, ..., 114)$,

where $M_j^k(C_l)$ is the message from x_j to x_k when $x_j = C_l$. However this pruning is not safe on patches containing pixel(s) to paint in. Due to lack of observations of these pixels, it will take several iterations for them to converge to the right values which may have very small posterior probabilities in the first one or two iterations. Therefore the right values tend to be pruned incorrectly if we prune aggressively. In order to reduce the search space of the inpainted patches, we use a heuristic method to identify the patches surrounded by background and prune their search space. This method is effective due to the higher prior probability of the background (white patches).

Based on the above analysis, we arrive at the following two-step strategy to accelerate the algorithm.

1. Find a global threshold thr_{prune} such that 90% of the pixels in the test image are below thr_{prune}. thr_{prune} is obtained by solving

$$\frac{\hat{\lambda} \cdot n_{\hat{\mu}_f,\hat{\sigma}_f^2}(thr_{prune})}{\hat{\lambda} \cdot n_{\hat{\mu}_f,\hat{\sigma}_f^2}(thr_{prune}) + (1-\hat{\lambda}) \cdot n_{\hat{\mu}_b,\hat{\sigma}_b^2}(thr_{prune})} = 90\%. \quad (3.44)$$

   For any patch x_j, define a pruning mask $PRUNE_j(l), (l = 0, 1, ..., 114)$. If $PRUNE_j(l)$ is $true$, C_l is pruned from the search space for solving x_j. Given a patch x_j and observed patch y_j centered at j_0, the pruning mask of x_j is initialized as $PRUNE_j(1) = false, PRUNE_j(2) = ... = PRUNE_j(114) = true$ if every observed pixel within a 9×9 neighborhood of j_0 is either above thr_{prune} or is marked for in-painting. Thus, all possible values of x_j will be pruned except the pure white patch. Otherwise, the pruning mask is initialized as

$PRUNE_j(1) = PRUNE_j(2) = ... = PRUNE_j(114) = false.$

2. In each iteration, skip any C_l in the search spaces of x_j or x_k in Equations (3.14) and (3.15) if $PRUNE_j(l)$ or $PRUNE_k(l)$ is $true$. Thus, Equation (3.14) becomes

$$\hat{x}_{j\,\mathrm{MAP}} = \underset{x_j,\ PRUNE_j(x_j) \text{ is } false}{\operatorname{argmax}} \Pr(x_j) \Pr(y_j|x_j) \prod_k M_j^k. \tag{3.45}$$

Equation (3.15) becomes

$$M_j^k(x_j) = \max_{x_k,\ PRUNE_k(x_k) \text{ is } false} \Pr(x_k|x_j) \Pr(y_k|x_k) \prod_{l \neq j} \tilde{M}_k^l, \text{ if } PRUNE_j(x_j) \text{ is } false \tag{3.46}$$

After each iteration, update the posterior probabilities $\Pr(x_j = C_l|y)$:

$$\Pr(C_l|y) = \frac{L_l}{\sum_l L_l} \quad (l = 1, 2, ..., 114) \tag{3.47}$$

where

$$L_l = \begin{cases} \Pr(x_j = C_l) \Pr(y_j|x_j = C_l) \prod_k M_j^k(C_l), \\ \quad \text{if } PRUNE_j(l) \text{ is } true \\ 0, \text{ otherwise.} \end{cases} \tag{3.48}$$

Switch any $PRUNE_j(l)$ $(l = 1, 2, ..., 114)$ to $true$ if $\Pr(x_j = C_l|y) < \Pr_{min}$, where $\Pr_{min}$is a threshold of pruning. Larger $\Pr_{min}$ makes the algorithm faster and less accurate.

We will show experimentally how different $\Pr_{min}$ affects the accuracy and the speed of the proposed algorithm. In general, we should choose a small $\Pr_{min}$ so that the algorithm does not depend on heuristic and is reliable.

3.5 Experimental Results and Analysis

3.5.1 Test Datasets

Our test data includes the PCR carbon forms and handwriting images from the IAM database 3.0 [41].

- PCR Forms

  In New York State all patients who enter the Emergency Medical System (EMS) are tracked through their pre-hospital care to the emergency room using the Pre-hospital Care Reports (PCR). The PCR is used to gather vital patient information. The PCR forms are scanned as color images at 300dpi. Handwriting recognition on this data set is quite challenging for several reasons: (i) handwritten responses are very loosely constrained in terms of writing style due to irrepressible emergency situations; (ii) images are scanned from noisy carbon copies and color background leads to low contrast and low signal-to-noise ratio (Figure 5.2); (iii) the (pre-printed) ruling lines often intersect text; (iv) medical lexicons of words are large (more than 4,000 entries). Very low word recognition rates (below 20%) have been reported on this dataset [43]. An example of the handwritten text and pre-printed ruling lines in the PCR forms is shown in Figure 5.2.

- IAM Database

  The IAM database contains high-quality images of unconstrained handwritten English text, which were scanned as grayscale images at 300dpi. Using rough estimates the content of the database can be summarized as follows:

  - 500 writers contributed samples of their handwriting
  - 1,500 pages of scanned text

Table 3.1. Comparison of the speed and accuracy of the proposed algorithm over different values of Pr_{min} tested on the PCR carbon form image (2420 × 370) in Figure 3.9.

Pr_{min}	Number of Different Pixels	Percentage of Different Pixels (%)	Time (sec)
0	0	0	3249
1×10^{-8}	0	0	204
1×10^{-7}	0	0	138
1×10^{-6}	56	0.0063	96
1×10^{-5}	145	0.016	72
1×10^{-4}	308	0.034	57
1×10^{-3}	1122	0.13	37

Table 3.2. Comparison of the speed and accuracy of the proposed algorithm over different values of Pr_{min} tested on the IAM image (2124×369) in Figure 3.12.

Pr_{min}	Number of Different Pixels	Percentage of Different Pixels (%)	Time (sec)
0	0	0	1694
1×10^{-8}	0	0	29
1×10^{-7}	0	0	25
1×10^{-6}	0	0	24
1×10^{-5}	0	0	24
1×10^{-4}	14	0.002	23
1×10^{-3}	107	0.014	23

- 10,000 isolated and labeled text lines
- 100,000 isolated and labeled words

3.5.2 Display of Preprocessing Results

First we applied our algorithm to the input image (Figure 3.9). This input image is cropped from a PCR form. Lines and unwanted machine-printed blocks are identified and marked in black. Our test images and images for training the prior model are from different writers. It is clear that the writing style in Figure 3.9 is not like any of the styles in Figure 3.7. The results after iterations 1, 2, 4, and 16 of belief propagation

run on Figure 3.9 are shown in Figure 3.11. After the first iteration, the message has not yet been passed between neighbors. The edges of strokes are jagged due to noisy background and error in the vector quantization discussed in section 3.4.3. All of the pre-printed lines are dropped. After 2 iterations, text edges are smoothed but most lines are not fully restored. After 4 iterations, nearly all the strokes are restored, with a few remaining glitches. After 16 iterations the glitches are mostly removed.

3.5.3 Results of Acceleration: Speed vs. Accuracy

We have tested the effect of different values of parameter $\Pr_{min}$ on the speed and accuracy of our algorithm using the PCR carbon form image in Figure 3.9 and the IAM handwriting image in Figure 3.12. In order to compare the results obtained by our algorithm with different values of $\Pr_{min}$, we have taken the output images of $\Pr_{min} = 0$ (which indicates no acceleration) as reference images, and have counted the pixels in the output images with various $\Pr_{min}$'s that are different from the reference images. The results are shown in Table 3.1. The running times are obtained on a PC with an Intel 2.8G Hz CPU.

In Table 3.1, even with a very small $\Pr_{min}$, *e.g.* 10^{-8}, the running time decreased significantly. The error rate of the low-quality PCR image is below 0.01% when $\Pr_{min} \leq 1 \times 10^{-6}$, and is zero when $\Pr_{min} \leq 1 \times 10^{-7}$. In Table 3.2, the error rate of the high-quality IAM image is below 0.01% when $\Pr_{min} \leq 1 \times 10^{-4}$, and is zero when $\Pr_{min} \leq 1 \times 10^{-5}$. In the following experiments on comparing OCR results, we chose $\Pr_{min} = 10^{-7}$.

3.5.4 Comparison to Other Preprocessing Methods

Our approach has been compared (Figure 3.13) with the preprocessing algorithm of Milewski *et al.* [43], Niblack algorithm [45], and Otsu algorithm [47]. Milewski algorithm performs both binarization and line removal. Niblack and Otsu are for binarization only. The text of the images shown in Figure 3.13 is "67 yo ♀ pt found

mfg X Ray". From the result of the MRF based algorithm, the text "67 yo ♀ pt found" is clear and the "X ray" is obscured but is still legible. In the output of the Milewski's algorithm, the words "pt", "mfg", "X", and "Ray" are not legible. The output of Niblack is noisier although it retains some details of the foreground. The result of Otsu is also very noisy and loses more foreground details than Niblack. Figure 3.14 shows our line removal (Figure 3.14(c)) achieves a smoother restoration of strokes which touch the form grid than the Milewski algorithm (Figure 3.14(b)).

In addition to the above qualitative comparison, we have also used the OCR test to obtain a quantitative comparison. First we tested the four algorithms on 100 PCR forms. All of the 3149 binarized word images extracted from the 100 form images were recognized using the word recognition algorithm in [33] with a lexicon of 4580 English words. We split the 3149 word images into two sets: set #1 contains 1203 word images that are not affected by overlapping form lines, *i.e.*, no intersection of stroke and line; set #2 contains 1946 pairs that are affected by form lines. Thus, the word recognition accuracy on set #1 measures the performance of binarization only and can be used to compare all four algorithms.

We calculated the top-n ($n \geq 1$) recognition rates instead of only the top-1 rate for comparison because top-n rates are of greater importance to the problem of indexing text with very high error rate [25]. Moreover, recognition rates measured in terms of multiple candidates provides a strong proof of the effectiveness of the preprocessing techniques. Table 3.3 shows that the MRF based method results in higher overall recognition rates and also performs more efficient line removal.

We have also run the MRF binarization algorithm on some images from IAM DB3.0 [41]. We generated zero-mean Gaussian noise with deviation σ=50, 70, and 100 in the IAM images to test the performance of binarization algorithms at different noise levels. For the group of images of σ=100, a 3×3 mean filter was applied to all the images before binarization. The top-1 word recognition rates of the original images

Table 3.3. Comparison of word recognition rates of Milewski algorithm, MRF based approach, Niblack and Otsu algorithms (set #1: sample word images not affected by forms lines; set #2: sample word images affected by forms lines; overall: set #1 + set #2).

Method		Milewski	**MRF**	Niblack	Otsu
Set #1	Top 1 rate	17.5%	**25.9%**	19.4%	11.6%
	Top 2 rate	24.4%	**36.6%**	26.9%	16.0%
	Top 5 rate	33.4%	**44.9%**	35.9%	23.3%
	Top 10 rate	39.6%	**51.7%**	42.3%	28.8%
Set #2	Top 1 rate	19.5%	**30.3%**	NA	NA
	Top 2 rate	28.1%	**40.7%**	NA	NA
	Top 5 rate	37.6%	**52.7%**	NA	NA
	Top 10 rate	45.0%	**60.0%**	NA	NA
Overall	Top 1 rate	18.7%	**28.6%**	NA	NA
	Top 2 rate	26.7%	**39.1%**	NA	NA
	Top 5 rate	36.0%	**49.7%**	NA	NA
	Top 10 rate	42.9%	**56.8%**	NA	NA

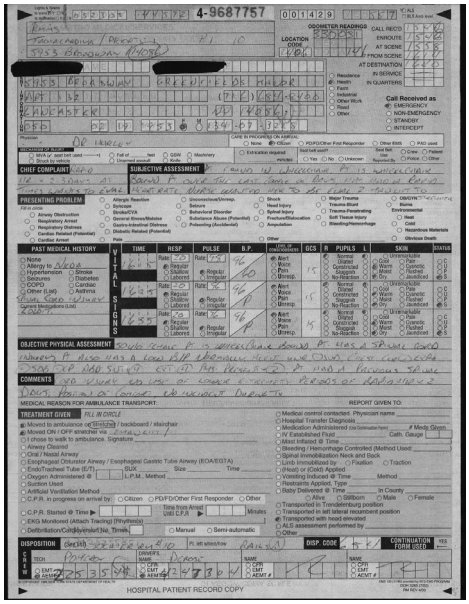
Table 3.4. Comparison of word recognition rates (top-one accuracies in percentage) of the MRF based method, Niblack algorithm and Otsu algorithm on images with different noise levels.

	Original images	Gaussian Noise ($\sigma = 50$)	Gaussian Noise ($\sigma = 70$)	Gaussian Noise ($\sigma = 100$) and 3×3 Mean Filter
MRF	**83.0%**	**70.3%**	**43.7%**	**48.1%**
Niblack	83.0%	60.7%	31.1%	38.5%
Otsu	82.2%	65.2%	37.0%	37.8%

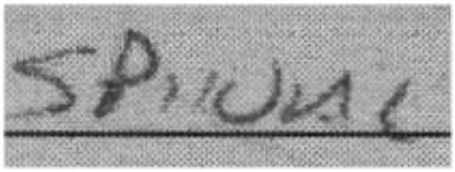
and the images with Gaussian noise binarized by the MRF based method, Niblack, and Otsu are shown in Table 3.4. Each group has 135 word images. We use a lexicon of 59 English words. The word recognition rates of the original images among all three methods are very close. The MRF based method shows higher recognition rates on the images with Gaussian noise.

