PURDUE UNIVERSITY
GRADUATE SCHOOL
Thesis/Dissertation Acceptance

This is to certify that the thesis/dissertation prepared

By Yandong Guo

Entitled
STATISTICAL MODEL-BASED BINARY DOCUMENT IMAGE CODING, RECONSTRUCTION, AND ANALYSIS

For the degree of Doctor of Philosophy

Is approved by the final examining committee:

CHARLES A. BOUMAN
Chair

JAN P. ALLEBACH

MARK R. BELL

MARY L. COMER

To the best of my knowledge and as understood by the student in the *Research Integrity and Copyright Disclaimer (Graduate School Form 20)*, this thesis/dissertation adheres to the provisions of Purdue University's "Policy on Integrity in Research" and the use of copyrighted material.

Approved by Major Professor(s): CHARLES A. BOUMAN

Approved by: V. Balakrishnan 08-01-2014
Head of the Graduate Program Date

STATISTICAL MODEL-BASED BINARY DOCUMENT IMAGE CODING, RECONSTRUCTION, AND ANALYSIS

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Yandong Guo

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

August 2014

Purdue University

West Lafayette, Indiana

To my parents.

ACKNOWLEDGMENTS

First and foremost, I would like to express my deepest appreciation to my advisor, Prof. Charles A. Bouman and Prof. Jan P. Allebach, for their countless support and help during my research. Their enthusiasm about research and education, and uncompromising attitude towards perfection has been a great inspiration to me. Particularly, they helped me realize that importance of presentation, and gave me a lot of training and opportunities to present my work. They even supported my trip to Australia to give presentation and know peers in the area. All these become priceless asset not only for my research at Purdue, but also for my entire life. I can never thank them enough.

I would also like to thank Prof. Mark R. Bell and Prof. Mary L. Comer, who both serve as my advisory committee members. Prof. Mary L. Comer has been giving me warm encouragement. Prof. Mark R. Bell helped me deeply understanding random variables and information theory. Both of them gave me a lot of help in my research.

I would also like to thank Hewlett-Packard Company for their years of continuous financial support and a summer internship in 2011. Also, I appreciate GE healthcare for supporting me the summer intern in 2013. I'm very glad to meet Thibault, Jean-Baptiste there, from whom I learned a lot. It's very memorable. I would also like to thank all my colleagues and friends, including Zhou Yu, Guangzhi Cao, Eri Haneda, Ruoqiao Zhang, Pengchong Jin, Tak-Shing Wong, Haitao Xue, Singanalur Venkatakrishnan, Leonardo R. Bachega, S. Jordan Kisner, Zhenhao Ge, Yufang Sun, Sirui Hu, Xing Liu, Weibao Wang, Zonghao Li, Cheng Lu, Huixi Zhao, and Yuhui Zheng.

Finally, from the bottom of my heart, I would like to thank my wonderful parents, Jinyao Guo and Xiaoling Yang, for their unconditional support and love to me.

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

ABSTRACT

Guo, Yandong Ph.D., Purdue University, August 2014. Statistical model-based Binary Document Image Coding, Reconstruction, and Analysis. Major Professors: Charles A. Bouman and Jan P. Allebach.

Binary document image is still one of the most important information carriers in this era of data. In this final exam, we will present two novel technologies to learn and understand low-level features from document images, and we also apply these technologies in the applications including compression, reconstruction, registration, and searching.

The first learning technology is the entropy-based dictionary learning, which is a method to learn a strong prior for document images. The information in this prior is used to encode the image effectively. If there are more than one page to be encoded, we impose hierarchical structure onto the dictionary, and dynamically update the dictionary. Compared with the best existing methods, we achieve much higher compression ratio.

The dictionary prior we proposed is also used to restore noisy document images. Our dictionary-based restoration improves the document image quality, and the encoding effectiveness simultaneously.

The second learning technology is layout structure detection for document images. We set up a cost function to describe the local text line property and by optimizing this cost function, we detect the overall layout of the document image. Our method is faster and more efficient, compared with conventional methods. Using this technology, we construct sparse feature set for document images, which is then used in our novel, efficient document image searching system.

1. A RATE OPTIMIZED LOSSLESS BINARY IMAGE CODER

1.1 Problem statement

The JBIG2 standard is widely used for binary document image compression primarily because it achieves much higher compression ratios than conventional facsimile encoding standards, such as T.4, T.6, and T.82 (JBIG1). A typical JBIG2 encoder works by first separating the document into connected components, or symbols. Next it creates a dictionary by encoding a subset of symbols from the image, and finally it encodes all the remaining symbols using the dictionary entries as a reference. In this paper, we propose a novel dictionary learning method for binary image compression compliant with the JBIG2 standard.

Our dictionary learning method includes the following contributions. First, we propose a conditional entropy estimation (CEE) method to measure the dissimilarity between a symbol and its associated dictionary entry. Compared to conventional measures of dissimilarity, our CEE measure has much higher correlation with the number of bits required to encode a symbol with a particular dictionary entry. Second, we propose a framework for optimal dictionary construction and symbol encoding based on a unsupervised clustering framework. Third, we proposed a fast algorithm for implementation of our method using prescreening procedures. In experiments with a variety of documents, we demonstrate that the deployment of our dictionary learning method results in approximately a 14% reduction in the overall bitrate of the JBIG2 encoded bitstream as compared to the best conventional dictionary learning methods for JBIG2 binary image compression.

1.2 Introduction

Binary image compression with dictionary learning consists of two stages, first, a strong prior, called dictionary, is learned from the image and encoded as part of the bitstream. Second, the image is encoded using the information contained in this dictionary. Dictionary learning is critical to binary image compression since a dictionary better representing the image can be used to encode the image with higher compression ratio.

In this paper, we propose a dictionary learning method to encode the binary images effectively. Our compression is compliant with the JBIG2 standard, while extendable to other dictionary learning-based binary image compression schemes. The JBIG2 standard, developed by the Joint Bi-level Image Experts Group [1], achieves higher compression ratios than previous binary image compression methods, such as T.4, T.6, and T.82 (JBIG1) [2], through the use of dictionaries, briefly described as follows.

A typical JBIG2 encoding is achieved by a combination of the following operations: image segmentation, symbol extraction, dictionary construction, and entropy encoding [3, 4]. First, the original binary document image is separated into regions corresponding to text (i.e., repeated symbols) and non-text. In this work, we will assume the entire JBIG2 document is treated as text. Next, individual connected components are extracted that typically correspond to individual text symbols. Then, the extracted symbols are used to create a representative dictionary of typical symbols, and the dictionary is encoded using a traditional entropy encoding method. Finally, the symbols for each page of the document are encoded by indexing them to individual entries of the dictionary, and these indices are entropy encoded. With lossly JBIG2 encoding, each symbol is represented approximately with a corresponding dictionary entry; but with lossless JBIG2 encoding, the difference between the symbol and corresponding dictionary entry is also entropy encoded.

While all JBIG2 encoders perform these basic tasks, some encoders achieve better compression by constructing better dictionaries and using those dictionary entries more effectively. For example, a larger dictionary can reduce the number of bits required for lossless encoding of each symbol because each dictionary entry can more precisely match its corresponding symbol. However, a larger dictionary will also increase the number of bits required to encode the dictionary itself, so the best coder must balance these two objectives.

Even if the dictionary is fixed, an encoder can improve compression by more optimally selecting the dictionary entry for each new symbol. Ideally, each symbol should be encoded using the dictionary entry that produces the lowest number of bits in the encoded output. However, this approach, which is known as the operational rate-distortion approach [5,6], is not practical since it requires too much computation. Therefore, any practical JBIG2 coders must use a more computationally efficient method to match each symbol to a dictionary entry.

Overall, the dictionary learning for binary images needs to solve the following two problems, dictionary indexing and dictionary construction. Dictionary indexing, also called sparse coding, is to select the best dictionary entry for a given symbol from a fixed dictionary. Dictionary construction is to build up dictionary entries so as to encode the image effectively with limited dictionary size.

For binary image compression, conventional dictionary learning methods select the dictionary entry by minimizing a measure of dissimilarity between the symbol and the dictionary entry. Dissimilarity measures widely used in JBIG2 include the Hamming distance, known as XOR [7], and weighted Hamming distance, known as WXOR [8,9]. The weighted Hamming distance is calculated as the weighted summation of the difference between a symbol bitmap and a dictionary entry bitmap. Zhang, Danskin, and Yong have also proposed a dissimilarly measure based on cross-entropy which is implemented as WXOR with specific weights [10, 11]. In addition, various methods have been proposed to solve the dictionary construction problem. These methods typically cluster the symbols into groups, according to a dissimilarity measure, using

K-means clustering or a minimum spanning tree [12–14]. Within each group, one dictionary entry is used to represent all the symbols of that group.

For gray scale images, more dictionary learning methods have been proposed, which provide some insight to the dictionary learning for binary image compression. Particularly, there have been information-theoretic dictionary learning methods, which measure the dissimilarity between two image patches by estimating the log of the conditional probability of one patch given the other one. In [15, 16], this log of conditional probability reduced to Euclidean distance between image patches under the assumption of Gaussian i.i.d. noise. While in [17–19], the conditional probability model of a image patch given another one is inferenced from training. These training-based methods have shown promising results for applications with gray scale images. The dictionary construction methods for gray scale images include the vector quantization (VQ) method [15], K-SVD method [16], and the recursive least squares dictionary learning algorithm (RLS-DLA) [20].

In this paper, we propose a novel dictionary learning method for binary image compression, consisting of the following technologies. First, we propose a conditional entropy estimation (CEE) method to measure the dissimilarity between a symbol and its associated dictionary entry. Based on conditional probability estimation, our CEE method provides a fast way to approximate the number of bits required to encode a symbol with its associated dictionary entry. Therefore, for a given symbol, we can select the associated dictionary entry by minimizing the approximated number of bits required to encode this symbol. A fundamental difference between our proposed CEE method and the cross-entropy method in [10, 11] is that our CEE method learns the statistics of the image being encoded to estimate the parameters in the conditional probability model, which leads to a more accurate estimate of the bitrate. As compared with the training-based conditional information estimation used for gray scale images in [17–19], our CEE method uses a very different conditional probability model since the image we are processing is binary. Moreover, the model inference in our CEE method is also different: we utilize a factorization strategy to decompose

the complicated conditional multivariate distribution into simple factors, which dramatically drops the training complexity as well as reduces the overall computational cost. We note that a preliminary version of the CEE method was presented in our conference paper of [21].

Second, we formulate the file size of the image bitstream as a function of the dictionary and the mapping between symbols and dictionary entries. Then we obtain the dictionary and the mapping by minimizing this cost function. To solve this optimization problem, we design a special greedy algorithm in the agglomerative clustering framework (GC). The following reasons forbid us to directly apply a standard clustering method, such as K-means in [12] or vector quantization in [22]. First, we have a constrained feasible set, since the dictionary we construct has to be binary. Second, the dissimilarity between symbols and dictionary entries are measured by CEE, rather than Euclidean distance (With binary signal, Euclidean distance reduces to XOR). Third, we need to make our algorithm faster than a standard clustering algorithm, since binary image compression is typically implemented in a computational capacity limited environment,

Last but not least, we propose a prescreening procedure to reduce the number of the times of CEE calculation. By skipping the CEE calculation in a smart way, our prescreening procedure reduces the overall computation time by sacrificing a little bit compression ratio. This strategy is especially useful for the embedded systems with limited computational capacity, for example, low end scanners.

With the three components described above, our dictionary learning method produces a better dictionary to represent the image, and uses the dictionary more effectively, compared with the previous dictionary learning methods. In our experiments with a variety of documents, our dictionary learning method substantially improves the lossless JBIG2 compression ratio. Finally, as a learning-based prior model construction method, our dictionary learning method also has a wide range of potential applications, such as image reconstruction [23, 24], compressed sensing [25], and dictionary-based image compression compliant with other kinds of schemes [15, 20].

The organization of the rest of the paper is as follows. In Sec. 1.3, we formulate the dictionary indexing and construction as a mathematical optimization problem. In Sec. 1.4, we describe our CEE method in detail. In Sec. 1.5, we present our optimization algorithm used to construct dictionary and the mapping. Section 1.6 presents the experimental results.

1.3 Problem formulation

The JBIG2 encoder extracts a sequence of symbols from the text region and then encodes them using a dictionary. More specifically, let $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^{N}$ denote the N symbols that are extracted from the document. Each symbol $\mathbf{y}_i$ contains the bitmap of the i^{th} symbol on the page, and each symbol is encoded using a corresponding dictionary entry $\mathbf{d}_j$ selected from a complete dictionary $\mathbf{D} = \{\mathbf{d}_j\}_{j=1}^{M}$, where M is the number of entries in the dictionary. Furthermore, let $j = f(i)$ denote the function that maps each individual symbol, $\mathbf{y}_i$, to its corresponding dictionary entry, $\mathbf{d}_j$.

Let $R(\mathbf{Y}; \mathbf{D}, f)$ denote the number of bits required to encode the symbols, $\mathbf{Y}$, using the dictionary, $\mathbf{D}$, and the mapping, $f(i)$. Since we are only considering lossless encoding, our objective is to select the $f^*(i)$ and construct $\mathbf{D}^*$ so that the bit rate is minimized.

$$\{\mathbf{D}^*, f^*\} = \underset{\mathbf{D},f}{\operatorname{argmin}} R(\mathbf{Y}; \mathbf{D}, f)\,. \tag{1.1}$$

According to the JBIG2 standard, we formulate the number of bits $R(\mathbf{Y}; \mathbf{D}, f)$ as,

$$\begin{aligned} & R(\mathbf{Y}; \mathbf{D}, f) \\ =& \sum_{i=1}^{N} \left[R_{y,i}\left(\mathbf{y}_i | \mathbf{d}_{f(i)}\right) + \log_2(M) + C_y \right] \\ +& \sum_{j=1}^{M} \left[R_{d,j}\left(\mathbf{d}_j\right) + C_d \right] \ , \end{aligned} \tag{1.2}$$

where the first summation represents the bits used to encode the symbols, and the second summation represents the bits used to encode the dictionary. In the first sum,

the term $R_{y,i}\left(\mathbf{y}_i|\mathbf{d}_{f(i)}\right)$ denotes the number of bits required to encode the bitmap of the symbol $\mathbf{y}_i$ using the dictionary entry $\mathbf{d}_{f(i)}$. The term C_y is a constant that denotes the overhead (in bits) required for encoding the symbol's width, height, and position; and the term $\log_2(M)$ accounts for the bits required to encode the index of the dictionary entry.

In the second sum, the term $R_{d,j}\left(\mathbf{d}_j\right)$ denotes the number of bits required to encode the bitmap of the dictionary entry $\mathbf{d}_j$. The term C_d is a constant that denotes the overhead (in bits) required for encoding the dictionary entry's width and height.

Unfortunately, it is too computationally expensive to minimize $R(\mathbf{Y};\mathbf{D},f)$ over $\mathbf{D}$ and f. This is mainly because the number of bits has a complicated dependency on the bitmap of symbols and dictionary entries. In a JBIG2 standard compliant bitstream, the number of bits $R_{y,i}\left(\mathbf{y}_i|\mathbf{d}_{f(i)}\right)$ is determined by not only the symbol $\mathbf{y}_i$ and the dictionary entry $\mathbf{d}_j$, but also the previously encoded symbols and their associated dictionary entries. Moreover, the number of bits $R_{d,j}\left(\mathbf{d}_j\right)$ is a determined by not only the dictionary entry $\mathbf{d}_j$, but also all the previously encoded dictionary entries. Therefore, we propose the following two approximation methods to simplify the dependency.

First, we propose to approximate $R_{y,i}\left(\mathbf{y}_i|\mathbf{d}_{f(i)}\right)$ based on an estimate of the conditional probability of the symbol $\mathbf{y}_i$ given its associated dictionary entry $\mathbf{d}_{f(i)}$,

$$R_{y,i}\left(\mathbf{y}_i|\mathbf{d}_{f(i)}\right) \cong -\log P(\mathbf{y}_i|\mathbf{d}_{f(i)};\boldsymbol{\phi})\,, \tag{1.3}$$

where $\boldsymbol{\phi}$ parameterizes the distribution. With this parameterization, the conditional probability $P(\mathbf{y}_i|\mathbf{d}_{f(i)};\boldsymbol{\phi})$ is only determined by $\mathbf{y}_i$ and $\mathbf{d}_{f(i)}$, so our approximation can be efficiently calculated. Details about $\boldsymbol{\phi}$ are provided in Sec. 1.4.

Second, we approximate the number of bits encode the dictionary entries as (1.4),

$$R_{d,j}\left(\mathbf{d}_j\right) \cong \frac{w_j^d h_j^d}{4}\,, \tag{1.4}$$

where w_j^d and h_j^d are the width and height of the j^{th} dictionary entry, respectively. We propose a coarser but more efficient approximation in (1.4) compared with the

approximation in (1.3), since the number of bits required to encode dictionary entries is much smaller than the number of bits required to encode symbols with reference.

With the approximations proposed above, we formulate our objective function as,

$$\begin{aligned}&\hat{R}(\mathbf{Y};\mathbf{D},f,\boldsymbol{\phi})\\ =&\sum_{i=1}^{N}\left[-\log P(\mathbf{y}_i|\mathbf{d}_{f(i)};\boldsymbol{\phi})+\log_2(M)+C_y\right]-\log p_\phi(\boldsymbol{\phi})\\ +&\sum_{j=1}^{M}\left[\frac{w_j^d h_j^d}{4}+C_d\right].\end{aligned}\tag{1.5}$$

where the term $p_\phi(\boldsymbol{\phi})$ is a prior distribution on the parameter $\boldsymbol{\phi}$ that is used to stabilize the estimation of the parameter $\boldsymbol{\phi}$ as described in Sec 1.4. Using these assumptions, we can obtain the best dictionary, mapping, and conditional probability distribution parameter by,

$$\{\hat{\mathbf{D}},\hat{f},\hat{\boldsymbol{\phi}}\}=\underset{\mathbf{D},f,\boldsymbol{\phi}}{\operatorname{argmin}}\,\hat{R}(\mathbf{Y};\mathbf{D},f,\boldsymbol{\phi})\,.\tag{1.6}$$

We design an alternating optimization scheme to solve (1.6), summarized as the pseudocode in Fig. 1.1. First, we estimate the conditional probability parameter $\boldsymbol{\phi}$ with fixed dictionary $\mathbf{D}^{(0)}$ and mapping $f^{(0)}$. The dictionary $\mathbf{D}^{(0)}$ and the mapping $f^{(0)}$ is obtained by applying a fast dictionary construction and indexing method to all the extracted symbols $\{\mathbf{y}_i\}_{i=1}^N$. Here we choose XOR-based one pass (XOR-OP) method in [7] due to its low computational cost. The model of the conditional probability $P(\mathbf{y}_i|\mathbf{d}_{f(i)};\boldsymbol{\phi})$ and the term $p_\phi(\boldsymbol{\phi})$, as well as the analytical solution of (1.7) are presented in details in Sec. 1.4.

Second, we rebuild the dictionary and estimate the mapping by minimizing (1.8) with fixed $\boldsymbol{\phi}$. We propose a greedy algorithm in the agglomerative clustering framework (GC) to optimize (1.8), with details provided in Sec. 1.5. Finally, we apply the standard JBIG2 encoder to encode the dictionary, the mapping, and the symbols. Note that each of the symbols are encoded with their associated dictionary entries. The bitstream we produce is compliant with the standard.

ROE_Encode $(\mathbf{Y})\,\{$

/* Estimate parameter $\boldsymbol{\phi}$ */

$$\{\mathbf{D}^{(0)}, f^{(0)}\} \leftarrow \text{XOR-OP } (\mathbf{Y}, T_{\text{XOR}})$$

$$\hat{\boldsymbol{\phi}} \leftarrow \underset{\boldsymbol{\phi}}{\operatorname{argmin}}\, \hat{R}(\mathbf{Y}; \mathbf{D}^{(0)}, f^{(0)}, \boldsymbol{\phi}) \tag{1.7}$$

/* Estimate dictionary $\mathbf{D}$ and mapping f */

$$\{\hat{\mathbf{D}}, \hat{f}\} \leftarrow \underset{\mathbf{D},f}{\operatorname{argmin}}\, \hat{R}(\mathbf{Y}; \mathbf{D}, f, \hat{\boldsymbol{\phi}}) \tag{1.8}$$

$\text{JBIG2\_ENCODE}\Big(\mathbf{Y}, \hat{\mathbf{D}}, \hat{f}\Big)$

$\}$

Fig. 1.1. Pseudocode of the rate optimization encoder (ROE) for a single page binary image. The XOR-based one pass (XOR-OP) method [4] is a fast method to construct dictionary and indexing. The parameter T_{XOR} is a parameter required by XOR-OP, which is set to the value suggested in [4]. The parameter $\boldsymbol{\phi}$ is introduced to approximate the number of bits efficiently. By alternating optimization shown in (1.7) and (1.8), we obtain the best parameter $\hat{\boldsymbol{\phi}}$, dictionary $\hat{\mathbf{D}}$, and mapping $\hat{f}$, which minimizes the approximated number of bits. Note that afterwards, we use standard JBIG2 encoder to encode the dictionary, the mapping, and the symbols. Our encoder produces a standard compliant bitstream.

1.4 Bitrate approximation

To approximate the bitrate efficiently requires us to estimate the parameter $\boldsymbol{\phi}$. With the dictionary $\mathbf{D}$ and the mapping f fixed, the cost function (1.7) is converted as,

$$\begin{aligned}\hat{\boldsymbol{\phi}} &= \underset{\boldsymbol{\phi}}{\operatorname{argmin}}\, \hat{R}(\mathbf{D}, f, \boldsymbol{\phi}) \\ &= \underset{\boldsymbol{\phi}}{\operatorname{argmax}} \sum_{i=1}^{N} \log P(\mathbf{y}_i | \mathbf{d}_{f(i)}; \boldsymbol{\phi}) + \log p_{\phi}(\boldsymbol{\phi})\,. \end{aligned} \tag{1.9}$$

As shown above, the first term in (1.9) is the log likelihood function. We design the term $p_\phi(\boldsymbol{\phi})$ as the prior distribution function of $\boldsymbol{\phi}$. In this way, the cost function (1.9) results in the maximum a posteriori (MAP) estimate $\boldsymbol{\phi}$.

In the following paragraphs, we propose a model for the sample distribution $P(\mathbf{y}_i|\mathbf{d}_{f(i)};\boldsymbol{\phi})$, and a model for the prior distribution $p_\phi(\boldsymbol{\phi})$. Afterwards, we introduce a method for computing the solution to the minimization problem of (1.9).

The conditional probability $P(\mathbf{y}_i|\mathbf{d}_{f(i)};\boldsymbol{\phi})$ can have a very complicated form, since both $\mathbf{y}_i$ and $\mathbf{d}_j$ are high dimensional random variables. This makes the parameter vector $\boldsymbol{\phi}$ contain too many elements to be estimated. To solve this problem, we model $P(\mathbf{y}_i|\mathbf{d}_{f(i)};\boldsymbol{\phi})$ as the product of a sequence of simple probability density functions,

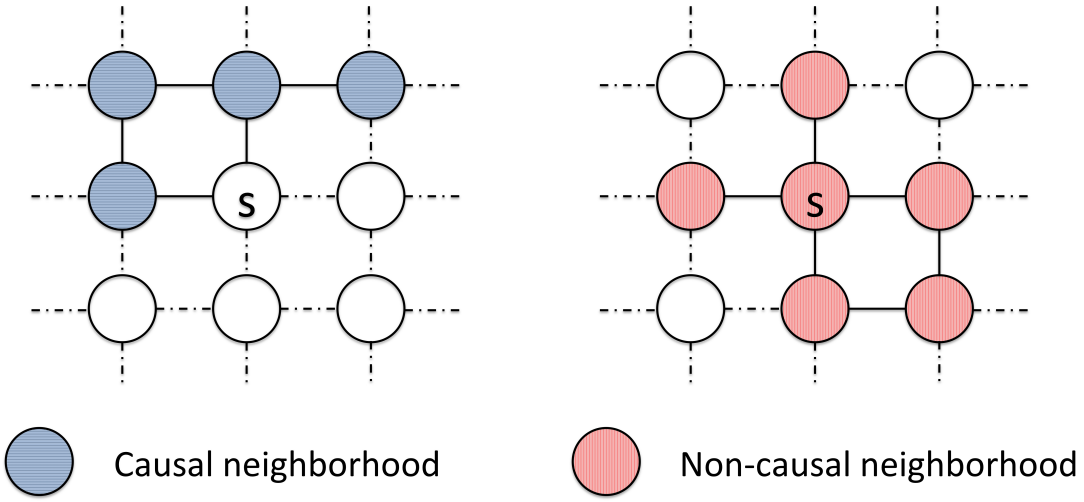
$$P(\mathbf{y}_i|\mathbf{d}_{f(i)};\boldsymbol{\phi}) = \prod_s p\left(y_i(s)|\mathbf{c}(\mathbf{y}_i,\mathbf{d}_j,s);\boldsymbol{\phi}\right) , \tag{1.10}$$

where the term $p\left(y_i(s)|\mathbf{c}(\mathbf{y}_i,\mathbf{d}_j,s);\boldsymbol{\phi}\right)$ is the conditional probability for the symbol pixel $y_i(s)$ conditioned on its reference context $\mathbf{c}(\mathbf{y}_i,\mathbf{d}_j,s)$, of which the definition is shown in Fig. 2.1.

Fig. 2.1. graphically illustrates the structure of the reference context. As shown, the reference context $\mathbf{c}(\mathbf{y}_i,\mathbf{d}_j,s)$ is a 10-dimensional binary vector, consisting of 4 causal neighborhood pixels of $y_i(s)$ in $\mathbf{y}_i$, and 6 non-causal neighborhood pixels of $d_j(s)$ in $\mathbf{d}_j$.

The decomposition in (1.10) is based on the assumption that, the symbol pixel $y_i(s)$, given its reference context $\mathbf{c}(\mathbf{y}_i,\mathbf{d}_j,s)$, is conditionally independent of its previous (in raster order) symbol pixels except its 4 casual neighbors. This conditional independency design makes our decompostion different from the existing decomposition/factorization methods in inference complicated distributions [26–28].

Since the symbol pixels are binary, we model their conditional distribution given a particular reference context as a Bernoulli distribution. We assume that, for a given document image, the natural parameter of $p\left(y_i(s)|\mathbf{c}(\mathbf{y}_i,\mathbf{d}_j,s);\boldsymbol{\phi}\right)$ is completely



(a) Neighbors in symbol

(b) Neighbors in dictionary entry

Fig. 1.2. The 4 causal neighborhood pixels of $y_i(s)$ in $\mathbf{y}_i$, and the 6 non-causal neighborhood pixels of $d_j(s)$ in $\mathbf{d}_j$. Note that this is not the only neighborhood system we can use. We choose to use the neighborhood system which is also used in the JBIG2 standard [1], but estimate the conditional probability in a different way, described in section 1.4.

determined by the reference context $\mathbf{c}(\mathbf{y}_i, \mathbf{d}_j, s)$. We use ϕ_c to denote this natural parameter, where $c = \mathbf{c}(\mathbf{y}_i, \mathbf{d}_j, s)$ is the value of the reference context vector.

$$p(y_i(s)|\mathbf{c}(\mathbf{y}_i, \mathbf{d}_j, s); \boldsymbol{\phi}) = \phi_c^{1-y_i(s)}(1-\phi_c)^{y_i(s)} \tag{1.11}$$

The reference context $\mathbf{c}(\mathbf{y}_i, \mathbf{d}_j, s)$ could possibly have 2^{10} different values with our 10 bit neighborhood system, so there are 2^{10} parameters to be estimated.

$$\boldsymbol{\phi} = [\phi_1, \phi_2, \ldots, \phi_{1024}]^T \tag{1.12}$$

Substituting (1.11) into (1.10), we obtain the likelihood function for $\boldsymbol{\phi}$.