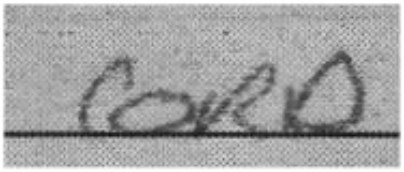
3.6 Summary

In this chapter we have presented a novel method for binarizing degraded document images of handwriting and removing pre-printed form lines. Our method models binarized objective image as a Markov Random Field. Different from related approaches, we reduce the large search space of the prior model to a class of 114 representatives by vector quantization, and learn the observation model directly from the input image. We also presented an effective method of pruning the search space of the MRF. Our work is the first attempt at applying a stochastic method to preprocessing of degraded high-resolution handwritten documents. Our model is targeted towards document images, and therefore may not handle intense illumination variations, complex backgrounds, and blurring that are common in tasks of video and scene text processing. However it is possible to generalize our model to these applications as well.

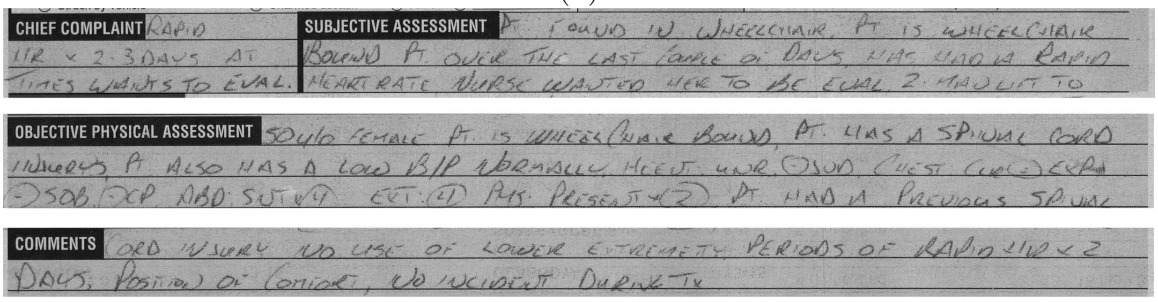


(a)



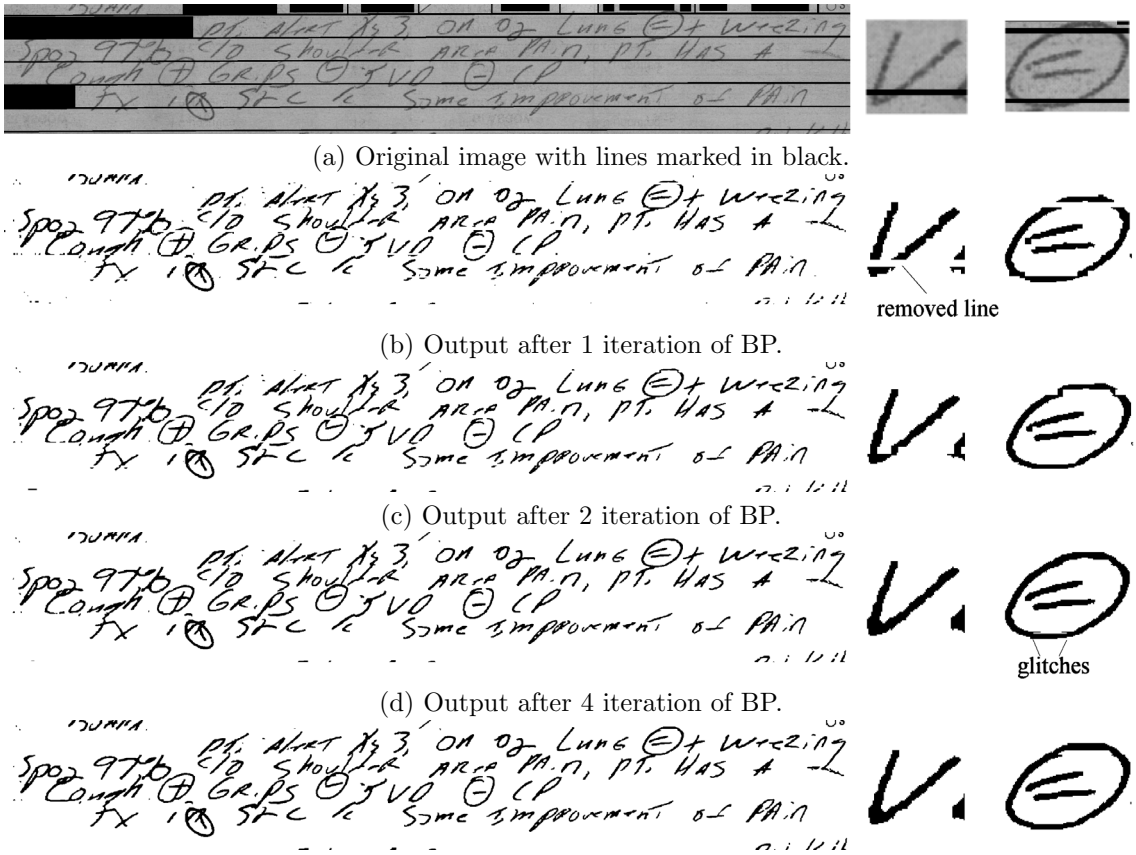


(b)



(c)

Figure 3.10. An example of PCR forms. (a) A entire PCR form. (b) A small local region showing obscure text and background noise array. (c) Fields of interest in the PCR form.



(a) Original image with lines marked in black.

(b) Output after 1 iteration of BP.

(c) Output after 2 iteration of BP.

(d) Output after 4 iteration of BP.

(e) Output after 16 iteration of BP.

Figure 3.11. The binarization and line removal result of the sample shown in figure 3.9.

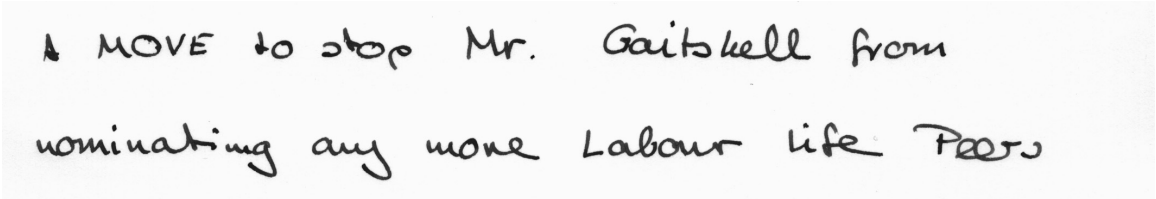
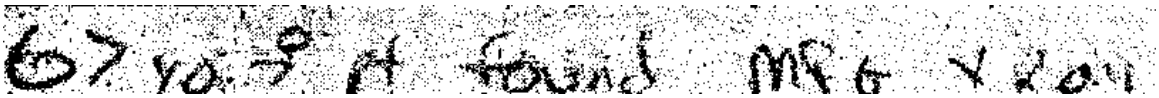


Figure 3.12. A sample from IAM database.

(a) Input image.

(b) Output of the Milewski algorithm.

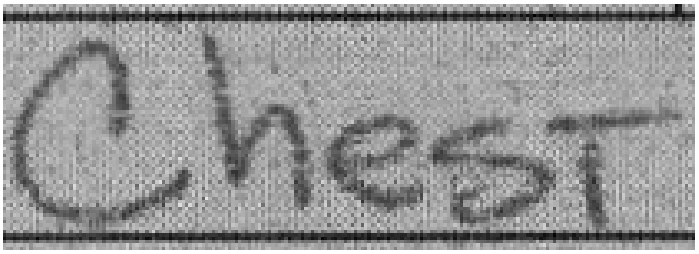
(c) Output of the MRF based algorithm.



(d) Output of the Niblack algorithm.

(e) Output of the Otsu algorithm.

Figure 3.13. Comparison of binarization results of the MRF based algorithm versus three other algorithms.



(a) Input image.

(b) Output of the Milewski algorithm.

(c) Output of the MRF based algorithm.

Figure 3.14. Comparison of line removal results of the Milewski algorithm and the MRF based algorithm.

CHAPTER 4

HANDWRITTEN DOCUMENT RETRIEVAL

4.1 Introduction

Our work is motivated by the lack of tools available to search handwritten documents. Although the development in optical character recognition (OCR) and information retrieval (IR) techniques have provided ways to digitalize and search printed of documents, a similar approach for handwritten documents is undermined by the errors occurring in document analysis and recognition [17, 3]. The state of the art word recognition accuracy is 60-70% on handwritten documents of good quality which makes IR results acceptable, but only 20-30% on low-quality documents such as historical manuscripts, carbon forms, *etc.* Therefore conventional IR algorithms perform poorly on these documents.

Several researches have been proposed to improve the IR performance of OCR'ed text to overcome this problem. Mittendorf *et al.* [44] adjust the term-weighting scheme of IR using a model of OCR errors. Ohta *et al.* [46] generate candidate terms for each "true" search term and add the retrieval results of candidate terms into the final result. Jing [28] uses a language model that takes common recognition errors into account to approximate an "uncorrupted" version of the document. These methods focus on modeling and correcting OCR errors and are primarily applicable to machine-printed documents.

There has been some work on handwritten document retrieval recently[8, 21, 49, 24, 37]. Lee *et al.*[37] run retrieval tests on text composed of top-k ($k > 1$) candidates of character recognition results of Hangul document images as opposed to the OCR'ed

text composed of top-1 candidates. The use of top-k ($k > 1$) candidates improves the recall performance of the IR system. Rath *et al.* [49] use an IR model that takes the product of frequencies of query terms in a document as the similarity between the query and the document. They assign different frequencies to terms according to the posterior probabilities of terms. Their method is to estimate probabilities directly from the vector space of profile features of word images which can be improved by using the probabilities produced by established probabilistic word recognition algorithms such as HMM. Howe *et al* [24] use the same IR model as [49] but they simply assume the word recognition probabilities to be inversely proportional to the recognition rank which is more effective than the probabilities estimated from the training set. In our prior work [13], we use the Vector IR Model [1] for retrieval and learn the term probabilities from word recognition results on the training set. The Vector Model takes weighted sum of term frequencies as the similarity measure and performs better (in our approach) than the model in [24, 49] that uses multiplicative similarity.

We present an approach to relevant retrieval of handwritten documents in this chapter. Our retrieval method is based on the modified Vector IR Model presented in our previous works [8, 21]. Different from text retrieval, the raw term frequency (the number of occurrences of a term in a document) required by the Vector Model is not immediately available. We estimate the raw term frequency from word segmentation and recognition results using a probabilistic method [14, 9]. By assuming perfect word segmentation, the existing methods [24, 49, 13] estimate the raw term frequency as the sum of word recognition probabilities. We improve upon the above methods by taking word segmentation errors and language model into account (Table 4.1). The solution to the term-weighting scheme that unifies segmentation probabilities, recognition probabilities and the language model (n-gram) is non-trivial due to large amount of search branches, as opposed to the scheme that only uses word recognition probabilities. We solve this problem using dynamic programming.

Table 4.1. Approaches to handwritten document retrieval.

Existing Approach I	Existing Approach II	Proposed Approach
Index is created using OCR'ed text [37]	Index is created using word recognition probabilities [8, 21, 49, 24]	Index is created using word recognition probabilities, word segmentation probabilities and language model (n-gram, $n > 1$)

4.2 Vector IR Model for Handwritten Documents

4.2.1 Classic Vector Model

In the classic Vector Model [1], the documents are represented by the vector space of terms. A term is a word from the vocabulary of all of the documents. Given the vocabulary $\{t_i\}, 1 \leq i \leq N$, the term frequency of document d_j is defined by formula

$$tf_{i,j} = \frac{freq_{i,j}}{L_j}, \quad i = 1, ..., N \tag{4.1}$$

where $freq_{i,j}$ is the number of occurrences of term t_i in document d_j (**raw term frequency**) and L_j is the total number of occurrences of all terms in document d_j, i.e., the length of d_j. For example, in a document d_j of 1000 words, if the term t_i="diseases" occurs 3 times, then the raw term frequency $freq_{i,j} = 3$ and term frequency $tf_{i,j} = 0.003$. Thus document d_j can be represented by the vector $[tf_{1,j}, tf_{2,j}, ..., tf_{N,j}]$.

The inverse document frequency (IDF) of a term is defined by the formula

$$idf_i = \log \frac{\#\{d_j\}}{\#\{d_j | freq_{i,j} > 0\}}, \quad i = 1, ..., N \tag{4.2}$$

where $\#\{\cdot\}$ denotes the number of elements in set $\{\cdot\}$. The IDF of a term shows the importance of the term based on the observation that a term that appears in most documents is less important than a term that appears in only a few documents.

A query is also represented by the vector of terms. The query term frequency (QTF) of query q is defined as

$$tf_{i,q} = \begin{cases} 1, & \text{if term } t_i \text{ is in } q \\ 0, & \text{otherwise} \end{cases} \qquad i = 1, ..., N \tag{4.3}$$

and the query is represented by vector $[tf_{1,q}, tf_{2,q}, ..., tf_{N,q}]$.

The similarity between document d_j and query q is defined as

$$\text{sim}(d_j, q) = \sum_{i=1}^{N} tf_{i,j} \cdot idf_i \cdot tf_{i,q}. \tag{4.4}$$

Now let's show an example of the Vector Model. Support we have the following two documents:

d_1="pt has a trauma", and

d_2="pt has breath difficulty",

where "pt" is the abbreviation for "patient".

Then there are 6 terms:

t_1="pt", t_2="has", t_3="a", t_4="trauma", t_5="breath", and t_6="difficulty".

The term frequency matrix is

$$(tf_{i,j}) = \begin{bmatrix} 0.25 & 0.25 & 0.25 & 0.25 & 0 & 0 \\ 0.25 & 0.25 & 0 & 0 & 0.25 & 0.25 \end{bmatrix} \tag{4.5}$$

The vector of IDF's is

$$\begin{aligned} (idf_i) &= \left[\log\frac{2}{2}, \log\frac{2}{2}, \log\frac{2}{1}, \log\frac{2}{1}, \log\frac{2}{1}, \log\frac{2}{1}\right] \\ &= [0, 0, \log 2, \log 2, \log 2, \log 2] \end{aligned} \tag{4.6}$$

Suppose a query is q: "breath difficulty", then the vector of QTF's is

$$(qtf_{i,q}) = [0, 0, 0, 0, 1, 1] \tag{4.7}$$

The similarity between document d_1 and query q is

$sim_{d_1,q} = 0.$

The similarity between document d_2 and query q is

$sim_{d_1,q} = 0.5 \log 2.$

This shows that document d_2 is more relevant to the query than document d_1.