In order to construct the prior distribution for $p_\phi(\boldsymbol{\phi})$, we assume that all the elements in $\boldsymbol{\phi}$ are independent and identically distributed, following Beta distribution,

$$p_\phi(\boldsymbol{\phi}) = \prod_c \text{Beta}(\phi_c|a,b)\,, \tag{1.13}$$

$$\text{Beta}(\phi_c|a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\phi_c^{a-1}(1-\phi_c)^{b-1}\,. \tag{1.14}$$

We set $a = b = 2$.

The prior distribution we designed introduces the following benefits. First, our prior makes the estimation of $\boldsymbol{\phi}$ more stable when the number of observations is limited. Second, this choice of prior results in a simple form for the solution of equation (1.9), as we will see in the following paragraphs.

With the likelihood (1.10) and (1.11), and the prior (1.13) and (1.14) presented in the last subsection, we update the MAP cost function in (1.9) as follows,

$$\begin{aligned}\hat{\boldsymbol{\phi}} &= \underset{\boldsymbol{\phi}}{\text{argmin}} \sum_{i=1}^{N} -\log P(\mathbf{y}_i|\mathbf{d}_{f(i)};\boldsymbol{\phi}) - \log p_\phi(\boldsymbol{\phi}) \\ &= \underset{\boldsymbol{\phi}}{\text{argmax}} \left\{ \sum_{i=1}^{N}\sum_s [1-y_i(s)] \log \phi_{c(\mathbf{y}_i,\mathbf{d}_{f(i)},s)} \right. \\ &\qquad + \sum_{i=1}^{N}\sum_s y_i(s) \log\left(1-\phi_{c(\mathbf{y}_i,\mathbf{d}_{f(i)},s)}\right) \\ &\qquad \left. + \sum_c \log\phi_c(1-\phi_c) \right\}\,. \end{aligned} \tag{1.15}$$

The sample data for this MAP estimation are the symbols $\mathbf{Y}$ extracted from the same page, the dictionary $\mathbf{D}^{(0)}$ and the mapping $f^{(0)}$ obtained in the initialization step. With these sample date fixed, the parameter vector $\boldsymbol{\phi}$ is estimated by maximizing (1.15). The estimation of ϕ_c is,

$$\hat{\phi}_c = \frac{N_O(c)+1}{N_A(c)+2}\,, \tag{1.16}$$

where $N_A(c)$ is the number of symbol bitmap pixels with the reference context $\mathbf{c}$, and $N_O(c)$ is the number of $0-$valued symbol bitmap pixels with the reference context $\mathbf{c}$.

1.5 Dictionary construction and indexing

In this section, we present our method to construct and index the dictionary entries with the conditional probability parameter $\boldsymbol{\phi}$ fixed. Given $\hat{\boldsymbol{\phi}}$, the best dictionary $\hat{\mathbf{D}}$ and the best mapping $\hat{f}$ is obtained by

$$\begin{aligned}\{\hat{\mathbf{D}}, \hat{f}\} &= \underset{\mathbf{D},f}{\operatorname{argmin}}\, \hat{R}(\mathbf{D}, f, \hat{\boldsymbol{\phi}}) \\ &= \underset{\mathbf{D},f}{\operatorname{argmin}} \left\{ \sum_{i=1}^{N} -\log P(\mathbf{y}_i|\mathbf{d}_{f(i)}; \hat{\boldsymbol{\phi}}) \right. \\ &\left. + \sum_{j=1}^{M} \frac{w_j^d h_j^d}{4} + MC_d + N\log_2(M) \right\}. \end{aligned} \tag{1.17}$$

1.5.1 Formulate the cost function as a clustering problem

To solve (1.17), We formulate this cost function as a clustering problem using the following design. We consider the set of symbols associated with the same dictionary entry as a cluster, called entropy-cluster. Moreover, we consider the dictionary entries as corresponding cluster representatives. Formally, we define the j^{th} entropy-cluster $\mathbb{Y}_j$ as,

$$\mathbb{Y}_j = \{\mathbf{y}_i | f(i) = j\}. \tag{1.18}$$

Each entropy-cluster $\mathbb{Y}_j$ is associated with one dictionary entry $\mathbf{d}_j$ as the cluster representative (codeword).

With the definition of the entropy-cluster, we re-organize the cost function in (1.17) into the form of a clustering problem in the entropy space, shown as follow,

$$\begin{aligned}\{\hat{\mathbf{D}}, \hat{f}\} = \underset{\mathbf{D},f}{\operatorname{argmin}} &\left\{ \sum_{j=1}^{M} \left[\sum_{\{i|f(i)=j\}} -\log P(\mathbf{y}_i|\mathbf{d}_j; \boldsymbol{\phi}) \right] \right. \\ &\left. + \sum_{j=1}^{M} \left(\frac{w_j^d h_j^d}{4} + C_d \right) + N\log_2(M) \right\}. \end{aligned} \tag{1.19}$$

The cost in (1.19) is composed of two parts. The first part, which we will refer to as the intra-cluster distortion, is the summation of the distortion within each of the

entropy-clusters, shown in the first line in (1.19). The second part, which we refer to as the clustering penalty, is a penalty of the number of clusters or equivalently dictionary entries used to represent the symbols, shown in the second line in (1.19). The intra-cluster distortion generally decreases when the number of dictionary entries increases, because the more dictionary entries we have, potentially the better associated dictionary entries we can find for symbols. On the other hand, the increase of the number of entropy-clusters (dictionary entries) generally increases the value of the clustering penalty. Therefore, the best dictionary size needs to balance between the intra-cluster distortion and the clustering penalty.

1.5.2 Optimization over D and f

With the clustering formulation, we propose an efficient method to solve (1.19), which is a greedy method in the unsupervised agglomerative clustering framework (GC). First, as the initial step, we construct much more dictionary entries (entropy-clusters) then needed. Then, we delete some of the dictionary entries by iteratively selecting and merging entropy-clusters. The merging procedure is specified as pseudo code in Fig. 1.3. The entropy-clusters to be merged are selected by applying a greedy strategy. Details of our algorithms are described as follows. The overall structure of our GC method is shown in Fig. 1.5.

Initialization

As the initial condition, we build up a large dictionary using the XOR-OP method in [7]. After the initial stage, we build up the best dictionary by selecting and removing the entries from this large dictionary, rather than constructing new entries, mainly due to the computational efficiency concern. Therefore, in order to provide sufficient dictionary entry candidates, we build up a much larger dictionary than what needed by an optimal encoder. More specifically, during the XOR-OP dictionary con-

```
Cluster_Merging(D, f, j, j') {      D ← D − {d_j}
                                    for i = 1 to N do
                                      if f(i) = j then
                                        f(i) ← j'
                                      end if
                                    end for
                                    return (D, f)
}
```

Fig. 1.3. Pseudocode to merge the entropy-cluster $\mathbb{Y}_j$ to $\mathbb{Y}'_j$. The cluster merging consists of two operations. One is to remove the dictionary entry $\mathbf{d}_j$ from the dictionary set $\mathbf{D}$. The other one is to associate the symbols in $\mathbb{Y}_j$ with the dictionary entry $\mathbf{d}_{j'}$. The input of this pseudocode include the current dictionary $\mathbf{D}$, mapping f, the index of the dictionary entry to be removed j, and the index of the dictionary entry to be used j'. The output is the updated dictionary and the updated mapping.

struction, we require each symbol $\mathbf{y}_i$ and its associated dictionary entry $\mathbf{d}_j$ to be very similar, formulated as,

$$d_{\mathrm{XOR}}(\mathbf{y}_i, \mathbf{d}_j) \leq T_{\mathrm{S}}\,, \tag{1.20}$$

where $d_{\mathrm{XOR}}(\mathbf{y}_i, \mathbf{d}_j)$ is the XOR dissimilarity calculated as the Euclidean distance over symbol size for binary signals [7], while T_{S} is a very small threshold to introduce large dictionaries.

We further discuss the way to tune the threshold T_{S} in details in the next subsection. This tuning plays trade off between the compression ratio and the computational cost.

Dictionary and mapping update

At the initial stage, we obtain a very large dictionary, so the intra-cluster distortion tends to be very small, while the clustering penalty is relatively large. In this step, we

Greedy_Cluster_Selection($\mathbf{Y}, \mathbf{D}, f, T_{\mathrm{L}}$) {

$$\left\{\hat{j}, \hat{j}'\right\} = \underset{\{j \neq j'\}}{\operatorname{argmin}} \left\{ \hat{R}(\mathbf{Y}; \mathbf{D}^{\setminus j}, f^{j' \leftarrow j}, \boldsymbol{\phi}) - \hat{R}(\mathbf{Y}; \mathbf{D}, f, \boldsymbol{\phi}) \right\} \tag{1.21}$$

Subject to: $w_j^d = w_{j'}^d$ (1.22)

$h_j^d = h_{j'}^d$ (1.23)

$\forall \mathbf{y}_i \in \mathbb{Y}_j,\ d_{\mathrm{XOR}}(\mathbf{y}_i, \mathbf{d}_{j'}) < T_{\mathrm{L}}$ (1.24)

$\forall \mathbf{y}_{i'} \in \mathbb{Y}_{j'},\ d_{\mathrm{XOR}}(\mathbf{y}_{i'}, \mathbf{d}_j) < T_{\mathrm{L}}$ (1.25)

return $\left\{\hat{j}, \hat{j}'\right\}$

}

Fig. 1.4. Pseudocode to select the entropy-cluster pair to be merged. As shown in (1.21), this selection is aiming to generate the greatest bitrate reduction by merging entropy-clusters, while this selection is subject to the constraints listed in (1.22), (1.23), (1.24), and (1.25). The terms $\mathbf{D}^{\setminus j}$ and $f^{j' \leftarrow j}$ are defined in (1.26). The constraints (1.24) and (1.25) require the entropy-cluster pair in the feasible set of (1.21) to have representatives (dictionary entries) of the same size. The term w_j^d and h_j^d are the width and height of the j^{th} dictionary entry, respectively. The constraints (1.24) and (1.25) mean that only entropy-clusters not very different from each other are considered to be selected to merge. The parameter T_{L}, typically larger than T_{S} in (1.20), is the threshold to control the number of entropy-cluster pairs in the feasible set of (1.21). The input of this pseudocode include the current dictionary $\mathbf{D}$, mapping f, symbols $\mathbf{Y}$, and the parameter T_{L}. The output are the indexes of the entropy-clusters to be merged $\hat{j}$ and $\hat{j}'$.

propose a dictionary and mapping update method to remove some of the dictionary entries and update the mapping to achieve a good balance between the clustering penalty and the intra-cluster distortion.

The dictionary and mapping is updated by iteratively selecting and merging entropy-clusters. We define the cluster merging first. Suppose we are given dictionary $\mathbf{D}$ and mapping f, merging the entropy cluster $\mathbb{Y}_j$ to $\mathbb{Y}_{j'}$ consists of two operations, shown as the pseudocode in figure 1.3. One is to remove the dictionary $\mathbf{d}_j$ from $\mathbf{D}$. The other one is to associate the symbols in $\mathbb{Y}_j$ with the dictionary entry $\mathbf{d}_{j'}$. We define $\mathbf{D}^{\backslash j}$ and $f^{j'\leftarrow j}$ as,

$$\left\{\mathbf{D}^{\backslash j}, f^{j'\leftarrow j}\right\} \equiv \text{Cluster\_Merging}\left(\mathbf{D}, f, j, j'\right) . \tag{1.26}$$

We propose a greedy method to select the entropy-clusters to be merged. The intuition of our method is to select clusters of which the merging would generate the greatest bitrate reduction. Moreover, in order to reduce the computational cost, we apply constraints to limit the feasible set of the selection of the entropy-clusters. The algorithm is listed in Fig. 1.4. Further detailed discussion on parameter tuning is provided in the next subsection.

The value of (1.21) is calculated by substituting (1.5) into (1.21),

$$\begin{aligned}\{\hat{j}, \hat{j}'\} = \underset{\{j\neq j'\}}{\text{argmin}} \Bigg\{ & -\frac{w_j^d h_j^d}{4} \\ & + \sum_{\mathbf{y}_i\in\{\mathbb{Y}_j\}} \left[\log P(\mathbf{y}_i|\mathbf{d}_j;\boldsymbol{\phi}) - \log P(\mathbf{y}_i|\mathbf{d}_{j'};\boldsymbol{\phi})\right] \Bigg\} .\end{aligned} \tag{1.27}$$

Termination

We iteratively apply the greedy entropy-cluster selection in Fig. 1.4 and entropy-cluster merging in Fig. 1.3, until the bit reduction due to cluster merging (minimum value of the cost in (1.21)) is not smaller than 0.

$$\hat{R}(\mathbf{Y};\mathbf{D}^{\backslash\hat{j}}, f^{\hat{j}'\leftarrow\hat{j}}, \boldsymbol{\phi}) - \hat{R}(\mathbf{Y};\mathbf{D}, f, \boldsymbol{\phi}) \geq 0 \tag{1.28}$$

This termination condition ensures the overall approximated number of bits (1.5) keeps monotonically decreasing during our clustering, until a minimum point (maybe local though) is arrived. Though our solution might be local optimum, in Sec. 1.6,

we conducted experiments to demonstrate that the dictionary and the mapping we obtained represent the test images effectively. Note that to search the global minimum of the cost function (1.17) is NP-hard.

The minimum value in (1.28) is calculated as,

$$\begin{aligned}
&\hat{R}(\mathbf{Y}; \mathbf{D}^{\setminus \hat{j}}, f^{\hat{j}' \leftarrow \hat{j}}, \boldsymbol{\phi}) - \hat{R}(\mathbf{Y}; \mathbf{D}, f, \boldsymbol{\phi}) \\
&= \sum_{\mathbf{y}_i \in \{\mathbb{Y}_{\hat{j}}\}} \left[\log P(\mathbf{y}_i | \mathbf{d}_{\hat{j}}; \boldsymbol{\phi}) - \log P(\mathbf{y}_i | \mathbf{d}_{\hat{j}'}; \boldsymbol{\phi}) \right] - \frac{w_{\hat{j}}^d h_{\hat{j}}^d}{4} \\
&\quad - N \log_2(M) + N \log_2(M-1) - C_d \,. \qquad (1.29)
\end{aligned}$$

The empirical value we used for C_d is 2.5.

After the clustering procedure, with fixed the dictionary $\hat{\mathbf{D}}$ and parameter $\hat{\boldsymbol{\phi}}$, we go through all the symbols again, and update the best dictionary entry index for each symbol,

$$\hat{f}(i) = \underset{j}{\operatorname{argmin}} - \log P(\mathbf{y}_i | \hat{\mathbf{d}}_{f(i)}; \hat{\boldsymbol{\phi}}) \,. \qquad (1.30)$$

1.5.3 Computational cost

Two threshold parameters T_{S} and T_{L} are introduced during our GC dictionary construction method to trade off between computational cost and compression ratio. If T_{L} is fixed, typically, smaller T_{S} in (1.20) introduces more dictionary entry candidates at the initial stage, which needs more merging operations but generally results in higher compression ratio. On the other hand, larger T_{S} introduces less dictionary entry candidates at the initial stage, which needs less merging operations but generally results in lower compression ratio.

If T_{S} is fixed, larger T_{L} in the constraints (1.24) and (1.25) gives more entropy cluster pair candidates to be selected, which typically needs more computing time but generally results in higher compression ratio. On the other hand, smaller T_{L} needs less computing time but generally results in lower compression ratio.

GC_Dictionary_Construction_Indexing$\left(\mathbf{Y}, \hat{\boldsymbol{\phi}}, T_{\mathrm{S}}, T_{\mathrm{L}}\right)$ {

/ * Initialization * /

$\{\mathbf{D}, f\} \leftarrow$ XOR-OP $(\mathbf{Y}, T_{\mathrm{XOR}} \leftarrow T_{\mathrm{S}})$

repeat

/ * Select two entropy-clusters * /

$$\left\{\hat{j}, \hat{j}'\right\} \leftarrow \text{Greedy\_Cluster\_Selection}\,(\mathbf{Y}, \mathbf{D}, f, T_{\mathrm{L}}) \tag{1.31}$$

/ * Update Dictionary and Mapping * /

$$\{\mathbf{D}, f\} \leftarrow \text{Cluster\_Merging}\left(\mathbf{D}, f, \hat{j}, \hat{j}'\right) \tag{1.32}$$

until $\hat{R}(\mathbf{Y}; \mathbf{D}^{\setminus \hat{j}}, f^{\hat{j}' \leftarrow \hat{j}}, \hat{\boldsymbol{\phi}}) - \hat{R}(\mathbf{Y}; \mathbf{D}, f, \hat{\boldsymbol{\phi}}) \geq 0$

for $i = 1$ to N **do**

/ * Update Mapping * /

$$f(i) \leftarrow \underset{j}{\operatorname{argmin}} - \log P(\mathbf{y}_i | \mathbf{d}_{f(i)}; \hat{\boldsymbol{\phi}}) \tag{1.33}$$

end for

return $(\mathbf{D}, f)$

}

Fig. 1.5. Pseudocode of the dictionary construction and indexing with fixed $\hat{\boldsymbol{\phi}}$. First, as the initial step, we construct much more dictionary entries (entropy-clusters) then needed. Then, we update the dictionary and mapping by iteratively selecting and merging entropy-clusters. The entropy-clusters are selected by the greedy strategy in (1.31). Afterwards, with fixed dictionary and parameter, the mapping is updated using (1.33).

By choosing different combinations of T_{S} and T_{L}, people can trade off between the compression ratio and the computing efficiency. Unfortunately, if people want

to obtain the smallest file size within the given encoding time, it is not practical to try all the possible combinations of T_{S} and T_{L}. Therefore, we conducted experiments with a lot of document images to obtain the relationship between the averaged file size l and the threshold parameters,

$$l = l(T_{\mathrm{S}}, T_{\mathrm{L}})\,, \tag{1.34}$$

as well as the relationship between the averaged encoding time τ and the threshold parameters,

$$\tau = \tau(T_{\mathrm{S}}, T_{\mathrm{L}})\,. \tag{1.35}$$

Then, the threshold parameters T_{S} and T_{L} which can achieve the smallest expected file size within the given encoding time $\tau_{\max}$ is obtained by,

$$\begin{aligned}[T_{\mathrm{S}}^*, T_{\mathrm{L}}^*] = \underset{T_{\mathrm{S}}, T_{\mathrm{L}}}{\operatorname{argmin}}\, & l(T_{\mathrm{S}}, T_{\mathrm{L}}) \\ \tau(T_{\mathrm{S}}, T_{\mathrm{L}}) \leq\, & \tau_{\max}\,.\end{aligned} \tag{1.36}$$

We introduce a slack variable λ to construct the equivalent Lagrange multiplier form.

$$[T_{\mathrm{S}}^*, T_{\mathrm{L}}^*] = \underset{T_{\mathrm{S}}, T_{\mathrm{L}}}{\operatorname{argmin}}\, l(T_{\mathrm{S}}, T_{\mathrm{L}}) + \lambda\tau(T_{\mathrm{S}}, T_{\mathrm{L}}) \tag{1.37}$$

As shown in Eq. (1.37), larger values of λ result parameter pairs with less computational cost but lower compression ratio, while smaller values of λ result parameter pairs with more computational cost but higher compression ratio.

1.6 Experimental Results

In this section, we conduct experiments by encoding a variety of documents to demonstrate the performance of our CEE-GC method in dictionary learning. First, we evaluate the bitrate approximation accuracy of our CEE method. For comparison, we also investigate the conventional methods including XOR [7], WXOR [8, 9], and cross-entropy-based pattern matching [10, 11]. Then, we conduct experiments to demonstrate that our CEE-GC method construct better dictionary entries and use

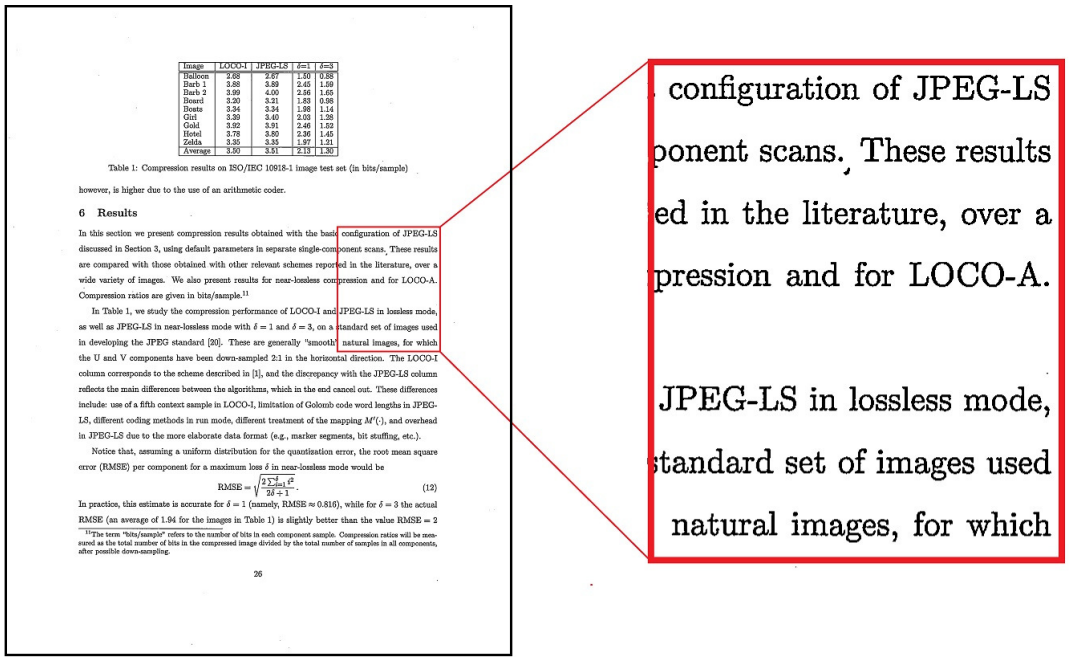


Fig. 1.6. The test image *img01.pbm* with a region (marked by the red rectangle) zoomed in.

those dictionary entries more effectively, compared with the other widely used dictionary design techniques. Last but not least, we investigated the trade off between computational cost and compression ratio.

The test images we used are 41 scanned binary document images. All of them were scanned at 300 dpi, and have size 3275×2525 pixels. The test images contain mainly text, but some of them also contain line art, tables, and generic graphical elements, but no halftones. The text in these test images has various typefaces and font sizes. The first test image *img01.pbm* is shown in Fig. 1.6 as an example. More details of the experimental procedure are provided in the following subsections.

1.6.1 Bitrate approximation

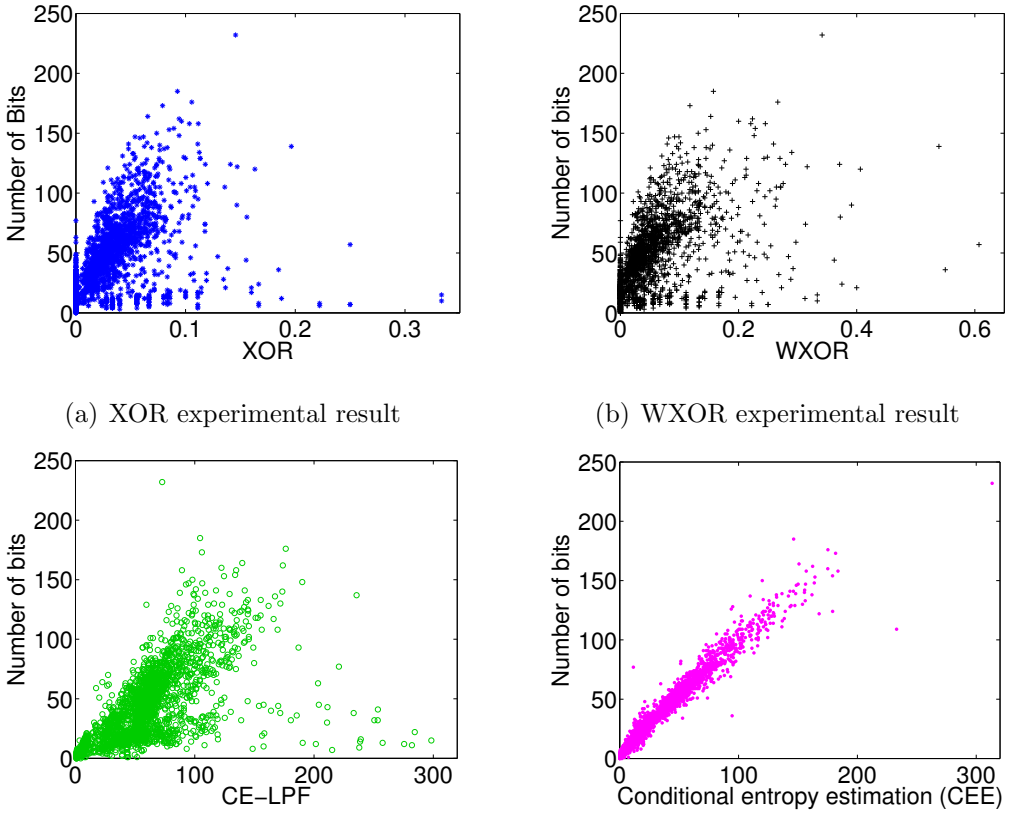
In this subsection, we evaluate the performance of our CEE method in approximating the number of bits used to encode symbols. For comparison, the following conventional dissimilarity measures are also investigated, the XOR method [7], WXOR method [8,9], and the cross entropy-based pattern matching method [10,11]. The formula for these conventional methods are listed in the Appendix for the review conveniency.

We used sample correlation to evaluate the performance of the bitrate approximation, with procedure described as follows. First, we encoded all the test images using the arithmetic entropy encoder specified in the JBIG2 standard [1]. For each of the test images, we calculated the sample correlation between the bitrate and each type of the dissimilarity measures listed in the previous paragraph. In the sample correlation calculation, each sample is a symbol extracted from the given test image. The sample measurements of the bitrate are the actual number of bits used to encode each of the symbols using its associated dictionary entry. Then, for each of the dissimilarity measures, we averaged its sample correlation with the bitrate across all the test images.

The experimental results are listed in Table 1.1. As shown, our CEE method provides a much better approximation to the bitrate in terms of sample correlation. For better visualization, in Fig. 1.7, we provide the scatterplot for the samples in the test image in Fig. 1.6. In Fig. 1.7, each sample (symbol) is represented as a dot.

1.6.2 Dictionary construction and indexing

In this subsection, we conduct experiments to empirically demonstrate that our CEE-GC method has advantages in the following aspects: dictionary size chosen, mapping estimation, and dictionary entry construction. These advantages are the underlying reasons that the encoders with our CEE-GC method achieved high com-



(a) XOR experimental result

(b) WXOR experimental result

(c) CE-LPF experimental result

(d) CEE experimental result

Fig. 1.7. Comparison of the performance of different methods for estimating the number of bits needed to encode the symbols in the image. Each dot in the figures corresponds to one symbol in the test image shown in Fig. 1.6. For each of the subfigures, the vertical coordinate of each dot is the number of bits used to encode the symbol using the associated dictionary entry in the JBIG2 bitstream. In the subfigure (a), the horizontal coordinates are the XOR between the symbols and the associated dictionary entries. In the subfigure (b), the horizontal coordinates are the WXOR between the symbols and the associated dictionary entries. In the subfigure (c), the horizontal coordinates are the CE-LPF between the symbols and the associated dictionary entries. In the subfigure (d), the horizontal coordinates are the approximation of the number of bits required to encode symbols using the associated dictionary entries using our CEE method. As can be seen in the figure, CEE achieved a much higher correlation with the number of bits used to encode symbols using the dictionary than the correlation XOR, WXOR, or CE-LPF achieved.