4.2.2 Modified Vector Model

The raw frequency $freq_{i,j}$ is not immediately available from the document image and need to be estimated. Thus we modify the definitions of TF and IDF in Equations (4.1) and (4.2): the modified TF is

$$tf'_{i,j} = \frac{E\{freq_{i,j}\}}{L_j}, \tag{4.8}$$

and the modified IDF

$$idf'_i = \log \frac{\#\{d_j\}}{\max\{1, \#\{d_j | E\{freq_{i,j}\} > 0.5\}\}} \tag{4.9}$$

where $E\{freq_{i,j}\}$ is an estimation of $freq_{i,j}$. Note that here we use $E\{freq_{i,j}\} > 0.5$ which is equivalent to a rounding function of the expected value of $freq_{i,j}$, *i.e.*, $round(E\{freq_{i,j}\}) \geq 1$.

The text length in Equation (4.8) is estimated by

$$L_j = \sum_{i=1}^{N} E\{freq_{i,j}\} \tag{4.10}$$

The similarity between document image d_j and the query q is given by

$$\text{sim}(d_j, q) = \sum_{i=1}^{N} tf'_{i,j} \cdot idf'_i \cdot tf_{i,q}. \tag{4.11}$$

We estimate $E\{freq_{i,j}\}$ using the MMSE method as follows. Suppose document d_j is composed of an observational sequence of image features denoted by $\overrightarrow{o} = o_1 o_2 ... o_N$, and $\overrightarrow{w} = w_1 w_2 ... w_L$ is any segmentation of sequence $\overrightarrow{o}$ where $w_1, ..., w_L$ are word images. The MMSE estimation of $freq_{i,j}$ is given by

$$E\{freq_{i,j}\} = \sum_{\overrightarrow{w}} \Pr(\overrightarrow{w}|\overrightarrow{o}) \cdot \sum_{\overrightarrow{\tau}} \Pr(\overrightarrow{\tau}|\overrightarrow{w}) \cdot \#_{t_i}(\overrightarrow{\tau}) \tag{4.12}$$

where $\overrightarrow{\tau} = \tau_1 ... \tau_L$ is a sequence of terms. $\Pr(\overrightarrow{w}|\overrightarrow{o})$ is the probability that $\overrightarrow{w}$ is a valid segmentation. $\Pr(\overrightarrow{\tau}|\overrightarrow{w})$ is the word sequence recognition probability. $\#_{t_i}(\overrightarrow{\tau})$ is the number of term t_i occurring in sequence $\overrightarrow{\tau}$.

Equation (4.12) can be simplified in some special situations. $\overrightarrow{w}$ is unique and $\Pr(\overrightarrow{w}|\overrightarrow{o}) \equiv 1$ if we assume the correct segmentation $\overrightarrow{w}$ is known. Thus Equation (4.12) is equivalent to

$$E\{freq_{i,j}\} = \sum_{\overrightarrow{\tau}} \Pr(\overrightarrow{\tau}|\overrightarrow{w}) \cdot \#_{t_i}(\overrightarrow{\tau}) \tag{4.13}$$

In addition to the assumption of knowing the correct segmentation, assuming the independence of terms $\tau_1, \tau_2, ..., \tau_L$, *i.e.*,

$$\Pr(\overrightarrow{\tau}|\overrightarrow{w}) = \prod_{k=1}^{L} \Pr(\tau_k|w_k), \tag{4.14}$$

then Equation (4.13) is equivalent to

$$E\{freq_{i,j}\} = \sum_{k=1}^{L} \Pr(\tau_k|w_k) \tag{4.15}$$

Equation (4.15) is used in [49, 24] for retrieval. It is a solution to Equation (4.12) based on the assumptions of perfect word segmentation and independence of terms. In the general case, given the probability of every single segmentation point and a language model (n-gram), we can solve Equation (4.12) by dynamic programming.

4.2.3 Estimating Raw Term Frequency $freq_{i,j}$

The observational sequence of a document image can be represented by a sequence of connected components sorted in the reading order. Since the following discussion focuses on a single document, we can omit the subscript j of d_j from notations like $freq_{i,j}$ without ambiguity.

Given N consecutive connected components $c_1, ...c_n$ and the set of terms $t_1, ...t_N$, we use a dynamic programming based algorithm to solve the raw term frequency. We assume a word image is composed of at most C connected components. The raw term frequency of t_i in sequence $c_1, ..., c_k$ $(0 < k \leq n)$ is denoted by $freq_i^k$. The probability that the last word of sequence $c_1, ..., c_k$ is term t_i is denoted by λ_i^k. The probability that the gap after the connected component c_k is a true word gap is denoted by σ_k. When we define $freq_i^k$ and σ_k on a sequence $c_1, ..., c_k$, we assume $\sigma_0 = \sigma_k = 1$.

When $k = 0$, the sequence is empty, and thus

$$E(freq_i^0) = 0 \tag{4.16}$$

When $k = 1$, the only possible segmentation is that c_1 is a word image, and thus

$$E(freq_i^1) = \frac{p_i \cdot \Pr(c_1|t_i)}{\sum_{i_2=1}^{N} p_{i_2} \cdot \Pr(c_1|t_{i_2})} \tag{4.17}$$

When $k = 2$, the last word image can be either c_2 or c_1c_2. The probability of c_2 being t_i equals

$$\sum_{i_1=1}^{N} \lambda_{i_1}^1 \cdot \frac{p_{i_1 \to i} \cdot \Pr(c_2|t_i)}{\sum_{i_2=1}^{N} p_{i_1 \to i_2} \cdot \Pr(c_2|t_{i_2})}, \tag{4.18}$$

where $p_{i_1 \to i_2}$ represents the transition probability from term t_{i_1} to term t_{i_2} and $\Pr(c_2|t_i)$ is the probability density of observation c_2 in class t_i. The probability of c_1c_2 being t_i equals

$$\frac{p_i \cdot \Pr(c_1c_2|t_i)}{\sum_{i_2=1}^{N} p_{i_2} \cdot \Pr(c_1c_2|t_{i_2})} \tag{4.19}$$

Thus

$$E(freq_i^2) = \sigma_1 \cdot (freq_i^1 + \sum_{i_1=1}^{N} \lambda_{i_1}^1 \cdot \frac{p_{i_1 \to i} \cdot \Pr(c_2|t_i)}{\sum_{i_2=1}^{N} p_{i_1 \to i_2} \cdot \Pr(c_2|t_{i_2})}) + (1 - \sigma_1) \cdot \frac{p_i \cdot \Pr(c_1c_2|t_i)}{\sum_{i_2=1}^{N} p_{i_2} \cdot \Pr(c_1c_2|t_{i_2})} \tag{4.20}$$

For an arbitrary $k > 0$, we can prove that

$$E(freq_i^k) =$$

$$\sum_{c=1}^{k-1}\sigma_{k-c}\cdot(\prod_{k-c<q<k}(1-\sigma_q))\cdot(freq_i^{k-c}+\sum_{i_1=1}^{N}\lambda_{i_1}^{k-c}\cdot\frac{p_{i_1\to i}\cdot\Pr(c_{k-c+1}...c_{k-1}c_k|t_i)}{\sum_{i_2=1}^{N}p_{i_1\to i_2}\cdot\Pr(c_{k-c+1}...c_{k-1}c_k|t_{i_2})})$$

$$+(\prod_{0<q<k}(1-\sigma_q))\cdot(\frac{p_i\cdot\Pr(c_1...c_{k-1}c_k|t_i)}{\sum_{i_2=1}^{N}p_{i_2}\cdot\Pr(c_1...c_{k-1}c_k|t_{i_2})})$$

if $k \leq C$;

$$E(freq_i^k) =$$

$$\sum_{c=1}^{C}\sigma_{k-c}\cdot(\prod_{k-c<q<k}(1-\sigma_q))\cdot(freq_i^{k-c}+\sum_{i_1=1}^{N}\lambda_{i_1}^{k-c}\cdot\frac{p_{i_1\to i}\cdot\Pr(c_{k-c+1}...c_{k-1}c_k|t_i)}{\sum_{i_2=1}^{N}p_{i_1\to i_2}\cdot\Pr(c_{k-c+1}...c_{k-1}c_k|t_{i_2})})$$

if $k > C$.

(4.21)

Similarly, we can prove that

$$\lambda_i^0 = \frac{1}{N};$$

$$\lambda_i^k = \sum_{c=1}^{k-1} \sigma_{k-c} \cdot \Big(\prod_{k-c<q<k} (1-\sigma_q) \Big) \cdot \Big(\sum_{i_1=1}^{N} \lambda_{i_1}^{k-c} \cdot \frac{p_{i_1 \to i} \cdot \Pr(c_{k-c+1}...c_{k-1}c_k|t_i)}{\sum_{i_2=1}^{N} p_{i_1 \to i_2} \cdot \Pr(c_{k-c+1}...c_{k-1}c_k|t_{i_2})} \Big)$$
$$+ \Big(\prod_{0<q<k} (1-\sigma_q) \Big) \cdot \Big(\frac{p_i \cdot \Pr(c_1...c_{k-1}c_k|t_i)}{\sum_{i_2=1}^{N} p_{i_2} \cdot \Pr(c_1...c_{k-1}c_k|t_{i_2})} \Big)$$

if $1 \leq k \leq C$;

$$\lambda_i^k = \sum_{c=1}^{C} \sigma_{k-c} \cdot \Big(\prod_{k-c<q<k} (1-\sigma_q) \Big) \cdot \Big(\sum_{i_1=1}^{N} \lambda_{i_1}^{k-c} \cdot \frac{p_{i_1 \to i} \cdot \Pr(c_{k-c+1}...c_{k-1}c_k|t_i)}{\sum_{i_2=1}^{N} p_{i_1 \to i_2} \cdot \Pr(c_{k-c+1}...c_{k-1}c_k|t_{i_2})} \Big)$$

if $k > C$.

(4.22)

The raw term frequencies $freq_i^n$ $(i = 1, 2, ..., N)$ are obtained by calculating $freq_i^k$'s and λ_i^k's recursively for k from 0 to n using Equations (4.16) - (4.22).

4.2.4 Estimating Word Segmentation Probability

Word segmentation is defined as the process of segmenting a line into words. In handwritten lines, the space between words is uneven. Moreover, the same amount of space may be present between words, and between characters within a word. Such cases arise due to differences in writing styles, and space constraints.

In our word segmentation method, for the gap between any two consecutive connected components, the probability of the gap being a valid word gap is estimated. A gap between two connected components is represented by three features:

1. **Euclidean Distance.** This feature is defined as the horizontal distance between the bounding boxes of the two consecutive connected components of the line image (Figure 4.1(a)).

2. **Minimum Run Length.** This feature represents the minimum horizontal white run length distance between the two adjacent connected components of the line image.

3. **Convex Hull Distance.** We compute the convex hulls of two consecutive connected components and draw a line connecting the mass centers of the two convex hulls. The Euclidean distance between points at which this line crosses the two convex hulls is defined as the Convex Hull distance of the two adjacent components.

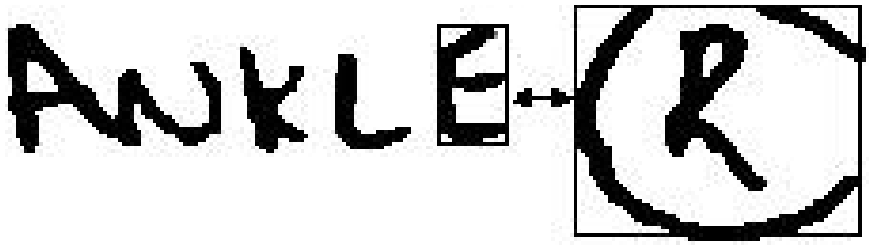
To eliminate the effect of different text sizes, we compute the average height of all the components and normalize the extracted features by dividing them by the average height of all components in the same line.

The segmentation probability of a gap g is given by the Bayes' Rule

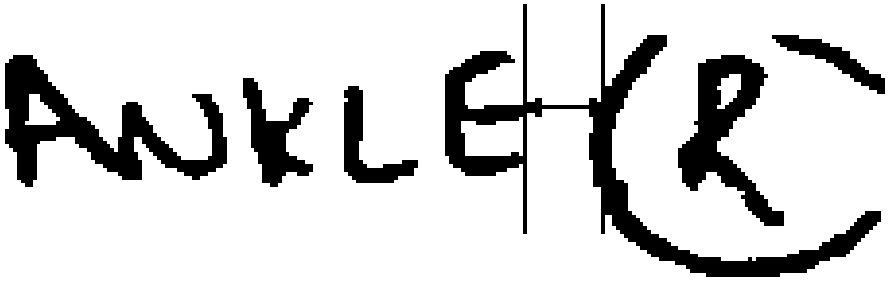
$$\sigma_g = \Pr(g|f_{1,g}, f_{2,g}, f_{3,g}) = \frac{\Pr(g)p(f_{1,g}, f_{2,g}, f_{3,g}|g)}{\Pr(g)p(f_{1,g}, f_{2,g}, f_{3,g}|g) + \Pr(\bar{g})p(f_{1,g}, f_{2,g}, f_{3,g}|\bar{g})} \quad (4.23)$$

where $\Pr(g)$ and $\Pr(\bar{g})$ are the prior probabilities of valid gaps and non-valid gaps, respectively. $f_{1,g}$, $f_{2,g}$ and $f_{3,g}$ are three features of g. $p(f_{1,g}, f_{2,g}, f_{3,g}|g)$ is the probability density of the features of valid gaps. $p(f_{1,g}, f_{2,g}, f_{3,g}|\bar{g})$ is the probability density of the features of non-valid gaps.