Table 1.1.
Performance of XOR, WXOR, CE-LPF, and CEE in estimating the number of bits needed to encode a page of symbols, as indicated by correlation.

	Averaged sample correlation
XOR	0.561 ± 0.010
WXOR	0.455 ± 0.016
CE-LPF	0.611 ± 0.018
CEE	0.955 ± 0.003

pression ratio in the experiments presented in the next subsection. The test image *img01.pbm* in Fig. 1.6 was used in this subsection.

First, the experimental results show that our CEE-GC dictionary learning method automatically generates dictionary of good size. For comparison, we first investigate the XOR-based one pass (XOR-OP) method [4]. The XOR-OP method requires one to specify a threshold parameter T_{XOR} to determine the dictionary size. In our experiment, we adjusted T_{XOR} to construct dictionaries of different sizes, and used each of these dictionaries to encode the test image separately. The results with XOR-OP are shown as the red curve in Fig. 1.8. As shown, dictionaries of different sizes lead to different bitstream file sizes. For this particular test image, the best dictionary contains 548 entries. Note for different images, the best value for T_{XOR} could be different .

Similarly, the WXOR-OP method [8, 9] also requires one to specify a threshold T_{WXOR} to determine the dictionary size. By applying the same schema used for XOR-OP, we obtained the results for the test image in Fig. 1.6 with the WXOR-OP method, shown as the black curve in Fig. 1.8.

We applied our CEE-GC method to construct a dictionary for the test image in Fig. 1.6. No parameter needed for our method to determine the size of the dictionary.

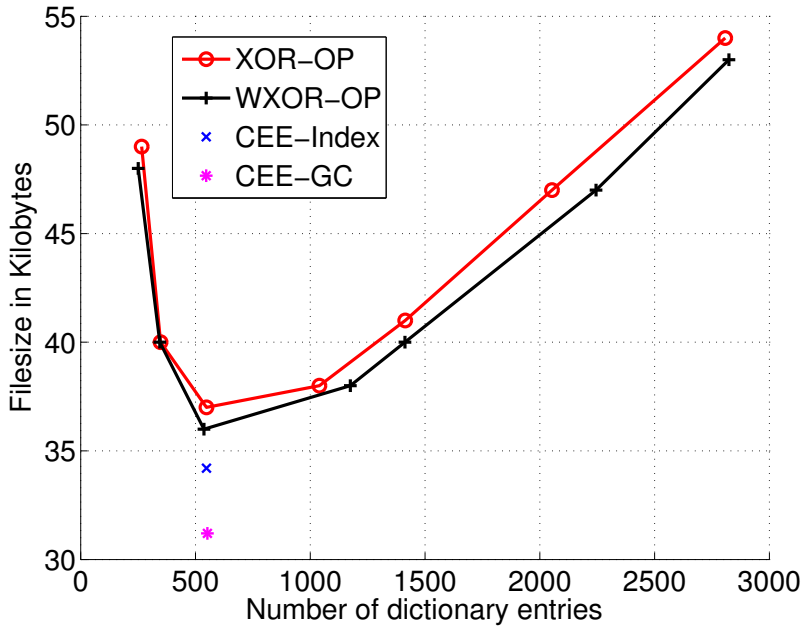


Fig. 1.8. Our CEE-based dictionary learning method can automatically obtain the dictionary of good size, and achieves high compression ratio. The XOR-OP/WXOR-OP dictionary construction method requires people to specify a parameter value before encoding. Different values of this parameter lead to dictionaries of different sizes, and different compression ratio.

As shown in Fig. 1.8, our CEE-GC method automatically constructed dictionary of good size, which provided very good balance between providing enough information to encode the image while limiting the size of itself.

Second, experimental results show that our CEE-GC dictionary learning select dictionary entries for each of the symbols more effectively. In order to evaluate the benefit only due to our CEE measurement-based dictionary selection in (1.30), we propose a method called CEE-index, described as follows. First, the dictionary entries are constructed by using WXOR-OP ($T_{\mathrm{WXOR}} = 0.27$ here). Then, for each of the symbols, its associated dictionary entry is selected using (1.30). As shown in Fig. 1.8, though the dictionary entries of CEE-index are identical to the ones of WXOR-OP,

the bitstream file size achieved with CEE-index method is still smaller than the file size with XOR-OP or WXOR-OP. This is because the CEE measurement we invented helps us use dictionary entries more effectively.

Third, experimental results show that our CEE-GC dictionary learning constructs better dictionary entries compared to the dictionary entries constructed using conventional methods. Though CEE-index and CEE-GC use the same strategy to index dictionary entries, CEE-GC achieved higher compression ratio. This is because the dictionary entries constructed by our greedy algorithm in clustering framework (GC) are more informative and better represent the image. Due to limited page space, we can not present detailed results for all the test images, but we list the compression ratio improvement for all the test images in the following subsection.

1.6.3 Compression ratio

In this section, we present the compression ratio improvement obtained by using CEE-GC. No distortion is considered since we focus on lossless compression. The one-pass (OP) dictionary construction and indexing algorithm proposed in [4] is listed for comparison due to its widely usage and comparable computing time. The lossless TIFF compression algorithm are also listed for comparison.

The OP algorithm can be based on XOR, WXOR, or CE-LPF. The compression ratio obtained by using XOR-OP [4] is sensitive to its threshold parameter T_{XOR}. In our experiment, the best T_{XOR} (in terms of compression ratio) for different images ranged from 0.01 to 0.09. Therefore, we compressed each image with XOR-OP four times using $T_{\text{XOR}} = 0.01$, 0.03, 0.06, and 0.09, respectively. Then for each test image, we averaged its bitstream file size obtained by using different T_{XOR} values. Afterwards, we averaged the bitstream file size across all the test images and report the results in Table 2.1.

The WXOR-OP method [8, 9] also requires a threshold parameter T_{WXOR}. The compression ratio obtained by using the WXOR-OP method is not sensitive to the

Table 1.2.
Compression performance comparison between different dictionary construction algorithms. The overall bitstream file size listed in the table is the sum of the bitstream file sizes of all 41 test images. The compression ratio is calculated as the raw image file size of all the test images divided by the overall bitstream file size.

Dictionary design method	Averaged file size	Compression ratio
Lossless-TIFF	53.7 KB	19.37
XOR-OP (Averaged)	35.4 KB	29.36
WXOR-OP	32.0 KB	32.54
CE-LPF-OP	32.0 KB	32.54
CEE-Indexing	30.7 KB	33.83
CEE-GC	27.8 KB	37.40

value of T_{WXOR}. In our experiment, we set $T_{\mathrm{WXOR}} = 0.27$, which produces the smallest bitstream file size for our test set, shown in Table 2.1.

Both our CEE-GC method and our CEE-index method are investigated. The parameter (control computing time) for CEE-GC in this experiment is $T_{\mathrm{S}} = 0$, $T_{\mathrm{L}} = 0.20$. The experimental results are listed in Table 2.1. The averaged bitstream file size in the table is the average of the bitstream file sizes across all the 41 test images in our test set described in the beginning of this section. The compression ratio is calculated as the raw image file size of all the test images divided by the bitstream file size.

As shown in Table 2.1, using our CEE-GC dictionary learning method improves the compression ratio by 26.26% compared with the XOR-OP (Averaged), and 14.45% compared with the WXOR-OP. Note that the bitstream we generate is still compatible with the standard JBIG2, and can be decoded by a public JBIG2 decoder.

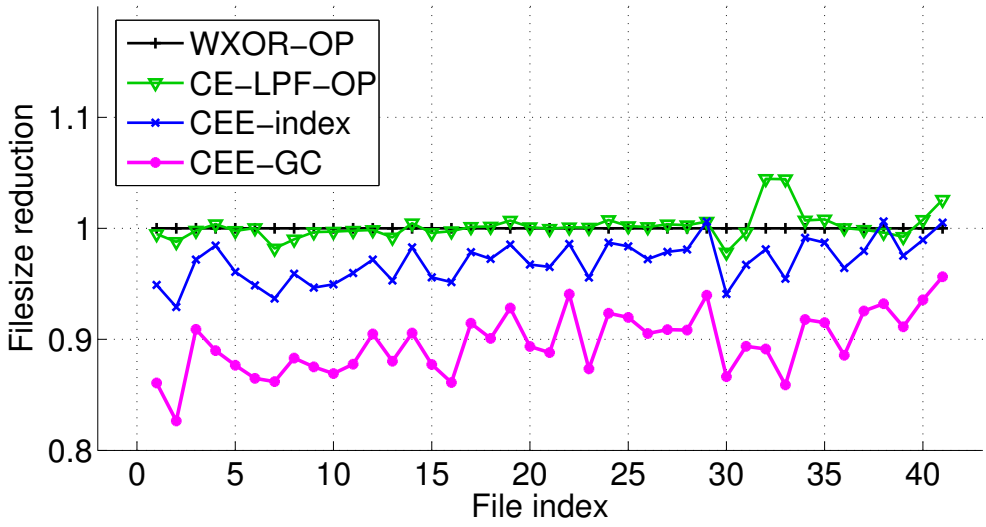
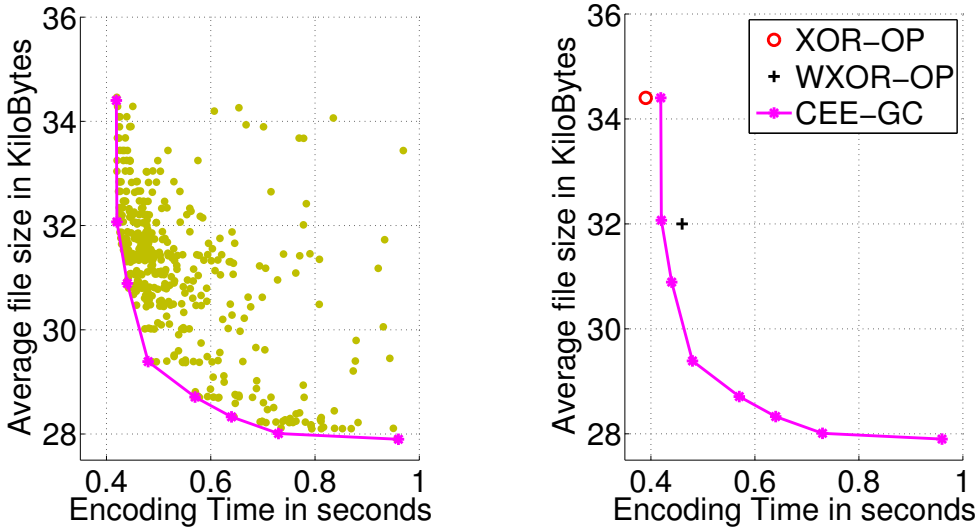


Fig. 1.9. The file size reduction (compared with the file size obtained by using WXOR-OP) for each of the test images.

More detailed experiment results on the file size reduction obtained by using our CEE-GC method for each of the test images are presented in Fig. 1.9. Since the WXOR-OP dictionary learning is widely used and considered as one of the most cutting-edge technologies, we use the bitstream file size for WXOR-OP as the baseline, and for any other given algorithm, we calculate its file size reduction as the bitstream file size using this algorithm over the bitstream file size using WXOR-OP.

As shown in Fig. 1.9, the CEE-index method reduces the file size (compared to WXOR-OP) in 38 out of 41 cases. The CEE-GC dictionary design and indexing can always produce smaller bitstream file sizes than WXOR-OP or CE-LPF-OP.

Note that our CEE measurement can also be incorporated with other dictionary construction methods, such as minimum spanning tree or K-means [12]. We do not list the results obtained by using the minimum spanning tree or K-means because the method with minimum spanning tree (or K-means) needs more than 100 seconds to construct dictionary for one page of image, as reported in [12]. The computing time



(a) CEE-GC with different prescreening parameters

(b) CEE-GC

Fig. 1.10. Different encoding time and bitstream file size obtained by using different values of T_{S} and T_{L}. Each blue dot in the figure corresponds to one case with a given pair of values of T_{S} and T_{L}.

for our method is typically less than 1 second, more details on computing time are reported in the next subsection.

1.6.4 Computing time

In the previous subsections, we mainly focus on the effectiveness of our CEE-GC method, while in this subsection, we conducted experiment to demonstrate the efficiency of our method in terms of computing time. All the 41 test images were used in this experiment. The CPU we used to conduct this experiment is only one core of AMD Opteron(tm) Processor 6172, which only has 800MHz. Either using CPU with higher frequency or using more cores with parallelized coding structure could reduce the computational time dramatically.

In our experiment, we set T_{S} and T_{L} to different values, then encoded all the 41 test images. For a given pair of values of T_{S} and T_{L}, we recorded the corresponding bitstream file size and encoding time for each of the test images. The encoding time includes the time for connected component extraction, dictionary learning, and arithmetic entropy coding. Then, we calculated the averaged bitstream file size and the averaged encoding time across the 41 test images, shown in Fig. 1.10.

In Fig. 1.10 (a), each golden dot corresponds to a pair of values of T_{S} and T_{L}. The horizontal coordinate of the dots are the averaged time to encode one document image, while the vertical coordinate of the dots are the averaged bitsream file size for one document image. By constructing the convex hull of these dots, we obtain the smallest file size for any given encoding time constraint, shown as the magenta curve in Fig. 1.10. The corresponding values of the vertexes on the boundary for this convex hull are reported in Table 1.3.

We also presented the averaged bitstream file size and encoding time with the method XOR-OP and WXOR-OP, shown in Fig. 1.10 (b). People can choose methods and set parameters according to the computing time budget and compression ratio expectation.