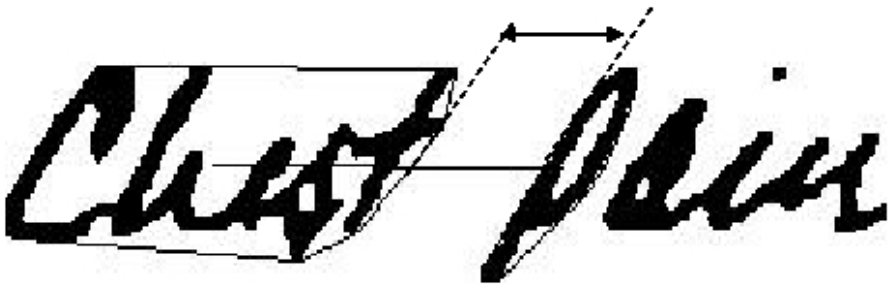
Given a set of gap features with the annotation of *"valid"* and *"non-valid"*, we can estimate $\Pr(g)$, $\Pr(\bar{g})$, $p(f_{1,g}, f_{2,g}, f_{3,g}|g)$ and $p(f_{1,g}, f_{2,g}, f_{3,g}|\bar{g})$ as follows. $\Pr(g)$



(a) Euclidean distance.



(b) Run length distance.



(c) Convex hull distance.

Figure 4.1. Three feature representing a gap between two consecutive connected components.

and $\Pr(\bar{g})$ are estimated from the ratio of the numbers of valid and non-valid gaps in the training set.

$$\Pr(g) = \frac{\#\{\text{valid gaps}\}}{\#\{\text{valid gaps}\} + \#\{\text{non-valid gaps}\}} \tag{4.24}$$

$$\Pr(\bar{g}) = 1 - \Pr(g) \tag{4.25}$$

$p(f_{1,g}, f_{2,g}, f_{3,g}|g)$ and $p(f_{1,g}, f_{2,g}, f_{3,g}|\bar{g})$ are estimated non-parametrically using Parzen window technique with a Gaussian kernel function.

4.2.5 Estimating Word Recognition Likelihood

We use a lexicon-driven word recognition algorithm [33] based on character segmentation and dynamic programming to find the best matching path. First a word image is segmented into candidate character images. Then the directional features are extracted from the contours of character images and matched to every word in the lexicon by searching all possible segmentations for the minimum sum of Euclidean distances from the features of the test image and the character templates in the training set. The minimum Euclidean distance indicates the similarity between the word image and the term in the lexicon. The square of the distance associated with a pair of a word image w and a term t_i is denoted by $s(w, t_i)$.

The word recognition likelihood is estimated from the recognition score using a Universal Background Model (UBM) [50]. In a Background Model, the posterior probability of the word recognition is given by Bayes' rule:

$$\Pr(w = t_i|s(w,t_i)) = \frac{\Pr(w = t_i)p_{t_i}(s(w,t_i)|w = t_i)}{\Pr(w = t_i)p_{t_i}(s(w,t_i)|w = t_i) + \Pr(w \neq t_i)p_{t_i}(s(w,t_i)|w \neq t_i)} \tag{4.26}$$

where $p_{t_i}(s(w,t_i)|w = t_i)$ is the likelihood of the genuine matching score of t_i, $p_{t_i}(s(w,t_i)|w \neq t_i)$ is the likelihood of the imposter matching score of t_i, and $\Pr(w =$

t_i), $\Pr(w \neq t_i)$ are the prior probabilities of genuine and imposter matches of t_i, respectively.

We need a term specific training set for every term to learn the background model. This is a drawback in applications using large number of terms. The Universal Background Model is an alternative approach that solves this problem. In the UBM, we use a single Background Model for all of the terms. The genuine matching probability is given by

$$UBM(s) = \Pr(Genuine|s) = \frac{\Pr(Genuine)p(s|Genuine)}{\Pr(Genuine)p(s|Genuine) + \Pr(Imposter)p(s|Imposter)} \tag{4.27}$$

where s is a matching score, $\Pr(Genuine)$, $\Pr(Imposter)$ are the prior probabilities of genuine match and imposter match, respectively, and $p(s|Genuine)$, $p(s|Imposter)$ are the likelihoods of the score of genuine match and imposter match, respectively. $\Pr(Genuine)$, $\Pr(Imposter)$, $p(s|Genuine)$, and $p(s|Imposter)$ are estimated from the scores of all of the terms.

We model $p(s|Genuine)$ and $p(s|Imposter)$ as Gamma distributions. Actually, the matching score s is a squared sum of distances between character-level feature vectors and the centers of clusters in the training features. In other words,

$$s = \sum_{l=1}^{L} D_l^2 \tag{4.28}$$

where D_l is a character matching distance. If we assume all the clusters of the training feature vector space are independent normal distributions, then the squared sum of the distances can be modeled as a gamma distribution. The probability density function of the gamma distribution can be represented by

$$f_S(s; k, \theta) = s^{k-1}\frac{e^{-s/\theta}}{\theta^k \Gamma(k)}, s > 0 \text{ and } k, \theta > 0 \tag{4.29}$$

where $\Gamma(k)$ is the gamma function:

$$\Gamma(k) = \int_0^\infty x^{k-1} e^{-x} dx. \tag{4.30}$$

If k is a positive integer, then $\Gamma(k) = (k-1)!$. There is no closed-form solution for the maximum likelihood estimation of k and θ [16]. However we can use a simple way to estimate the Gamma distribution. First we can prove that the mean and variance of the Gamma distribution are $k \cdot \theta$ and $k \cdot \theta^2$, respectively. Then, given N genuine matching scores $s_1, s_2, ...s_N$, we can compute the ML estimation of mean and variance:

$$\begin{cases} \bar{\mu} = \dfrac{1}{N} \displaystyle\sum_{i=1}^{N} s_i \\ \bar{\sigma}^2 = \dfrac{1}{N} \displaystyle\sum_{i=1}^{N} (s_i - \bar{\mu})^2 \end{cases} \tag{4.31}$$

Let $\bar{k} \cdot \bar{\theta} = \bar{\mu}$ and $\bar{k} \cdot \bar{\theta}^2 = \bar{\sigma}^2$, then

$$\begin{cases} \bar{k} = \dfrac{\bar{\mu}^2}{\bar{\theta}^2} \\ \bar{\theta} = \dfrac{\bar{\theta}^2}{\bar{\mu}} \end{cases} \tag{4.32}$$

A Genuine probability/score curve estimated from 5461 genuine matching scores and 1,226,022 imposter matching scores is shown in Figure 4.2.

We estimate the posterior probabilities by amending Equation (5.6):

$$\Pr(t_i|w) = \frac{\Pr(t_i) UBM(s(w, t_i))}{\displaystyle\sum_{j=1}^{N} \Pr(t_j) UBM(s(w, t_j))}, i = 1, 2, ..., N \tag{4.33}$$

By Bayes' rule, the likelihood

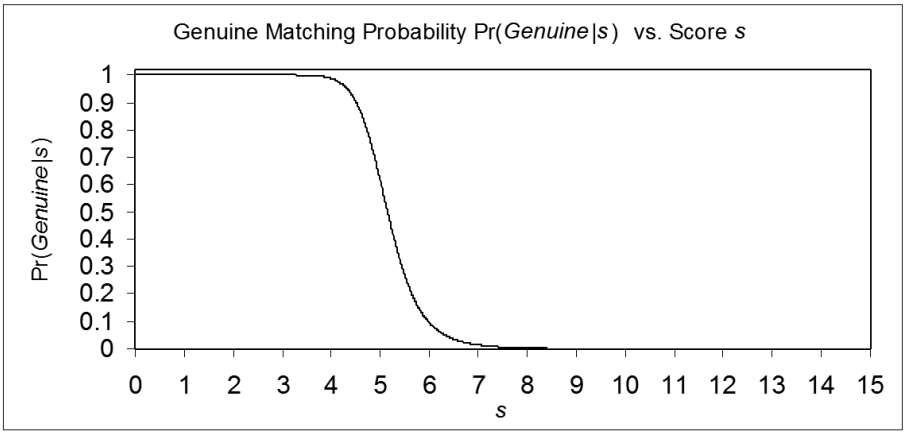


Figure 4.2. Genuine matching probability/score curve estimated from training set.

$$\begin{aligned}\Pr(w|t_i) &= \frac{\sum_{j=1}^{N} \Pr(t_j)\Pr(w|t_j)}{\Pr(t_i)} \cdot \Pr(t_i|w) \\ &= UBM(s(w,t_i)) \cdot \frac{\sum_{j=1}^{N} \Pr(t_j)\Pr(w|t_j)}{\sum_{j=1}^{N} \Pr(t_j) UBM(s(w,t_j))}\end{aligned} \tag{4.34}$$

where $\frac{\sum_{j=1}^{N} \Pr(t_j)\Pr(w|t_j)}{\sum_{j=1}^{N} \Pr(t_j) UBM(s(w,t_j))}$ is an invariant of t_i and can be reduced from the fractions in Equations (4.16) - (4.22). Thus we can use

$$p(w|t_i) \propto UBM(s(w,t_i)) \tag{4.35}$$

to estimate the likelihoods in Equations (4.16) - (4.22).

4.2.6 Search Engine Based on Modified Vector Model

A search engine for handwritten document is built using the modified Vector Model and raw term frequency estimation method discussed in the previous sections.

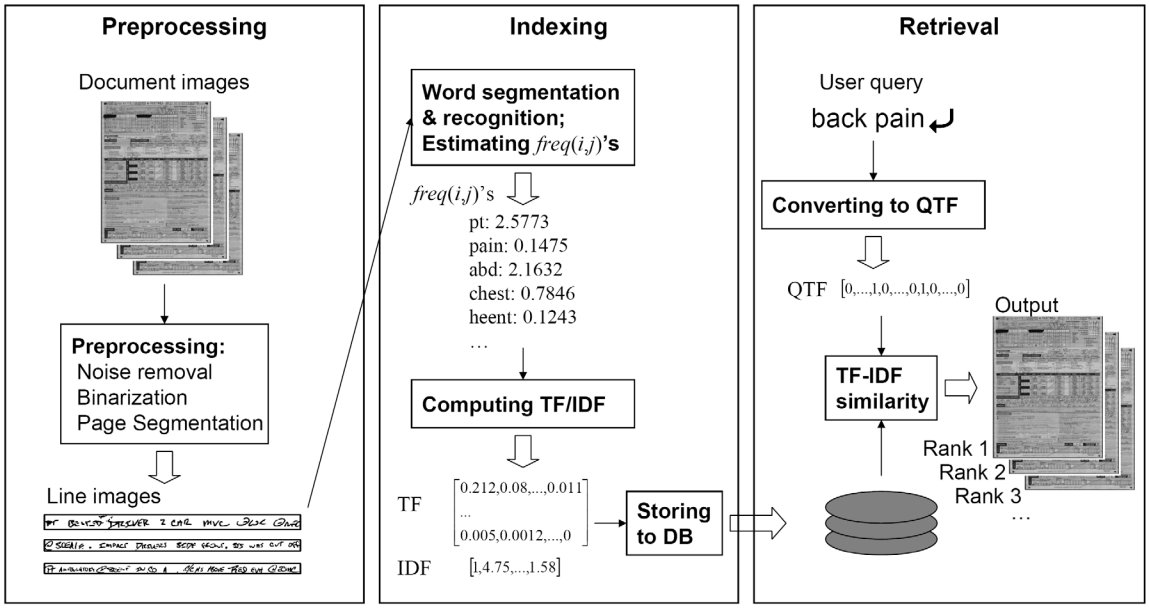


Figure 4.3. Flowchart of the search engine.

The flowchart of the search engine (Figure 4.3) shows three phases of the system: preprocessing, indexing, and document retrieval.

In the preprocessing phase, image enhancement such as noise filtering and binarization are performed, and text lines are identified by page segmentation.

Indexing includes word segmentation and recognition with the estimation of probabilities. We use these probabilities to estimate the term frequency (TF) and inverse document frequency (IDF) and store the estimated TF and IDF values for retrieval.

When searching the database for relevant documents, the user input query is converted to a query vector and the similarity of the Vector Model is calculated for each document. Documents are ranked in the decreasing order of similarity and top documents are returned.

4.2.7 Computational Issues

Only the non-zero values of the TF matrix are needed to be stored in the index and thus the space to store the index and time complexity of retrieval are both linear

$$\begin{bmatrix} 0 & 0 & 5 & 0 & ... & 0 \\ 0 & 1 & 0 & 0 & ... & 0 \\ 0 & 0 & 3 & 0 & ... & 0 \\ ... & ... & ... & ... & ... & ... \\ 0 & 2 & 1 & 0 & ... & 0 \end{bmatrix}$$

(a) The TF matrix from a text IR application

$$\begin{bmatrix} 0.001 & 0.005 & 4.82 & 0.0009 & ... & 0.002 \\ 0.005 & 0.75 & 0.001 & 0.0002 & ... & 0.03 \\ 0.0009 & 0.002 & 3 & 0.0007 & ... & 0.001 \\ ... & ... & ... & ... & ... & ... \\ 0.003 & 1.62 & 3.85 & 0.0008 & ... & 0.001 \end{bmatrix}$$

(b) The TF matrix from a document image IR application

Figure 4.4. TF matrices from text IR and document image IR. The TF matrix for document image IR can be approximated by a sparse matrix if we turn the shadowed elements that are below a threshold to 0.

in the number of non-zero values in the TF matrix. The TF matrix for text retrieval is usually sparse so the size of index file and the retrieval speed are not issues. But the TF matrix are no longer sparse when indexing document images (using the proposed method). Practically, we can convert the TF matrix into a sparse one without affect performance much: we can choose a threshold THR_{sparse}, and turn those elements from the TF matrix that are less or equal to THR_{sparse} (see shadowed elements in Figure 4.4 (b)). We set THR_{sparse} to 0.002 in our experiments.

4.3 Experimental Results and Discussions

4.3.1 Test Corpus

Our test corpus is the New York State Pre-hospital Care Reports (PCR forms). In New York State all patients who enter the Emergency Medical System (EMS) are tracked through their pre-hospital care to the emergency room using the PCR.

The PCR is used to gather vital patient information. Retrieval on this data set is quite challenging for several reasons: (i) handwritten responses are very loosely constrained in terms of writing style, format of response, and choice of text due to irrepressible emergency situations, (ii) images are scanned from noisy carbon copies and color background leads to low contrast and low signal-to-noise ratio (Figure 5.2), (iii) medical lexicons of words are very large (more than 4,000 entries). This leads to difficulties in the automatic transcription of forms. The word recognition rate of the forms using Word Model Recognizer (WMR) [33] is below 30%. Each PCR contains only about 100 handwritten words on average so the content is very short and ordinary IR methods perform badly since some of the terms are often absent from the OCR result.

4.3.2 Preprocessing and Recognition of PCR Form Images

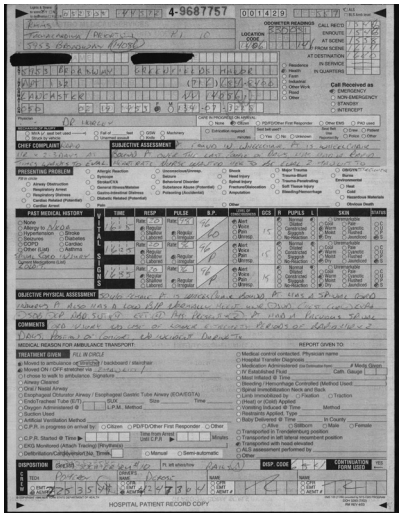
First we detect and remove the skew of every PCR form image as follows.

1. We manually de-skew a form and take it as a template. Two special regions are taken from the template as anchors.

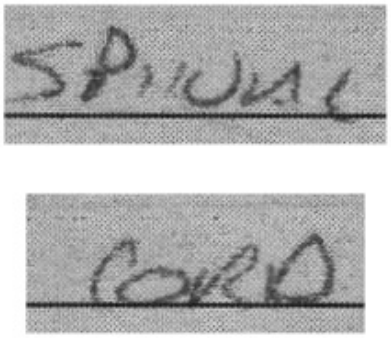
2. The positions of two anchoring regions in any test image are found by cross-correlation.

3. The skew angle of the test image is obtained by the relative skewing between the test image and the template. We de-skew the image by rotating to the opposite direction.

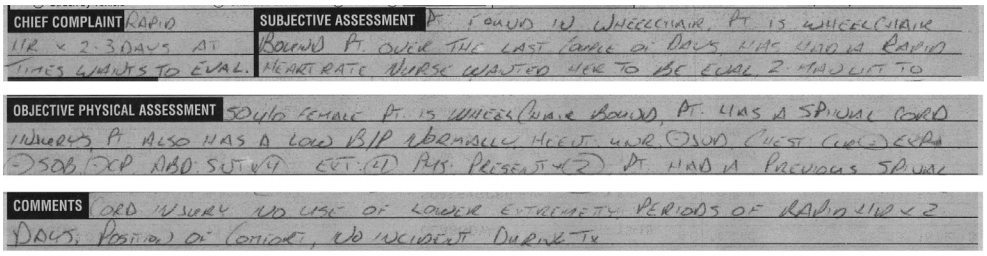
By aligning the test image to the template image, we can also obtain the position of each form cell containing a line of text. The template-matching based de-skewing and page segmentation work well on the PCR form images since they have a fixed layout and are scanned at the same resolution. Our approach is applicable to other types of forms as well.



(a)

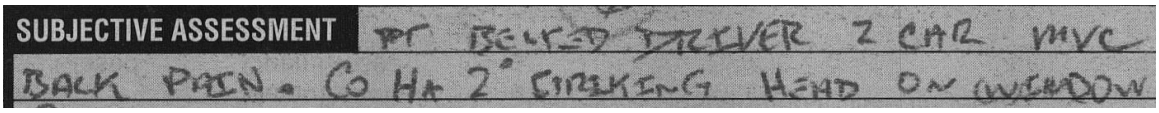


(b)

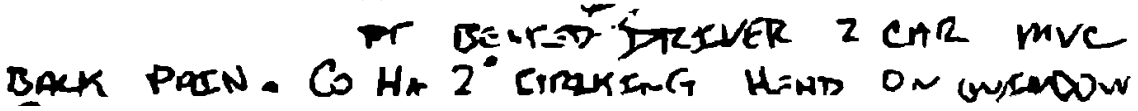


(c)

Figure 4.5. An example of PCR forms. (a) A entire PCR form. (b) A small local region showing obscure text and background noise array. (c) Fields of interest in the PCR form.



(a) The original grayscale image.



(b) The bnarized image. Grid lines are removed and broken strokes are fixed.

Figure 4.6. An example of the binarization and line removal result.

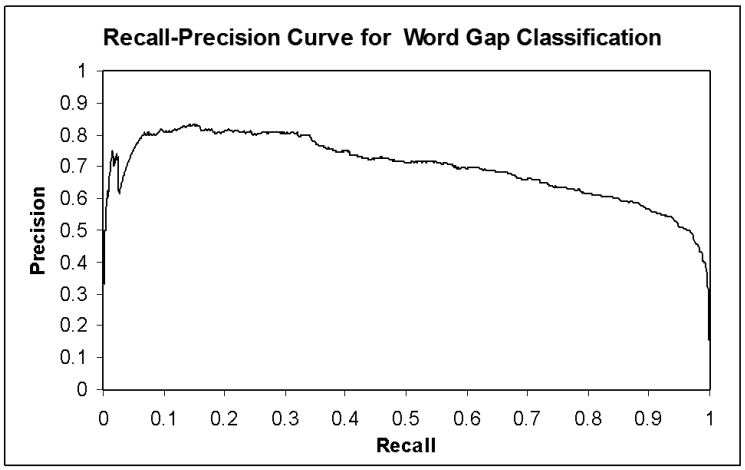


Figure 4.7. The performance of word segmentation (recall-precision curve).

We use the MRF based document image preprocessing algorithm [11] to binarize the form image and remove the grid lines from the image. Assuming the binarized objective image is x and the grayscale image is y, we solve the maximum a posteriori (MAP) estimation $\hat{x} = \underset{x}{\operatorname{argmax}} \Pr(x|y)$ using the Markov Random Fields (MRF). An example of binarization and line removal result is shown in Figure 5.3. The MRF based preprocessing method improves the word recognition accuracy from 18.7% (obtained by the PCR form preprocessing algorithm in [42]) to 28.6%.

We use 1099 valid word gaps and 5138 non-valid word gaps to train the word gap classifier using the method presented in Section 5.2.2. The classifier is evaluated on a test deck of 791 valid word gaps and 4369 non valid word gaps. If we take probability

p_{thr} as a threshold to determine the validity of a gap, we can compute the recall and precision values obtained from the given test deck. Thus a precision-recall curve (Figure 4.7) is obtained by taking various values of threshold, p_{thr}.

The WMR handwritten word recognizer is trained using 21054 character images extracted from images of the US Postal Service database. A lexicon of 4670 English words is generated from the ground truth of 783 PCR forms. We also learn the prior probabilities and bi-gram model from these 783 forms. A word recognition rate of 28.6% is obtained on the PCR forms.

4.3.3 Evaluation Metrics of IR Test

The IR tests are evaluated in terms of Mean Average Precision (MAP) and R-Precision [1]. The Mean Average Precision is obtained in the following way:

1. For each query, check the returned documents starting from rank 1. Whenever a relevant document is found, record the precision of the documents from the one with rank 1 to the current one. The average value of the recorded precisions for the query is the Average Precision of the query.

2. The mean value of the Average Precisions of all the queries is the Mean Average Precision of the test.

R-Precision of a query is the mean value of precisions computed for each query when R documents are retrieved, where R is the number of relevant documents. The mean value of the R-Precisions of all queries is the R-Precision of all of the queries. For example, suppose 100 documents are relevant to query q_1, and 30 of the top 100 retrieved documents are relevant to the query, then the R-Precision of query q_1 is $30/100 = 30\%$. Suppose the R-Precision of another query q_2 is 20%, then the R-Precision of q_1 and q_2 is $(30\% + 20\%)/2 = 25\%$.

Table 4.2. 28 query phrases used in our IR tests.

"head pain"	"emesis"	"breath difficulty short"
"trachea"	"lung"	"chest pain"
"fracture"	"rib fracture"	"head fracture"
"ankle fracture"	"cancer"	"trauma"
"glucose"	"diabetes"	"foot"
"tender"	"hurts"	"ambulate"
"cardiac"	"dizzy dizzyness dizziness"	"cardiac monitor"
"wrist"	"arthritis"	"shoulder pain"
"syncope"	"mri"	"blind"
"dementia"		

In addition to the Mean Average Precision and R-Precision, the performance of the IR system can also be visualized using a 11-point precision. First, the 11 interpolated precisions at recalls 0, 0.1, ..., 1 are calculated for each query. Then the average precision of all of the queries at each of the 11 recalls is calculated. Finally we get 11 precisions.

4.3.4 IR Tests

The document images involved in our IR tests are 342 PCR forms with manually transcribed ground truth and coordinates of each word. We have 28 queries, and manual annotation of relevance of the 342 forms to these queries. An example of an entire PCR form and handwritten regions of interest in the PCR form are shown in Figure 5.2(c). The queries used in our IR tests are shown in Table 4.2

We compare the performances of the following 7 IR tests:

Tests 1-4: IR tests on OCR'ed text

We apply the classic Vector Model on OCR'ed text. First we apply word segmentation to the 342 form images as follows. For any m ($m \leq 16$) consecutive connected components $c_q c_{q+1} ... c_{q+m}$, suppose $\sigma_{q-1}, \sigma_q, ...,$ and σ_{q+m} are gap validity probabilities obtained by the gap classification algorithm presented in

Section 5.2.2, then the probability of the concatenation $c_q c_{q+1} ... c_{q+m}$ being a word image is $\sigma_{q-1} \cdot (1 - \sigma_q) \cdot ... \cdot (1 - \sigma_{q+m-1}) \cdot \sigma_{q+m}$.

We recognize all the word images with the word segmentation probability above 0.3. The OCR'ed text is composed of the top-S word recognition candidates of every word image. The parameter $S = 1$, 3, 7, and 15 in four separate tests. IR tests based on the Classic VM are performed on the OCR'ed text of 342 form images.

Test 5: Vector IR Model + HR Estimation

We apply the Modified Vector Model to 342 form images for document retrieval. The raw term frequencies are estimated from handwriting recognition (HR) results using Equation (4.13) by assuming perfect word segmentation and identical independent distribution (i.i.d.) of terms, *i.e.*,

$$E\{freq_{i,j}\} = \sum_{k=1}^{L} \Pr(\tau_k | w_k) \tag{4.36}$$

We use the same word segmentation method in Test 1-4.

Test 6: Probabilistic IR Model + Isolated Word Estimation

We apply the probabilistic IR model [49, 24] to 342 form images for document retrieval. In this model, the doc-query similarity is defined as

$$\text{sim}(d_j, q) = \prod_{1 \le i \le N,\ tf_{i,q}=1} tf_{i,j}, \tag{4.37}$$

and the raw frequency is estimated by Equation (4.36). We use the same word segmentation method in Test 1-4. The difference between [49, 24] and our implementation is the way word recognition probabilities $\Pr(\tau_k | w_k)$ are estimated.

Test 7: Vector IR Model + Word Sequence Estimation

We apply the Modified Vector Model to 342 form images for document retrieval. The raw term frequencies are obtained by the word sequence based

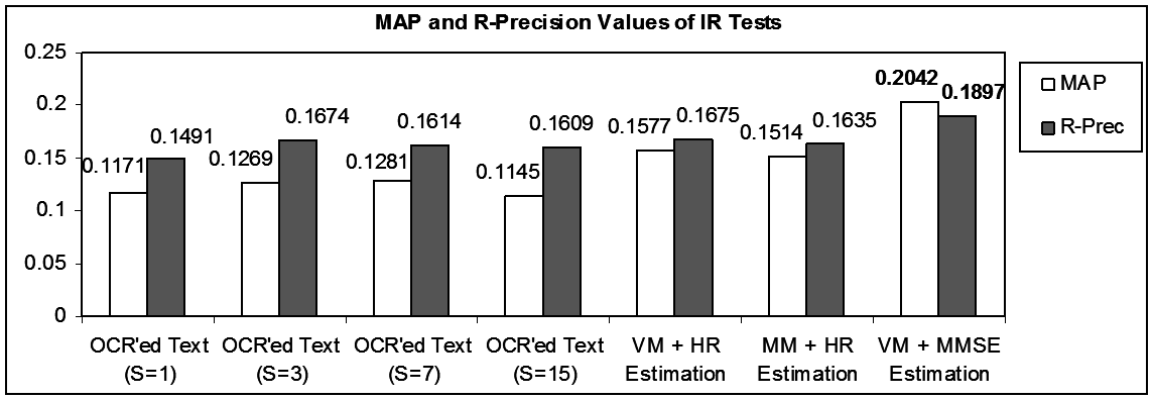
estimation using Equations (4.16) - (4.22).

The MAP and R-Precision values of the above IR tests are compared in Figure 4.8-(a). A trivial average precision of 4.76% is obtained by generating random retrieval results for the 28 queries. We amend the metrics by subtracting the trivial AP from the MAP and R-Precision values. The amended metrics show the incremental improvement from the trivial result. The amended MAP and R-Precision values of the above IR tests are compared in Figure 4.8-(b). Tests 1-4 show that the improvement of using more word recognition candidates (S=3, 7, and 15) compared to the result of IR test on top-1 word recognition text is very slight. Even a naive estimation of the raw term frequencies (Equation (4.36)) improves the IR performance compared to the tests based on OCR'ed text. But the use of the word segmentation probabilities and the language model (Test 7) resulted in better IR performance than the estimation method that only uses isolated word recognition results.