1.7 Conclusion

In this paper, we propose a novel dictionary learning method that can construct better dictionary entries and use those dictionary entries more effectively. There are three major novelties in our dictionary learning method. First, we propose an accurate and efficient approximation (CEE) of the number of bits required to encode a symbol using its associated dictionary entry. Second, we propose an efficient greedy algorithm in the agglomerative clustering framework to construct dictionary. Third, a prescreening strategy is designed to reduce the computational cost. We applied our dictionary learning in the JBIG2 compression and achieved promising results. The experimental results show that our CEE can provide much more accurate prediction

Table 1.3.
Trade-off between computating time and compression ratio. The parameter λ can be adjusted to balance this trade-off.

Encoding time (Second)	Averaged file size (Kilobyte)
0.42	34.4
0.42	32.1
0.44	30.9
0.48	29.4
0.57	28.7
0.64	28.3
0.73	28.0
0.96	27.8

with just a little more computational cost compared with conventional methods including XOR and WXOR. The averaged sample correlation between CEE and the number of bits used to encode each symbol in a sample document is larger than 90%, while the conventional methods can only provide an averaged sample correlation around 50%. The experimental results also show that the compression ratio obtained by using our CEE-GC dictionary learning method is about 14% higher than the compression ratio with the best existing methods.

1.8 Appendix

The conventional dissimilarity measure between symbols and dictionary entries include XOR [7], WXOR [8,9], and the cross entropy-based pattern matching method

[10, 11]. Among these conventional methods, XOR [7] and WXOR [8, 9] are widely used. The WXOR between the symbol $\mathbf{y}_i$ and dictionary entry $\mathbf{d}_j$ is computed as

$$d_{\mathrm{WXOR}}(\mathbf{y}_i, \mathbf{d}_j) = \frac{\sum_{s \in S_i} a(s)(y_i(s) - d_j(s))^2}{w_i \cdot h_i}, \tag{1.38}$$

where

$$a(s) = \frac{\sum_{r \in \{-1,0,+1\} \times \{-1,0,+1\}} [y_i(s+r) - d_j(s+r)]^2}{9}. \tag{1.39}$$

The cross entropy-based pattern matching in [10, 11] is implemented based on a low pass filter, so we denote the dissimilarity measured by using this method as $d_{\mathrm{CE-LPF}}$. The $d_{\mathrm{CE-LPF}}$ between the symbol $\mathbf{y}_i$ and dictionary entry $\mathbf{d}_j$ is computed as

$$d_{\mathrm{CE-LPF}}(\mathbf{y}_i, \mathbf{d}_j) = \sum_{s \in S_i} -\log\left(1 - |y_i(s) - \tilde{d}_j(s)|\right), \tag{1.40}$$

where $\tilde{\mathbf{d}}_j$ is obtained by applying a 3×3 low pass filter g to the dictionary entry $\mathbf{d}_j$. The filter g used in [10, 11] is fixed as

$$g = \begin{bmatrix} 1/36 & 4/36 & 1/36 \\ 4/36 & 16/36 & 4/36 \\ 1/36 & 4/36 & 1/36 \end{bmatrix}. \tag{1.41}$$

2. MODEL-BASED ITERATIVE RECONSTRUCTION FOR BINARY IMAGE COMPRESSION WITH DICTIONARY LEARNING

2.1 Problem statement

The JBIG2 standard is widely used for binary document image compression primarily because it achieves much higher compression ratios than conventional facsimile encoding standards. The noise introduced during the image procedure, including printing, scanning and quantization, of binary document image consumes extra bits, which lower the compression ratio of JBIG2 encoding. In this paper, we propose a model-based iterative reconstruction method to remove the noise introduced during the imaging procedure. The image procedure is modeled as a low pass filter and implemented as a circulant sparse matrix. We propose a patch-based prior to regularize the reconstruction procedure. The non-local information of the document image is utilized via online learning dictionary. Experiments with a variety of document images demonstrate the effectiveness of our method. No substation error was found in our test image set. Meanwhile, the compression ratio was substantially improved, compared with encoding the document image without denoising.

2.2 Reconstruction model

Maximum a posteriori (MAP) estimation is used to estimate the original noise free image $\mathbf{x} \in \{0,1\}^K$ from the observed image $\mathbf{y} \in \{0,1\}^K$. How to setup the following frame work is described in the following subsections.

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \{ -\log p(\mathbf{y}|\mathbf{x}) - \log p(\mathbf{x}) \} . \tag{2.1}$$

2.2.1 Forward model and likelihood term

Assume the original image (noise-free) is $\mathbf{x}$, we model the printing and scanning procedure as low pass filter, due to the limited resolution of these procedure. The low pass filter is denoted by A, which is a sparse and circulant matrix, satisfying the following condition,

$$\sum_{l} A_{k,l} = 1\,. \tag{2.2}$$

The scanned image is binarized. Therefore, the scanned image, denoted by $\mathbf{y}$ is,

$$\mathbf{y} = A\mathbf{x} + \boldsymbol{\epsilon}\,, \tag{2.3}$$

where the term $\boldsymbol{\epsilon} \in [-1, 1]$ is the quantization error.

Heuristically, we make the following two assumptions. First, the probability of the quantization error at the k^{th} pixel is related with the value of the error. More specifically,

$$p(y_k|\mathbf{x}) = 1 - \|y_k - \sum_{l} A_{k,l}\mathbf{x}\| \tag{2.4}$$

The second assumption is that we assume the quantization error is independently distributed, following the same distribution.

$$p(\mathbf{y}|\mathbf{x}) = \prod_{k} p(y_k|\mathbf{x}) \tag{2.5}$$

2.2.2 Prior model with dictionary learning

Since we are dealing with document images, we assume that there exists repeated patterns in the image. This non-local information could be explored and utilized in a dictionary learning procedure. Following the dictionary learning method in our previous paper, we have

$$\log p(\mathbf{x}) \propto \sum_{i} \log p(B_i\mathbf{x}|\mathbf{d}_{f(i)}) + \sum_{j} \log p(\mathbf{d}_j)\,, \tag{2.6}$$

where B_i is the operator used to extract the i^{th} symbol in the image, and $\mathbf{d}_{f(i)}$ is the j^{th} dictionary entry. The term $p(B_i\mathbf{x}|\mathbf{d}_{f(i)})$ denotes the conditional probability of the i^{th} symbol given the j^{th} dictionary entry.

2.3 Optimization

WIth the description on the likelihood term and the prior term, we obtain the cost function to be optimized as,

$$\{\hat{\mathbf{x}}, \hat{\mathbf{D}}, \hat{f}\} = \underset{\mathbf{x},\mathbf{D},f}{\operatorname{argmin}} - \sum_{k} \log(1 - \|y_k - \sum_{l} A_{k,l}\mathbf{x}\|) \tag{2.7}$$

$$- \sum_{i} \log p(B_i\mathbf{x}|\mathbf{d}_{f(i)}) - \sum_{j} \log p(\mathbf{d}_j) \tag{2.8}$$

We use alternating optimization strategy. First, we fix $\mathbf{x}$, then update $\{\mathbf{d}_{j=1}^{M}$ and f. Second, we fix $\{\mathbf{d}_{j=1}^{M}$ and f, then update $\mathbf{x}$. We repeat the above two steps until converge. At the initial stage, we set,

$$\mathbf{x} = \mathbf{y}\,. \tag{2.9}$$

2.4 Experimental result

In this section, we conduct experiments by encoding a variety of documents to demonstrate the performance of our MBIR-image compression with dictionary learning.

The test images we used are 41 scanned binary document images. All of them were scanned at 300 dpi, and have size 3275×2525 pixels. The test images contain mainly text, but some of them also contain line art, tables, and generic graphical elements, but no halftones. The text in these test images has various typefaces and font sizes. More details of the experimental procedure are provided in the following subsections.

We verified the compressed images using both tesseract-OCR and manually checked each of the symbols in the image. No substitution error was found. Meanwhile, the compression ratio was substantially improved compared with encoding method without denoising, shown in the Table 2.1.

The image quality is visually improved for some of the symbols in the image.

Table 2.1.
Compression ratio improvement by reconstruction.

Dictionary design method	Averaged file size	Compression ratio
Lossless-TIFF	53.7 KB	19.37
XOR-OP	35.4 KB	29.36
CEE-GC	27.8 KB	37.40
MBIR-DL	21.5 KB	48.01

(a) Original image (b) Scanned image (c) Estimated image

Fig. 2.1. The quality of the image obtained by MBIR-DL.

2.5 Conclusion

We proposed a model-based reconstruction with dictionary learning to reduce the noise in the scanned image. No substitution error is found in our experiments with a variety of document images. Meanwhile, the compression ratio is improved substantially, compared with the encoding without denoising preprocessing.

3. DYNAMIC HIERARCHICAL DICTIONARY DESIGN FOR MULTI-PAGE BINARY DOCUMENT IMAGE COMPRESSION

3.1 Problem statement

The JBIG2 standard is widely used for binary document image compression primarily because it achieves much higher compression ratios than conventional facsimile encoding standards. In this paper, we propose a dynamic hierarchical dictionary design method (DH) for multi-page binary document image compression with JBIG2. Our DH method outperforms other methods for multi-page compression by utilizing the information redundancy among pages with the following technologies. First, we build a hierarchical dictionary to keep more information per page for future usage. Second, we dynamically update the dictionary in memory to keep as much information as possible subject to the memory constraint. Third, we incorporate our conditional entropy estimation algorithm to utilize the saved information more effectively. Our experimental results show that the compression ratio improvement by our DH method is about 15% compared to the best existing multi-page encoding method.

3.2 Introduction

Binary document image compression is widely used for document scanning, storage, and transmission. Very often, people compress multi-page binary document images. Since these images usually come from consecutive pages of the same document source, there is typically information redundancy among pages. In this paper, we focus on how to utilize this information redundancy to improve the compression ratio for multi-page binary document image compression.

The JBIG2 compression standard, developed by the Joint Bi-level Image Experts Group [1], is considered to be the best existing standard for binary image compression because it can achieve much higher compression ratios than the previous methods, such as T.4, T.6, and T.82 [2, 29–32]. The high compression ratio of JBIG2 compression comes from its dictionary-symbol encoding procedure, briefly described as follows. A typical JBIG2 encoder works by first separating the document into connected components, or symbols. Next it creates a dictionary by encoding a subset of symbols from the image, and finally it encodes all the remaining symbols using the dictionary entries as a reference [3, 4, 12, 33, 34].

There are two straightforward methods to compress multi-page document images with the JBIG2 encoder. The first method creates an independent and static dictionary (IS) for each of the pages. Since the IS method does not utilize the information redundancy among pages, it generally results in the highest bit rate. Alternatively, the global-dictionary approach builds a single dictionary for the entire document, so it can generally achieve the lowest bit rate. However, the global-dictionary approach is not practical, since it requires enormous memory to buffer the entire multi-page document before any compression can occur.

In fact, practical JBIG2 encoders usually load only one page (sometimes even part of one page) to compress, and do not load the next page until the compression is finished [35]. Therefore, the information redundancy among pages is utilized by sharing dictionaries among pages, which is supported by the standard [1]. The first and widely cited practical algorithm, called LDM, was proposed by Ye and Cosman in [14, 36]. The LDM algorithm forms a dictionary for each of the pages by starting with the dictionary from the previous page, adding new dictionary entries needed for the current page, and expunging the dictionary entries (from the previous page) not used for the current page. The LDM algorithm improves the compression ratio compared to the IS method, and is memory efficient. Figuera, Yi, and Bouman also proposed a multi-page encoding algorithm referred to as "dynamic symbol caching (DSC)" in [9, 37]. The DSC algorithm is claimed to have higher compression ratio

compared to LDM, because DSC only discards dictionary entries when there is no space available in the memory. The discarded dictionary entries are the least recently used ones.

In this chapter, we introduce a dynamic hierarchical (DH) dictionary design method for efficient compression of multi-page documents [38]. The encoding efficiency improvement of the document symbols needs better utilizing of the information redundancy, which can be accomplished by constructing and sharing large dictionaries. However, the large dictionaries consume large number of bits to be encoded, and large size of memory to be retained. Our DH method solves these contradictory by using hierarchical structure, which encodes large dictionaries much more efficiently, and dynamic strategy, which keeps as much information subject to the memory constraint. In addition, we use the conditional entropy estimation technique of [21,39] to measure the information redundancy between two dictionary entries. This results in further improvements to the encoding efficiency of the dictionary design and symbol encoding processes. Our experimental results shows that the DH method improves compression by approximately 15% relative to the best existing methods.

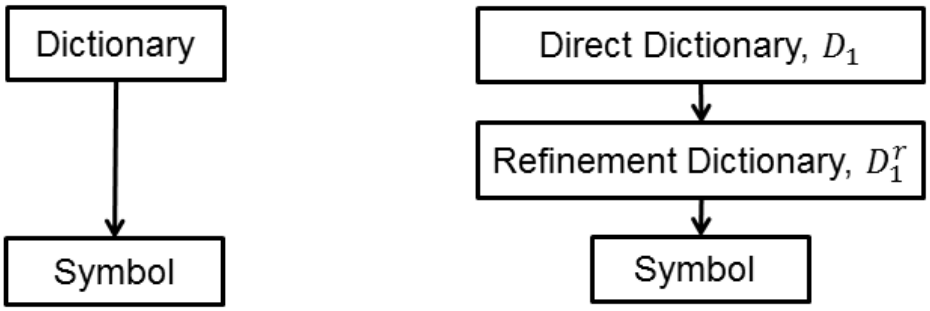
The rest of the paper is organized as follows. We present the dynamic hierarchical dictionary design method in details in Sec. 3.3. The experimental results are shown in Sec. 3.4 and the conclusion is in Sec. 3.5.

3.3 Dynamic hierarchical design

In this section, we introduce the dynamic update approach for encoding of hierarchical dictionaries. Sec. 3.3.1 explains how the first page is encoded. Sec. 3.3.2 and Sec. 3.3.3 then explain the dynamic updating strategy for successive pages.