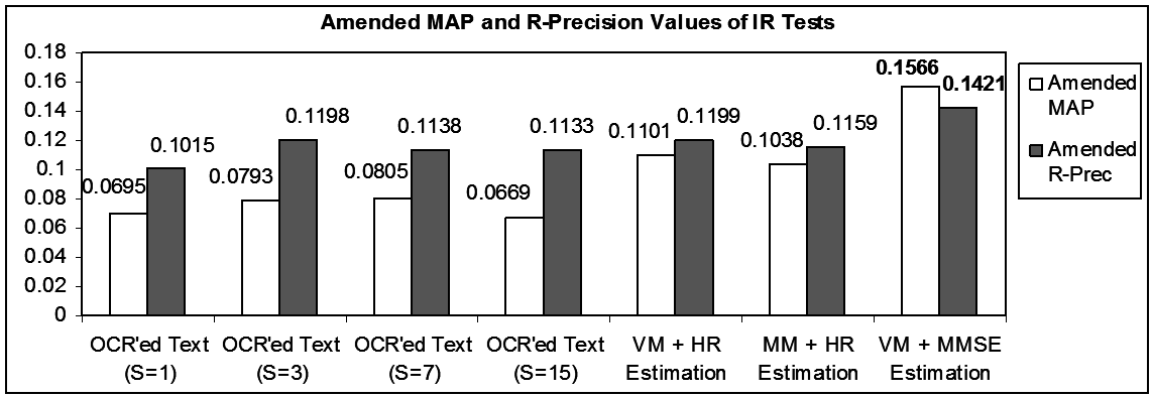
The interpolated 11-point precision curves of tests 1 (OCR'ed text, $S = 1$), 5 (VM + isolated word estimation) and 7 (VM + word sequence estimation) are shown in Figure 4.9 (a). The IR performance of building the index on the ground truth is also shown in Figure 4.9 (a). Tests 5 and 7 produce similar precisions at low recall (around 0) but Test 7 produces significantly higher precisions at higher recalls.

For better comparison, the above 11-point precision curves can also be amended this way: we first get two addition precisions at each recall level: trivial precision and ground-truth precision, and then normalize the recall-precision coordinates so that the trivial precision is always 0 and the ground-truth precision is always 1. The trivial precision is defined as the precision obtained by ranking all the documents randomly:

$$Prec_{trivial} = \frac{\text{average number of relevant documents per query}}{\text{number of documents}} \tag{4.38}$$



(a) Original MAP's and R-Precisions.



(b) Amended MAP's and R-Precisions.

Figure 4.8. The MAP and R-Precision values of 7 IR tests.

Table 4.3. Approaches to handwritten document retrieval.

	Test 1	Test 5	Test 7
Size of Indexing file	17.0 MB	22.5 MB	74.4 MB
Retrieval Speed	119 queries/sec	83 queries/sec	47 queries/sec

The ground-truth precision $Prec_{truth}$ at a recall level is the precision obtained by IR test performed on the index built on ground-truth text. The amended precision of an original precision p is defined as

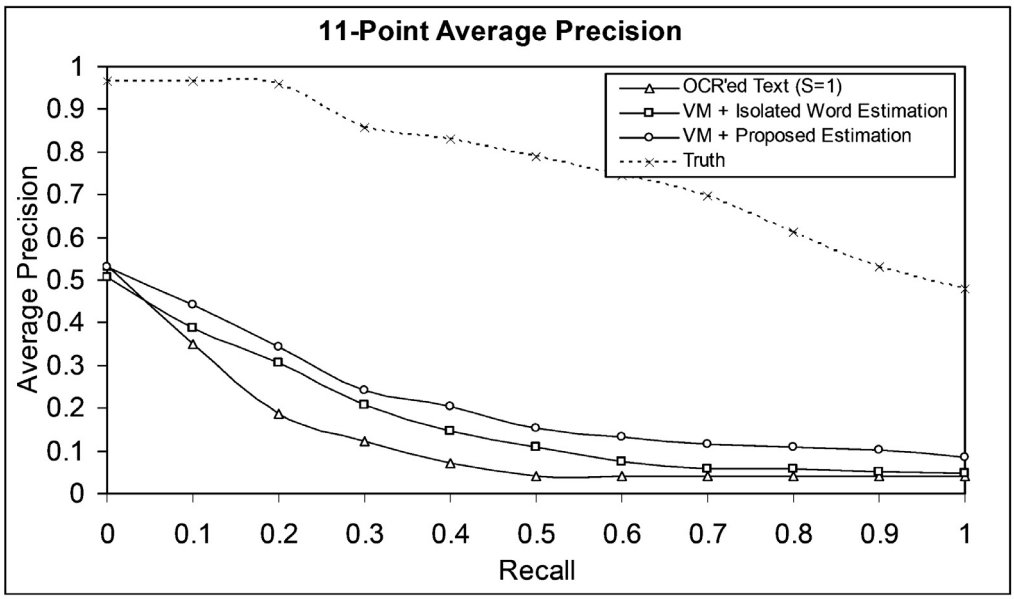
$$Prec_{amended} = \frac{p - Prec_{trivial}}{p - Prec_{truth}} \times 100\% \tag{4.39}$$

The amended 11-point precision curves in Figure 4.9 (b) shows that the proposed method obtained improvement at almost all recall levels but especially improved the precisions at high recall rates (¿50%). The two existing methods perform very poorly at high recall levels by giving nearly zero precisions. But the proposed method still obtained about 10% precision at the recall level of 100%.

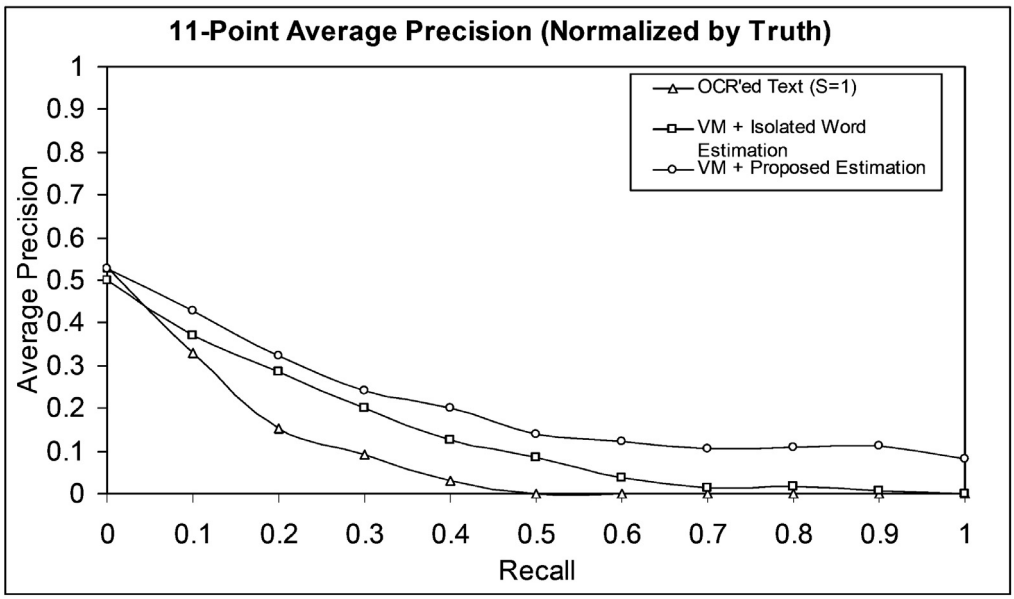
The sizes of the indexing files and retrieval speeds of the above three tests are compared in Table 4.3. From Test 1 to Test 7, as we used more recognition and segmentation hypothesis, the increased requirements of space and running time of retrieval are still acceptable in practice: the space required to store the index increased about 4 times and the running time increased about 2 times.

4.4 Summary

This chapter presents a vector model based method for indexing and retrieval of handwritten document images. Instead of finding the best transcription (which is the objective of handwriting recognition), tracking and weighting all possible transcriptions is more important to the indexing and retrieval of handwritten documents. We improve the term-weighting scheme of existing IR techniques by estimating the raw



(a) Original recall-precision curves.



(b) Amended recall-precision curves.

Figure 4.9. The 11-point average precision curves of tests 1, 5 and 7.

term frequencies using the MMSE criteria. The MMSE estimation of raw term frequencies integrates word segmentation, word segmentation and language model into a statistical approach. Our work is validated by the improvement of IR performance compared with other term-weighting schemes.

CHAPTER 5

HANDWRITTEN KEYWORD RETRIEVAL

5.1 Introduction

Keyword retrieval in document image is generally referred to as *word spotting*. There can be two approaches to keyword retrieval in the handwritten document images. In the first approach, we can first perform a handwriting recognition followed by the indexing step to keep track of the transcription and other useful information (positions and recognition scores of word images) [12, 49, 24]. Retrieval can be performed by some measurements of similarity between the keyword and the word image.

In the second approach [31, 63, 38, 26, 35, 57, 39, 27], the index of word images can be built from images features. During retrieval, each keyword is converted into a word image. This can be done by annotating a small set of word images designated for generating query images. The generated query image is compared to the word images in the database. The similarity between them is measured using a certain distance between the feature vectors of the word images. When a user provides a query word, the similarity between the query and the word image in the database is computed, and word images are returned in the decreasing order of similarities. In [31, 38], the similarity between the feature vectors of two word images is achieved by Dynamic Time Warping (DTW) matching of profile features computed using various definitions of distances [31, 4, 51, 38] in the feature space. Similarity [63] is based on bitwise matching of the GSC features of two word images. The second approach is referred to as "word spotting". As we know a handwriting system is not easy to implement.

The word spotting methods can be an alternative at very low cost of implement when a handwriting recognition system is not available.

However, word spotting requires on-line matching which is time-consuming. Trade-off between accuracy and speed has to made in order to be scalable to large database. Thus Image feature based indexing approaches are limited in feature selection and the complexity of matching and training methods and are only applicable to constrained handwriting such as when dealing with a single writer or small lexicons. In contrast, OCR score based indexing approaches [12, 49, 24] conquered the speed problem. In these methods, the indices are built from OCR scores such posterior probabilities or feature vector observational likelihoods (probability density) converted from distances returned by word recognizer. In general, The first approach provides much more accuracy than the second approach.

Another question is whether to adopt a word-lexicon. The index for fast retrieval can be built on the results of word level recognition in lexicon-driven mode [49, 24]. In the lexicon driven mode, any word that is not in the lexicon can't be retrieved. Thus one need to select the set of keywords to meet the requirement of the application. [12] performs word recognition on character level and search for optimal matches in the series of character recognition scores. However searching in character recognition scores requires additional time and is time-consuming for large-scale data. We adopt a word-lexicon-driven method in this thesis for maximum accuracy and efficiency but sacrifice some loss due to the OOV problem.

We improve the OCR score based indexing method by integrating word segmentation probabilities into the ranking scheme of word images. Word spotting methods mentioned above assume perfect word segmentation: word images are given by word segmentation algorithm, and the ranks of word images are obtained by ordering the word recognition scores. However it is difficult to get a nearly perfect word segmentation in unconstrained handwriting due to irregular variation of the word gaps. Thus,

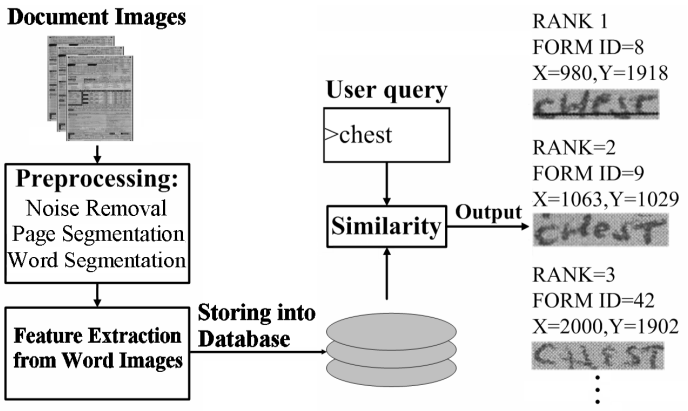


Figure 5.1. Diagram of the keyword spotting system.

the performance of word spotting can be improved by modeling word segmentation probabilities. In this chapter, we describe a probabilistic model of word spotting that integrates word segmentation probabilities and word recognition probabilities. The word segmentation probabilities are obtained by modeling the conditional distribution of multivariate distance features of word gaps. The word recognition results are also represented by a probabilistic model so that they are compatible with the probabilistic word spotting model. The modeling of the word recognition probabilities is obtained from the distances returned by the word recognizer.

5.2 Model Definition

5.2.1 Word Spotting Model

Given a series of consecutive connected components $c_1, c_2, ..., c_n$ and a possible word image w represented by $c_i, c_{i+1}, ...c_j$ $(1 \leq i, j \leq n)$, then the similarity between w and a query word q can be represented by

$$\begin{aligned} &\text{sim}(w, q) = \\ &\sigma_{i-1} \cdot (1 - \sigma_i) \cdot ... \cdot (1 - \sigma_{j-1}) \cdot \sigma_j \cdot \Pr(q|w) \end{aligned} \tag{5.1}$$

where σ_k $(1 \leq k \leq n-1)$ is the probability of the gap between c_{k-1} and c_k being a valid word gap, $\sigma_0 = \sigma_n = 1$, and $\Pr(q|w)$ is the word recognition probability. Here we assume the gaps are independent and thus the word segmentation probability is $\sigma_{i-1} \cdot (1-\sigma_i) \cdot ... \cdot (1-\sigma_{j-1}) \cdot \sigma_j$.

The size required to store the index can be reduced by applying constraints on the number of connected components within a word image and minimum similarity:

Given a series of consecutive connected components $c_1, c_2, ..., c_n$,

For i from 1 to n

For j from i to $\min(i + C_{max} - 1, n)$

If the similarity $\text{sim}(c_i...c_j, q) > \text{sim}_{min}$

Then store the document number, coordinates of the word image and the similarity into index.

C_{max} is the maximum number of connected components within a word image. sim_{min} is the minimum similarity that can be stored into the index. We assume $C_{max} = 16$ and $\text{sim}_{min} = 0.1\%$ in our experiment.

5.2.2 Estimating Word Segmentation Probability

Word segmentation is defined as the process of segmenting a line into words. In handwritten lines, the space between words is uneven. Moreover, the same amount of space may be present between words, and between characters within a word. Such cases arise due to differences in writing styles, and space constraints.

In our word segmentation method [9], the probability of the gap between any two consecutive connected components being a valid word gap is estimated from feature space of distance measures. A gap between two connected components is represented by three distance features:

1. **Euclidean Distance.** This feature is defined as the horizontal distance between the bounding boxes of the two consecutive connected components of the line image (Figure 4.1(a)).

2. **Minimum Run Length.** This feature represents the minimum horizontal white run length distance between the two adjacent connected components of the line image.

3. **Convex Hull Distance.** We compute the convex hulls of two consecutive connected components and draw a line connecting the mass centers of the two convex hulls. The Euclidean distance between points at which this line crosses the two convex hulls is defined as the Convex Hull distance of the two adjacent components.

To eliminate the effect of different text sizes, we compute the average height of all the components and normalize the extracted features by dividing them by the average height of all components in the same line.

The segmentation probability of a gap g is given by the Bayes' Rule

$$\sigma_g = \Pr(g|f_{1,g}, f_{2,g}, f_{3,g}) = \frac{\Pr(g)p(f_{1,g}, f_{2,g}, f_{3,g}|g)}{\Pr(g)p(f_{1,g}, f_{2,g}, f_{3,g}|g) + \Pr(\bar{g})p(f_{1,g}, f_{2,g}, f_{3,g}|\bar{g})} \tag{5.2}$$

where $\Pr(g)$ and $\Pr(\bar{g})$ are the prior probabilities of valid gaps and non-valid gaps, respectively. $f_{1,g}$, $f_{2,g}$ and $f_{3,g}$ are three features of g. $p(f_{1,g}, f_{2,g}, f_{3,g}|g)$ is the probability density of the features of valid gaps. $p(f_{1,g}, f_{2,g}, f_{3,g}|\bar{g})$ is the probability density of the features of non-valid gaps.

Given a set of gap features with the annotation of *"valid"* and *"non-valid"*, we can estimate $\Pr(g)$, $\Pr(\bar{g})$, $p(f_{1,g}, f_{2,g}, f_{3,g}|g)$ and $p(f_{1,g}, f_{2,g}, f_{3,g}|\bar{g})$ as follows. $\Pr(g)$

and $\Pr(\bar{g})$ are estimated from the ratio of the numbers of valid and non-valid gaps in the training set.

$$\Pr(g) = \frac{\#\{\text{valid gaps}\}}{\#\{\text{valid gaps}\} + \#\{\text{non-valid gaps}\}} \tag{5.3}$$

$$\Pr(\bar{g}) = 1 - \Pr(g) \tag{5.4}$$

$p(f_{1,g}, f_{2,g}, f_{3,g}|g)$ and $p(f_{1,g}, f_{2,g}, f_{3,g}|\bar{g})$ are estimated non-parametrically using Parzen window technique with a Gaussian kernel function.

5.2.3 Estimating Word Recognition Probability

We use a lexicon-driven word recognition algorithm [33] that performs character segmentation and finds the best matching path using dynamic programming. First a word image is segmented into candidate character images. Then the directional features are extracted from the contours of character images and matched to every word in the lexicon by searching all possible segmentations for the minimum sum of Euclidean distances from the features of the test image and the character templates in the training set. The minimum Euclidean distance indicates the similarity between the word image and the term in the lexicon. The square of the distance associated with a pair of a word image w and a term t_i is denoted by $s(w, t_i)$.

The word recognition probability is estimated from the recognition score using a Universal Background Model (UBM) [50, 9]. In a Background Model, the posterior probability of the word recognition is given by Bayes' rule:

$$\Pr(w = t_i|s(w,t_i)) = \frac{\Pr(w = t_i)p_{t_i}(s(w,t_i)|w = t_i)}{\Pr(w = t_i)p_{t_i}(s(w,t_i)|w = t_i) + \Pr(w \neq t_i)p_{t_i}(s(w,t_i)|w \neq t_i)} \tag{5.5}$$

where $p_{t_i}(s(w,t_i)|w = t_i)$ is the likelihood of the genuine matching score when the word is t_i, $p_{t_i}(s(w,t_i)|w \neq t_i)$ is the likelihood of the imposter matching score when

the word is t_i, and $\Pr(w = t_i)$, $\Pr(w \neq t_i)$ are the prior probabilities of genuine and imposter matches of t_i, respectively.

We need a term specific training set for every term to learn the background model. This is a drawback in applications using large number of terms. The Universal Background Model is an alternative approach that solves this problem. In the UBM, we use a single Background Model for all of the terms. The genuine matching probability is given by

$$\Pr(Genuine|s) = \frac{\Pr(Genuine)p(s|Genuine)}{\Pr(Genuine)p(s|Genuine) + \Pr(Imposter)p(s|Imposter)} \tag{5.6}$$

where s is a matching score, and $\Pr(Genuine)$ and $\Pr(Imposter)$ are the prior probabilities of genuine match and imposter match, respectively, and $p(s|Genuine)$, $p(s|Imposter)$ are the likelihoods of the score of genuine match and imposter match, respectively. $\Pr(Genuine)$, $\Pr(Imposter)$, $p(s|Genuine)$, and $p(s|Imposter)$ are estimated from the scores of all of the terms. We model $p(s|Genuine)$ and $p(s|Imposter)$ as Gamma distributions.

5.3 Experimental Results

5.3.1 Data Collection

Our keyword retrieval algorithm has been tested on the New York State Prehospital Care Reports (PCR forms). The task is quite challenging for several reasons: (i) handwritten responses were very loosely constrained in terms of writing style, format of response, and choice of text due to irrepressible emergency situations, (ii) images are scanned from carbon copies and are very noisy (Figure 5.2), (iii) medical lexicons of words are very large (more than 4,000 entries). This leads to difficulties in the automatic transcription of forms. The word recognition rate of the forms using word recognizer [33] is about 20-30%.

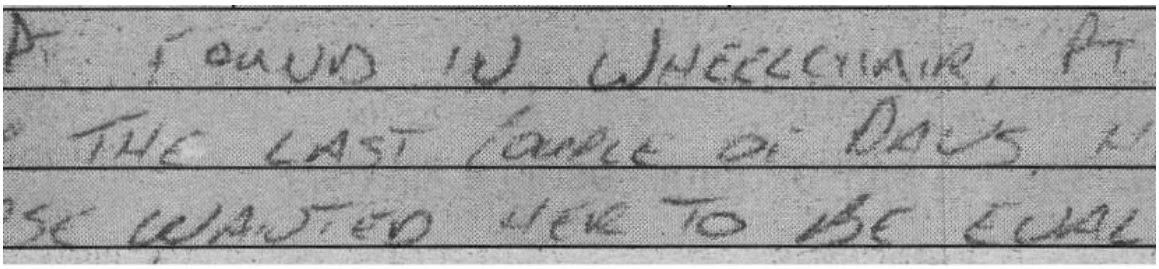


Figure 5.2. The text in a PCR form.

5.3.2 Preprocessing

First we detect and remove the skew of every PCR form image as follows.

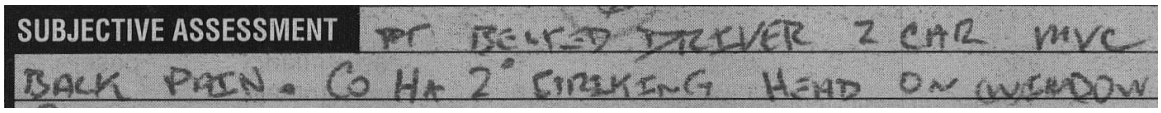
1. We manually de-skew a form and take it as a template. Two special regions are taken from the template as anchors.

2. The positions of two anchoring regions in any test image are found by cross-correlation.

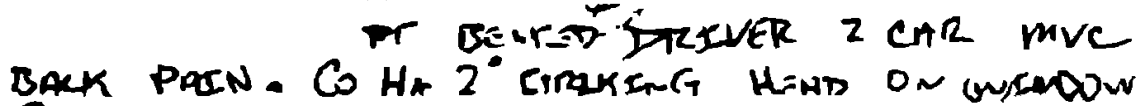
3. The skew angle of the test image is obtained by the relative skewing between the test image and the template. We de-skew the image by rotating to the opposite direction.

By aligning the test image to the template image, we can also obtain the position of each form cell containing a line of text. The de-skewing and page segmentation method using template-matching works well on the PCR form images since they have a fixed layout and are scanned at the same resolution. Our approach is applicable to other types of forms as well.

We use the MRF based document image preprocessing algorithm [11] to binarize the form image and remove the grid lines from the image. Assuming the binarized objective image is x and the grayscale image is y, we solve the maximum a posteriori (MAP) estimation $\hat{x} = \underset{x}{\operatorname{argmax}} \Pr(x|y)$ using the Markov Random Fields (MRF).



(a) The original grayscale image.



(b) The bnarized image. Grid lines are removed and broken strokes are fixed.

Figure 5.3. An example of the binarization and line removal result.

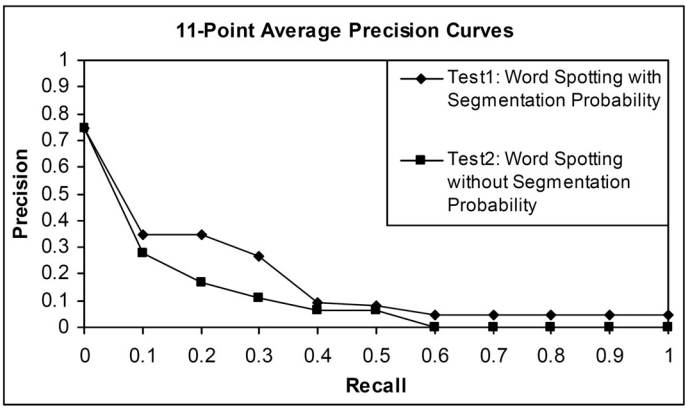


Figure 5.4. 11-point average precision curves of Tests 1-2.

An example of binarization and line removal result is shown in Figure 5.3. The MRF based preprocessing method improves the word recognition accuracy from 18.7% (obtained by the PCR form preprocessing algorithm in [42]) to 28.6%.

5.3.3 Evaluation Metrics

The performance of word spotting can be evaluated using the precisions at standard recall levels $(0, 0.1, ..., 1)$. We may also use single value measures such as the Mean Average Precision (MAP) [1] to evaluate the word spotting performance. The Mean Average Precision is computed as follows:

1. For each query, check the returned word images starting from rank1. Whenever a relevant word image is found, record the precision of the word images from the one with rank 1 to the current one. The Average Precision (AP) of a given

query q is weighted sum of the recorded precisions:

$$\mathrm{AP}(q) = \frac{\sum_{1 \le r \le N_q, Rel(r) \text{ is true}} Prec(r)}{R_q} \tag{5.7}$$

where N_q is the number of word images returned, R_q is the number of relevant documents, $Prec(r)$ is the precision of top-r returned word images, and $Rel(r)$ is a Boolean function of the relevance of rank r.

2. The Mean Average Precision (MAP) of all the queries is:

$$\mathrm{MAP} = \frac{\sum_q R_q \times \mathrm{AP}(q)}{\sum_q R_q} \tag{5.8}$$

5.3.4 Keyword Retrieval Results

We performed two word spotting tests to show the improvement due to the use of word segmentation probability.

Test 1: Word spotting with segmentation probability

We searched 342 PCR forms for 33 keywords using the similarity function in Equation (5.1) and the estimation methods described in Section 5.2.

Test 2: Word spotting without segmentation probability

In this test we evaluate the performance of word spotting using connected component clustering based word segmentation method without probability annotation. This test is based on the same idea used by [24] but we implemented different word segmentation and recognition method. We performed word spotting test on the same PCR forms and keywords that we performed on in Test 1. The word images in test 2 were obtained by grouping adjacent connected components with

gap segmentation probability $\sigma_k < C_{gap}$. The cutoff threshold $C_{gap} = 0.297$ in our test. The recall rate of the gap classification reached maximum (0.394) when $C_{gap} = 0.297$. We used the same word recognition method that we used in Test 1. The word recognition probability returned by the Universal Background Model (UBM) is taken as the similarity between the word image and the query.

The 11-point average precision curves of Tests 1-2 are shown in Figure. The Mean Average Precision (MAP) scores of Test 1 and Test 2 are 4.7% and 2.8%, respectively. The test results show the improvement obtained by using word segmentation probabilities.

5.4 Summary

This chapter describes a method to search handwritten document images for keywords. The image/text similarity of the proposed method is defined as the product of word segmentation probability and word recognition probability. Test results show the improvement of integrating the probabilistic annotation of word segmentation on handwritten document images with word segmentation errors.

CHAPTER 6

CONCLUSION

6.1 Contributions of the Thesis

This thesis described the methodologies and outcomes we have obtained on the research of degraded handwritten document preprocessing and retrieval. We proposed and validated a series of methods to solve problems in the following respects:

1. In Chapter 3, we described a method for extracting binarized text from low-quality carbon form handwritten input. It also incorporated pre-printed ruling-line removal and inpainting. We tested our method on two data-sets: PCR (real degraded images which are the main data-set for our purpose of testing the preprocessing and retrieval) and IAM DB (handwritten document images with artificially added Gaussian noise.) For the PCR data, our preprocessing method obtained remarkable gain of word recognition accuracy from below 20% to 28.6% comparing to existing methods (Otsu, Niblack, and Milewski). Improvement of word recognition accuracy is also obtained on the IAM data with synthetic noise.

2. In Chapter 4 and 5, we explored methods of improving IR performance on handwritten documents. The IR tasks include document retrieval and keyword retrieval. In Chapter 4, we described a method to improve the estimation of the term frequency from document images for better retrieval performance. Different from text retrieval, the term frequency of a document image is not immediately available without document analysis and recognition. The traditional way of indexing and retrieving document images is to build index on the

OCR'ed text returned by OCR software which is treated as a black-box. In our method, we use the word segmentation and recognition scores to the maximal extent. First the word segmentation/recognition scores are converted into probabilistic representations. Then the term frequency is estimated from the word segmentation and recognition probabilities and a language model (n-gram.) The estimated term frequency is incorporated with standard IR techniques such as vector model for retrieval of the PCR data-set.