3.3.1 Encoding of First Page

The Figs. 3.1 (b) illustrates how the hierarchical dictionary structure is used to encode the first page of the document. First, we encode a smaller dictionary, called



(a) Single dictionary (b) Hierarchical dictionary

Fig. 3.1. Single dictionary and hierarchical dictionary structure for the first page. The hierarchical dictionary structure efficiently encodes a large refinement dictionary, which more efficiently encodes the document symbols.

the direct dictionary, denoted as D_1. This direct dictionary is encoded using the direct coding mode of the JBIG2 standard [1]. Next, we use the refinement coding mode of the JBIG standard [1] to encode a much larger refinement dictionary, denoted by D_1^r. The refinement dictionary is compressed very efficiently because refinement coding uses a reference symbol from the direct dictionary to encode each new symbol in the refinement dictionary. Finally, we encode all the symbols in the first document page by using the refinement dictionary as a reference.

In order to construct the hierarchical dictionary of Figs. 3.1b), we use a bottom-up procedure. First, we extract all the distinct symbols on the first page. Then, for each of the distinct symbol, we construct one refinement dictionary entry. Next, we group the similar refinement dictionary entries into clusters, and create one representative for each of the clusters. These representatives are the dictionary entries which form the direct dictionary. In order to perform the clustering, we use the conditional entropy estimation-based dictionary indexing and design algorithm (CEE-DI) of [21].

3.3.2 Encoding of Successive Pages

The Fig. 3.2 illustrates the dynamic hierarchical dictionary construction of successive pages of the document. For successive pages, we introduce the stored dictionary, denoted as D_k^s, for the k^{th} page ($k \neq 1$). When there is no memory constraint, the stored dictionary D_k^s is the union of all dictionaries from the previous pages (of which indices $< k$). The case with memory constraint will be discussed in the next subsection.

Once again, the refinement dictionary D_k^r is formed by every unique symbols in the k^{th} page. For each of the given refinement dictionary entries, we try to find a good match in the stored dictionary, D_k^s, to encode it efficiently. Typically, most of the entries in the refinement dictionary will have a good match in the stored dictionary. Thus, in this case, the refinement dictionary is encoded very efficiently.

However, there will typically be some refinement dictionary entries, that do not have a good match in the stored dictionary. In order to encode these unmatched refinement dictionary entrees, we form a new direct dictionary D_k. We build this direct dictionary using the conditional entropy estimation-based dictionary indexing algorithm (CEE-I) [21].

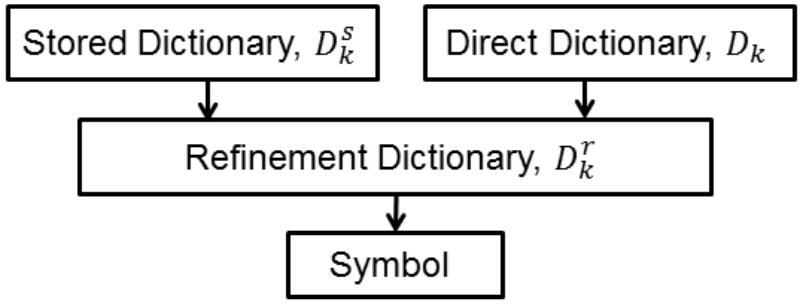


Fig. 3.2. Hierarchical dictionary structure for the k^{th} page. The stored dictionary D_k^s is the pruned union of all the dictionaries from previous pages. Some entries in the refinement dictionary are encoded using the stored dictionary, while the rest are encoded using the direct dictionary.

The criteria to determine whether we can find a good match for the given refinement dictionary entry in the stored dictionary is based on the conditional entropy estimation (CEE) in [21]. Let $\boldsymbol{d}^r_{k,j}$ denote the j^{th} entry in D^r_k, and $\boldsymbol{d}^s_{k,i}$ denote the i^{th} entry in D^s_k. The best match for $\boldsymbol{d}^r_{k,j}$ in D^s_k is found by

$$\boldsymbol{d}^s_{k,\hat{i}(j)} = \underset{\boldsymbol{d}^s_{k,i} \in D^s_k}{\operatorname{argmin}} \hat{H}(\boldsymbol{d}^r_{k,j} | \boldsymbol{d}^s_{k,i}) \,, \tag{3.1}$$

where $\hat{H}(\boldsymbol{d}^r_{k,j} | \boldsymbol{d}^s_{k,i})$ is the estimation of the conditional entropy of $\boldsymbol{d}^r_{k,j}$ given $\boldsymbol{d}^s_{k,i}$. If the conditional entropy of $\boldsymbol{d}^r_{k,j}$ given $\boldsymbol{d}^s_{k,\hat{i}(j)}$ is smaller than the predefined threshold T_R,

$$\hat{H}(\boldsymbol{d}^r_{k,j} | \boldsymbol{d}^s_{k,\hat{i}(j)}) \leq T_R \,, \tag{3.2}$$

we encode $\boldsymbol{d}^r_{k,j}$ using the stored dictionary entry $\boldsymbol{d}^s_{k,\hat{i}(j)}$ as a reference. Otherwise, we do not encode $\boldsymbol{d}^r_{k,j}$ using the stored dictionary.

3.3.3 Dynamic updating strategy

In order to make our algorithm practical, we must ensure that the size of the dictionaries retained in the memory will not grow beyond our available memory size.

The method we used is to discard some of the stored dictionary entries whenever the memory size for all the dictionaries for the k^{th} page is larger than 1M bytes. We choose the threshold value to be 1M because the standard requires a decoder to have at least a 1M byte of storage for the dictionary [35].

The memory size for the dictionaries for the k^{th} page is the summation of the memory size for D_k, D^r_k, and D^s_k, which can be calculated using the formula in [35]. The entry to be discarded is the stored dictionary entry $\boldsymbol{d}^s_{k,\hat{m}}$ satisfying both of the following two conditions.

1. The entry $\boldsymbol{d}^s_{k,\hat{m}}$ is not referred by any entry in D^r_k.

2. The entry $\boldsymbol{d}^s_{k,\hat{m}}$ is least distinct, defined as

$$\hat{m} = \underset{m}{\operatorname{argmin}} \, d_{XOR}(\boldsymbol{d}^s_{k,m}, \boldsymbol{d}_{k,n}) \,, \tag{3.3}$$

where $\boldsymbol{d}_{k,n}$ is any dictionary entry different from $\boldsymbol{d}^s_{k,m}$, that belongs to D_k, D^r_k, or D^s_k. The function d_{XOR} calculates the Hamming distance between two dictionary entries.

Similar dictionary entries typically have more mutual information. Therefore, by the above strategy, we maintain as much total information as possible in the memory under the memory size constraint.

3.4 Experimental results

In this section, we compare the DH method with the best existing method DSC of [9, 37]. The DSC method in our experiments is based on the widely used weighted Hamming distance (WXOR) [8, 40] for the dissimilarity measurement between symbols and dictionary entries. For the DH method, we investigated two versions. The first one is as described in the Sec. 3.3, called DH-CEE, since it uses the conditional entropy estimation (CEE) as the dissimilarity measure. For the second one, we substitute the CEE dissimilarity measure with the WXOR dissimilarity measure, in order to see the benefit due only to the dynamic hierarchical dictionary design. We call this method DH-WXOR.

The test image set contains three documents, *EEPaper*, *Vita*, and *I9intro*. Each of them was scanned from consecutive pages of the same document at 300 dpi. *EEPaper*, contains 9 images with 3275×2525 pixels. *Vita* contains 10 images with 3300×2550, while *I9intro* contains 6 images with 3300×2550. These test images contain mainly text, but some of them also contain line art, tables, and graphical elements, but no halftones. The JBIG2 lossless text mode was used for all experiments.

We limited the memory usage for dictionaries to be less than 1MB, as described in 3.3.3. Unless otherwise stated, we adjusted the parameters of all the methods so that each of the methods achieved its optimal compression ratio.

3.4.1 Compression ratio improvement

In the experiment in this subsection, all the document images were compressed. The following plot in Fig. 3.3 shows a comparison of the DH-CEE method to the alternative methods in encoding the document *EEPaper*. In the plot, everything is relative to the independent and static dictionary constructed with the WXOR dissimilarity measure (IS-WXOR). Notice that, our method DH-CEE has the highest compression ratio for all the pages. For the entire document *EEPaper*, the DH-CEE improved the compression ratio by 14% compared to DSC, while 15% for *Vita* and 17% for *I9intro*.

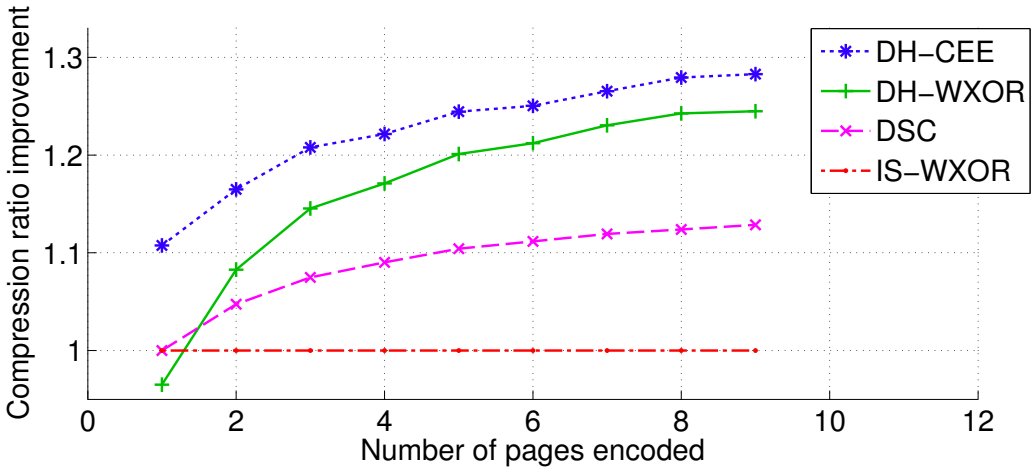
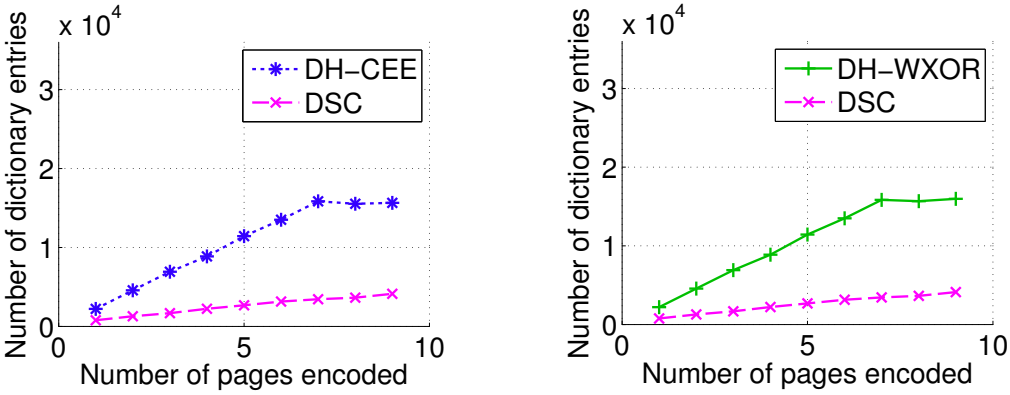


Fig. 3.3. Compression ratio improvements by different dictionary design methods (relative to IS-WXOR).

The main reason for the compression ratio improvement by DH-CEE over DSC is that DH-CEE produced a much larger dictionary for each of the pages, shown in Figs. 3.4 (a), and the larger dictionaries can more efficiently encode the documents. Meanwhile, using DH-CEE, only a small overhead is needed to encode the large dictionary. The efficient encoding of large dictionaries is demonstrated with an example in the next subsection.



(a) DH-CEE vs. DSC (b) DH-WXOR vs. DSC

Fig. 3.4. The number of the dictionary entries obtained by using different dictionary design methods. The large dictionaries produced by the DH methods encode the documents efficiently. For DH-CEE and DH-WXOR, the dynamic updating controls the size of their dictionaries after the 7^{th} page due to the memory constraint.

3.4.2 Efficient encoding of large dictionaries

In this subsection, we show that the DH method allows us to encode a large dictionary with a relatively little overhead using the following experiment. We used DH-CEE and DSC to create large dictionaries for the first page in *EEPaper*, and compared the numbers of bits they used to encode their dictionaries.

The refinement dictionary produced by the DH-CEE method is large in size, because DH-CEE creates one refinement dictionary entry for each of the distinct symbols in the page. For the DSC algorithm, we adjusted its parameters to obtain a single dictionary, which is as large as the refinement dictionary with DH-CEE.

As shown in Fig. 3.5, the bitstream filesize obtained by using DH-CEE are significantly smaller than that obtained with DSC. This is due to the hierarchical structure of DH-CEE: DH-CEE builds up the direct dictionary using CEE-ID to encode the refinement dictionary efficiently.

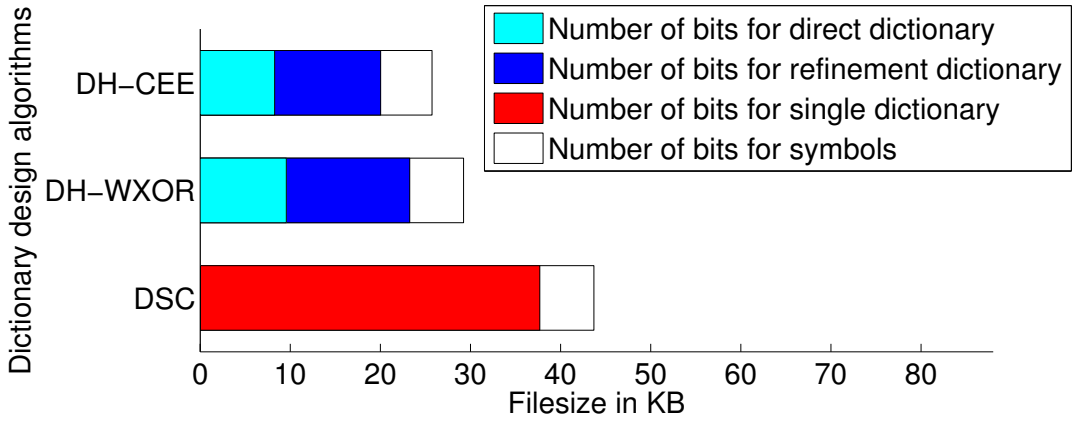


Fig. 3.5. Comparison of bit rate using three methods for dictionary compression. Note that DH-CEE results in the smallest encoded dictionary.