In Chapter 5, we applied the probability modeling of word segmentation directly to the scheme of computing keyword retrieval similarity and obtained improved performance comparing with the traditional method that neglects the evaluation of word segmentation outcomes.

The contributions of this thesis can be summarized as follows:

- **The first work to apply MRF to the binarization of high-resolution document images.**
- **Use of speeding-up strategies which are proper in the MRF configuration designed for the banirazation problem:**
  - Belief Propagation (BP)
  - Vector quantization of the states of image patch
  - Pruning of the search space of BP
- **Ruling-line removal using the MRF.**
- **A modified vector IR model designated for handwritten document retrieval**
- **Probabilistic modeling of word segmentation and recognition scores which is incorporated with MMSE estimation of raw term frequency**

- **Use of word segmentation probabilities for high keyword retrieval performance**

6.2 Future Works

Our MRF preprocessing method can only work the the model learned from images of the same or similar resolution. Use of the trained model on re-sampled images will lead to inaccurate results. Practically, we can learn the MRF models at multiple image resolutions and automatically select the model that fits the input image during preprocessing. For example, we may want to use down-sampled scanned image to train the MRF and apply it to video text. We will investigate the method of estimating the resolution and selecting the model in the future work.

In our thesis, we improved the IR performance by using all possible information provided by the handwriting recognition systems (Top-n word recognition choices, multiple segmentation points). In addition to handwritten document retrieval, there may be other applications such as Document Categorization and Machine Translation (MT) that are also dependent on the handwriting recognition result.

Document categorization is to assign a document to one or more categories based on the content of the document. The document categorization techniques are similar to document retrieval in terms of the way to index the keywords in the documents. The TF-IDF representation can also be used. It will be an interesting topic to apply the techniques proposed in this thesis to the document categorization problem.

Machine translation is to use the computer to translate speech or text from one natural language to another. In the automatic transcription of handwriting, we will be struggling with the recognition errors and the translation will be unreliable. Similar idea of using multiple word recognition and segmentation choices may also be able to apply to machine translation. However this is not very straightforward to implement because, unlike the document retrieval and categorization problems, machine

translation is modeled differently and the estimation of term frequencies is not the kernel problem as it is in the other two applications. We will be investigating these new methods in the future works.

BIBLIOGRAPHY

[1] Baeza-Yates, Ricardo A., and Ribeiro-Neto, Berthier A. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[2] Bai, Zhen-Long, and Huo, Qiang. Underline detection and removal in a document image using multiple strategies. *Proceedings of International Conference on Pattern Recognition 2*, 578–581.

[3] Beitzel, Steven M., Jensen, Eric C., and Grossman, David A. A survey of retrieval strategies for ocr text collections. In *Proceedings of the Symposium on Document Image Understanding Technologies* (Greenbelt, Maryland, April 2003).

[4] Belonge, S., Malik, J., and Puzicha, J. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24* (2002), 509–522.

[5] Bertalmio, M., Sapiro, G., Caselles, V., and Ballester, C. Image inpainting. *Computer Graphics (SIGGRAPH 2000)* (2000), 417–424.

[6] Bozinovic, R.M., and Srihari, S.N. Off-line cursive script word recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence 11*, 1 (1989), 68–83.

[7] Bunke, H., Roth, M., and Schukat-Talamazzini, E.G. Off-line cursive handwriting recognition using hidden Markov models. *Pattern Recognition 28*, 9 (1995), 1399–1413.

[8] Cao, H., Farooq, F., and Govindaraju, V. Indexing and retrieval of degraded handwritten medical forms. In *Proceedings of the Workshop on Multimodal Information Retrieval at IJCAI-2007* (2007).

[9] Cao, H., and Govindaraju, V. Handwritten document retrieval using MMSE estimation of raw term frequencies. *submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence*.

[10] Cao, H., and Govindaraju, V. Preprocessing of low quality handwritten documents using markov random fields. *to appear in IEEE Transactions on Pattern Analysis and Machine Intelligence*.

[11] Cao, H., and Govindaraju, V. Handwritten carbon form preprocessing based on markov random field. *Proceedings of the 2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'07)* (2007).

[12] Cao, H., and Govindaraju, V. Template-free word spotting in low-quality manuscripts. In *the Sixth International Conference on Advances in Pattern Recognition (ICAPR)* (Calcutta, India, 2007), vol. 5296, pp. 45–53.

[13] Cao, H., and Govindaraju, V. Vector model based indexing and retrieval of handwritten medical forms. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition* (2007).

[14] Cao, H., and Govindaraju, V. Processing and retrieving handwritten medical forms. *Proceedings of the Digital Government Conference (DG.O)* (2008).

[15] Chen, W.-T., Gader, P., and Shi, H. Lexicon-driven handwritten word recognition using optimal linear combinations of order statistics. *IEEE Transactions on Pattern Analysis and Machine Intelligence 21*, 1 (1999), 77–82.

[16] Choi, S. C., and Wette, R. Maximum likelihood estimation of the parameters of the gamma distribution and their bias. *Technometrics 11*, 4 (April 1969), 683–690.

[17] Croft, W. B., Harding, S. M., Taghva, K., and Borsack, J. An evaluation of information retrieval accuracy with simulated OCR output. In *Proceedings of the Symposium on Document Analysis and Information Retrieval* (1994).

[18] Freeman, W. T., and Pasztor, E. C. Learning low-level vision. *Proc. of International Conference on Computer Vision* (1999), 1182–1189.

[19] Freeman, W. T., Pasztor, E. C., and Carmichael, O. T. Learning low-level vision. *International Journal of Computer Vision 40*, 1 (2000), 25–47.

[20] Geman, S., and Geman, D. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence 6*, 6 (1984), 721–741.

[21] Govindaraju, V., and Cao, H. Indexing and retrieval of handwritten medical forms. *Proceedings of the Digital Government Conference (DG.O)* (2007).

[22] Gupta, M. D., Rajaram, S., Petrovic, N., and Huang, T. S. Restoration and recognition in a loop. *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* (2005).

[23] Gupta, M. D., Rajaram, S., Petrovic, N., and Huang, T. S. Models for patch based image restoration. *Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop* (2006).

[24] Howe, Nicholas R., Rath, Toni M., and Manmatha, R. Boosted decision trees for word recognition in handwritten document retrievals. In *Proceedings of the SIGIR* (2005), pp. 377–383.

[25] Howe, Nicholas R., Rath, Toni M., and Manmatha, R. Boosted decision trees for word recognition in handwritten document retrievals. In *Proceedings of the SIGIR* (2005), pp. 377–383.

[26] Jain, A., and Namboodiri, A. Indexing and retrieval of on-line handwritten documents. In *Proceedings of the International Conference on Document Analysis and Recognition* (2003), pp. 655–659.

[27] J.Edwards, Y.W.Teh, D.A.Forsyth-R.Bock M.Maire G.Vesom. Making latin manuscripts searchable using hmms. In *Proceedings of Neural Information Processing Systems* (2004), pp. 385–392.

[28] Jing, Hongyan. Using hidden Markov modeling to decompose human-written summaries. *Computational Linguistics 28*, 4 (2002), 527–543.

[29] Jojic, N., Frey, B. J., and Kannan, A. Epitomic analysis of appearance and shape. *Proceedings of the Ninth IEEE International Conference on Computer Vision* (2003).

[30] Kamel, M., and Zhao, A. Extraction of binary characters/graphics images from grayscale document images. *CVGIP: Graphic Models Image Processing 55*, 3 (1993).

[31] Kane, S., Lehman, A., and Partridge, E. Indexing George Washington's handwritten manuscripts. In *CIIR Technical Report MM-34* (Center for Intelligent Information Retrieval, University of Massachusetts Amherst, 2001).

[32] Kato, N., Suzuki, M., Omachi, S., Aso, H., and Nemoto, Y. A handwritten character recognition system using directional element feature and asymmetric mahalanobis distance. *IEEE Trans. Pattern Anal. Mach. Intell. 21*, 3 (1999), 258–262.

[33] Kim, G., and Govindaraju, V. A lexicon driven approach to handwritten word recognition for real-time applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence 19* (April 1997), 366–379.

[34] Kundu, A., He, Yang, and Chen, Mou-Yen. Alternatives to variable duration HMM in handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence 20*, 11 (1998), 1275–1280.

[35] Kwok, T., Perrone, M., and Russell, G. Ink retrieval from handwritten documents. In *Proceedings of the Second International Conference on Data Mining, Financial Engineering, and Intelligent Agents* (2000), pp. 461–466.

[36] Langville, A. N., and Meyer, C. D. In *Google's PageRank and Beyond: The Science of Search Engine Rankings* (2006), Princeton University Press.

[37] Lee, Duk-Ryong, Kim, Woo-Youn, and Oh, Il-Seok. Hangul document image retrieval system using rank-based recognition. In *Proceedings of the International Conference on Document Analysis and Recognition* (2005), vol. 2, pp. 615–619.

[38] Manmatha, R., and Rath, T. M. Indexing of handwritten historical documents-recent progress. In *Symposium on Document Image Understanding Technolegy (SDIUT)* (2003), pp. 77–85.

[39] Marinai, S., Marino, M., and Soda, G. Indexing and retrieval of words in old documents. In *Proceedings of the International Conference on Document Analysis and Recognition* (2003), p. 223C227.

[40] Marti, U., and Bunke, H. On the influence of vocabulary size and language models in unconstrained handwritten text recognition. In *Proceedings. Sixth International Conference on Document Analysis and Recognition* (2001), vol. 2, pp. 260–265.

[41] Marti, U., and Bunke, H. The IAM-database: an English sentence database for off-line handwriting recognition. *Int. Journal on Document Analysis and Recognition 5* (2006), 39–46.

[42] Milewski, Robert, and Govindaraju, Venu. Extraction of handwritten text from carbon copy medical form images. In *Document Analysis Systems* (2006), pp. 106–116.

[43] Milewski, Robert, and Govindaraju, Venu. Extraction of handwritten text from carbon copy medical form images. *Document Analysis Systems 2006* (2006), 106–116.

[44] Mittendorf, Elke, Schauble, Peter, and Sheridan, Paraic. Applying probabilistic term weighting to OCR text in the case of a large alphabetic library catalogue. In *Research and Development in Information Retrieval* (1995), pp. 328–335.

[45] Niblack, W. An introduction to digital image processing. *Englewood Cliffs, N.J. Prentice Hall* (1986).

[46] Ohta, M., Takasu, A., and Adachi, J. Retrieval methods for English text with misrecognized OCR characters. In *Proceedings of the International Conference on Document Analysis and Recognition,* (1997).

[47] Otsu, N. A. A threshold selection method from gray-level histogram. *IEEE Transactions on System Man Cybernetics 9*, 1 (1979).

[48] Pearl, J. Probalistic reasoning in intelligent systems: networks of plausible inference. *Morgan Kaufmann Publishers Inc.* (1988).

[49] Rath, Toni M., Manmatha, R., and Lavrenko, Victor. A search engine for historical manuscript images. In *Proceedings of the 27th annual international ACM SIGIR conference on research and development in information retrieval* (2004).

[50] Reynolds, D. A., Quatieri, T. F., and Dunn, R. B. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing 10*, 1-3 (2000), 19–41.

[51] Rothfeder, J. L., Feng, S., and Rath, T. M. Using corner feature correspondences to rank word images by similarity. In *CIIR Technical Report MM-44* (Center for Intelligent Information Retrieval, University of Massachusetts Amherst, 2003).

[52] S, Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Trans on Acoustics, Speech, and Signal Processing 35*, 3 (1987), 400–401.

[53] Sauvola, J., Seppanen, T., Haapakoski, S., and Pietiktinen, M. Adaptive document binarization. *Proceedings of the 4th International Conference on Document Analysis and Recognition* (1997), 147–152.

[54] Seeger, M., and Dance, C. Binarising camera images for OCR. *Proceedings of the Sixth International Conference on Document Analysis and Recognition*, 54–58.

[55] Sun, N., Abe, M., and Nemoto, Y. A handwritten character recognition system by using improved directional element feature and subspace method. *Trans. IEICE J78-D-II*, 6 (1995), 922–930.

[56] Tsukumo, J. Improved algorithm for direction pattern matching and its application for handprinted kanji character classification. *IEICE Technical Report* (1990).

[57] Uchiashi, S., and Wilcox, L. Automatic index creation for handwritten notes. In *Proceedings of the International Conference on Acoustic, Speech and Signal Processing* (1999), pp. 3453–3456.

[58] Wolf, C., and Doermann, D. Binarization of low quality text using a Markov random field model. *Proceedings of International Conference on Pattern Recognition* (2002).

[59] Xue, Hanhong, and Govindaraju, V. Hidden Markov models combining discrete symbols and continuous attributes in handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence 28*, 3 (2006), 458–462.

[60] Yang, Y., and Yan, H. An adaptive logical method for binarization of degraded document images. *Pattern Recognition* (2000), 787–807.

[61] Yasuda, M., Ohkubo, J., and Tanaka, K. Digital image inpainting based on Markov random field. *Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce 2* (2005), 747–752.

[62] Yoo, Jin-Yong, Kim, Min-Ki, Han, Sang Yong, and Kwon, Young-Bin. Line removal and restoration of handwritten characters on the form documents. *Proceedings of the 4th International Conference on Document Analysis and Recognition* (1997), 128–131.

[63] Zhang, B., Srihari, S. N., and Huang, C. Word image retriecal using binary features. In *Document Recognition and Retrieval XI, SPIE* (Greenbelt, Maryland, April 2004), vol. 5296, pp. 45–53.

[64] Zhang, J., Ding, X., and Liu, C. Multi-scale feature extraction and nested-subset classifier design for high accuracy handwritten character recognition. In *Proceedings of the International Conference on Pattern Recognition* (2000).