3.4.3 Conditional entropy estimation

The compression ratio improvement by DH-CEE also comes from the conditional entropy estimation (CEE). For comparison, we investigate the DH-WXOR method, which substitutes CEE with WXOR.

First, we repeated the single page experiment in Sec. 3.4.2 with the DH-WXOR method. The refinement dictionaries obtained by using DH-WXOR and DH-CEE are identical since they used the same method to create their refinement dictionaries. As shown in Fig. 3.5, the bit rate obtained by using DH-WXOR is smaller than that with DSC because of the hierarchical dictionary design. One the other hand, the bit rate with DH-WXOR is larger than that of DH-CEE. This is because CEE used in DH-CEE provides better measurement for the information redundancy between dictionary entries than WXOR in DH-WXOR [21]. And thus DH-CEE creates a better direct dictionary to encode the refinement dictionary.

Then, we repeated the multi-page experiment in Sec. 3.4.1 with the DH-WXOR method. As shown in Figs. 3.3, DH-WXOR improved the compression ratio by 11% compared to DSC. This improvement purely comes from the large dictionaries produced by the dynamic hierarchical design of DH-WXOR, shown as Figs. 3.4 (b). On

the other hand, The compression ratio with DH-WXOR is about 4% less than that of DH-CEE. This is because, based on CEE, DH-CEE creates better direct dictionaries and selects better stored dictionary entries to encode the refinement dictionary in Eq. (3.1) than DH-WXOR.

Please note that all the above experiments were conducted subject to the 1MB memory constraint. As shown in Figs. 3.4 (a), The dynamic updating has kept discarding stored dictionary entries since the 7^{th} page was encoded. If we release the memory constraint, no dictionary entries are discarded and extra memory is consumed. However, according to our experimental results, the extra memory usage only improved the compression ratio by less than 1%. This is because the dynamic updating minimizes the information loss caused by discarded dictionary entries by selecting the least distinct stored dictionary entries.

3.5 Conclusion

In this paper, we proposed the dynamic hierarchical dictionary design method (DH) for the multi-page binary document image compression. A typical dictionary design method for multi-page image improves the encoding efficiency by maintaining and utilizing the information from previous pages to encode the successive pages. The DH method outperforms the previously existing methods using the following technologies. First, hierarchical design allows us to maintain more information per page. Second, the dynamic updating helps us to maintain as much information as possible subject to the memory size constraint. Third, the conditional entropy estimation helps us to utilize the maintained information more efficiently.

4. TEXT LINE DETECTION BASED ON COST OPTIMIZIED LOCAL TEXT LINE DIRECTION ESTIMATION, AND APPLICATIONS IN REGISTRATION AND SEARCH

4.1 Problem statement

Text line detection is a critical step for applications in document image processing. In this paper, we propose a novel text line detection method. First, the connected components are extracted from the image as symbols. Then, we estimate the direction of the text line in multiple local regions. This estimation is, for the first time, formulated in a cost optimization framework. We also propose an efficient way to solve this optimization problem. Afterwards, we consider symbols as nodes in a graph, and connect symbols based on the local text line direction estimation results. Last, we detect the text lines by separating the graph into subgraphs according to the nodes' connectivity. Preliminary experimental results demonstrate that our proposed method is very robust to non-uniform skew within text lines, variability of font sizes, and complex structures of layout. Our new method works well for documents captured with flat-bed and sheet-fed scanners, mobile phone cameras and with other general imaging assets.

4.2 Introduction

Text line detection is a critical step for tasks such as document layout analysis [41, 42]. Text line detection with low computational cost and high accuracy is still considered as an open problem for complex document images or natural images. The complexity of document images comes from irregular layout structure, a mixture

of machine printed and hand written charters, and/or the variability of text line directions.

Many methods have been proposed for accurate text line detection. One category of popular methods are top-down approaches, such as the Hough transform-based methods [43–45]. Hough transform-based methods detect text lines using the "hypothesis-validation" strategy: The potential text lines (collinear alignments of symbols), are hypothesized in the Hough domain, and validated in the image domain. This "hypothesis-validation" strategy is computationally expensive. Moreover, in order to deal with non-straight text lines, or complex layout structure, extra pre-processing and post-processing strategies are required to make this method robust.

Another category of popular methods are the smearing methods in [46, 47], which follow a bottom-up scheme. The basic idea is to grow the text line region by recursively finding and incorporating the closest characters. Compared with Hough transform-based methods, smearing methods can deal with fluctuating lines better. But the smearing methods contain parameters that need to be accurately and dynamically tuned. A nice review of the text line detection methods can be found in [48–51].

We also notice that a lot pre/post processing methods have recently been proposed to improve the text line detection performance. One pre-processing example is the edge-enhancement based connect component extraction [41, 52]. In the same paper [41], a text line is rejected (as post processing) if a significant portion of the objects in this text line are repetitive. Our paper is purely focus on the text line detection, but can potentially incorporate with these pre-post processing.

In our paper, we propose a novel text line detection method composed of two algorithms that are applied in sequence: the local text line direction estimation (LTDE), and the graphical model-based text line construction (GMTC). In the LTDE algorithm, we, for the first time, to the best of our knowledge, define a cost function based on the following two observations. First, within a relatively small region, the symbols in the same text line tend to fall on a straight line. Second, along the direction of the

text line, the symbol density is higher than the symbol density is in the other directions. We propose an efficient way to solve this optimization problem. By optimizing the cost function, we obtain the text line direction in the local region, and simultaneously the collinearity relationship for symbol pairs within the local region. In GMTC, we build a graphical model by considering each of the symbols as a node. Our model is different from the graphical models previous used for text line detection since the messages passed between nodes in our model are from the local collinearity obtained by our LTDE. Then, we group symbols into text lines by separating the graph into subgraphs based on the local text line direction estimation results. Experiments with a variety of images demonstrate that the proposed method is very fast and robust to non-uniform skew within text lines, variability of font sizes, and complex structures of layout.

In section 4.3, we present our text line detection method. In section 4.4, we present experimental results.

4.3 Text line detection

A text line consists of a set of symbols, the centroids of which fall on a smooth curve. In this section, we propose a text line detection method.

First, we extract the N symbols from the image, denoted by $\{\mathbf{s}_i\}_{i=1}^{N}$, [21, 38, 53]. The centroid of the i^{th} symbol is denoted by $\mathbf{x}_i = (x_{i,1}, x_{i,2}, 1)^T$, where $x_{i,1}$ and $x_{i,2}$ are its horizontal and vertical coordinates, respectively. In this paper, we use homogeneous coordinate to introduce compactness in our formulation. Then, the text line is constructed according to the geometric locations of the centroids of the symbols, with details described in the following subsections.

4.3.1 Local text line direction estimation

Although text lines may contain non-uniform skew, within a relatively small region, the centroids of the symbols in the same text line tend to fall on a straight line.

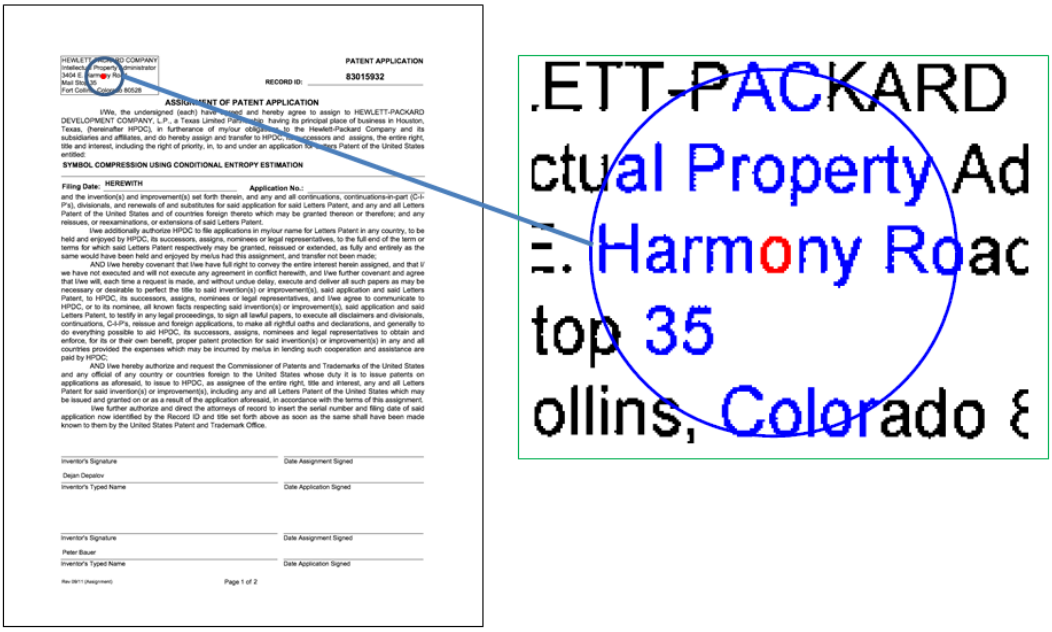


Fig. 4.1. We estimate the direction of text lines in multiple local regions. This figure shows an example of the local region we use. For each given symbol $\mathbf{s}_i$, e.g. the letter "o" colored red in the figure, we define a small region centered at its centroid. The local direction of the text line containing $\mathbf{s}_i$ is estimated according to the location of the symbols $\mathbf{s}_i$ and $\mathbf{s}_{\partial i}$ in the local region. Here, the term ∂i denotes the indexes of the symbols within the local region centered at the centroid of $\mathbf{s}_i$. As shown in the figure, the set of symbols $\mathbf{s}_{\partial i}$ are colored blue. The left figure is the original document image, while the right figure is obtained by zooming in this document image.

Therefore, we estimate the directions of the text lines in different local small regions separately.

More precisely, we go through all the symbols. For the i^{th} symbol, we define a local region centered at its centroid $\mathbf{x}_i$. All the symbols in this local region are denoted by $\mathbf{s}_{\partial i}$. An example is shown in Fig. 4.1. In the local region centered at $\mathbf{x}_i$, the direction of the text line containing $\mathbf{s}_i$ is denoted as θ_i, while the location of the

text line is controlled by a scalar $\beta_{i,3}$. Following the homogenous coordinates used in the definition of $\mathbf{x}_i$, we define a vector

$$\boldsymbol{\beta}_i = [\cos\theta_i, \sin\theta_i, \beta_{i,3}]^T \ . \tag{4.1}$$

With this homogenous framework, it can be shown that the distance between the centroid $\mathbf{x}_j$ and the straight line represented by $\boldsymbol{\beta}_i$ is

$$d(\mathbf{x}_j, \boldsymbol{\beta}_i) = |\boldsymbol{\beta}_i^T \mathbf{x}_j| \, . \tag{4.2}$$

In addition to the above assumption on the local collinearity of symbol centroids, we estimate $\boldsymbol{\beta}_i$ based on the assumption that the symbol density is higher along the text line direction than the symbol density is in other directions. Since not all the symbols in $\mathbf{s}_{\partial i}$ belong to the same text line, we propose a binary variable $\lambda_{i,j}$ to describe whether the j^{th} symbol belongs to the text line containing the i^{th} symbol.

$\lambda_{i,j} = 1 \leftrightarrow \mathbf{s}_j$ belongs to the text line containing $\mathbf{s}_i$;

$\lambda_{i,j} = 0 \leftrightarrow \mathbf{s}_j$ does not belong to the text line containing $\mathbf{s}_i$;

With the notation above, we design the following cost function to estimate $\boldsymbol{\beta}_i$ and $\boldsymbol{\lambda}_i = \{\lambda_{i,j} | j \in \partial i\}$,

$$\left\{\hat{\boldsymbol{\beta}}_i, \hat{\boldsymbol{\lambda}}_i\right\} = \underset{\boldsymbol{\beta}_i, \boldsymbol{\lambda}_i}{\operatorname{argmin}} \left\{ \sum_{j \in \partial i} \alpha_j \left|1 - \lambda_{i,j}\right| \right.$$
$$\left. + d(\mathbf{x}_i, \boldsymbol{\beta}_i)^2 + \sum_{j \in \partial i} \lambda_{i,j} d(\mathbf{x}_j, \boldsymbol{\beta}_i)^2 \right\} \ , \tag{4.3}$$

where the term $d(\mathbf{x}_i, \boldsymbol{\beta}_i)^2 + \sum_{j \in \partial i} \lambda_{i,j} d(\mathbf{x}_j, \boldsymbol{\beta}_i)^2$ accounts for the local collinearity assumption. During the minimization procedure, this term produces the estimation of $\boldsymbol{\beta}_i$ by fitting a straight line onto the centroids of symbols with non-zero $\lambda_{i,j}$. Simultaneously, this term tends to prune out symbols from the local text line by encouraging $\lambda_{i,j}$ to be 0.

On the other hand, the penalty term $\sum_{j \in \partial i} \alpha_j \left|1 - \lambda_{i,j}\right|$, encourages the local text line to contain as many symbols as possible. Without this term, after the minimization, none of the symbols in $\mathbf{s}_{\partial i}$ will be considered to be in the text line containing $\mathbf{s}_i$,

since purely minimizing the second line of (4.3) would make all the $\lambda_{i,j}$ to be 0. The strength of the penalty term is controlled by the parameter α_j. We design the parameter α_j to introduce geometric meaning to our cost function. Suppose the width and height of the j^{th} symbol are denoted by w_j and h_j, respectively. The parameter α_j is calculated as

$$\alpha_j = \left(\frac{\max\{w_j, h_j\}}{2} \right)^2 . \tag{4.4}$$

The way we calculate α_j has the following geometric interpretation. Calculating the partial derivative of (4.3) according to $\lambda_{i,j}$, we get

$$\text{If } d(\mathbf{x}_j, \boldsymbol{\beta}_i) \leq \frac{\max\{w_j, h_j\}}{2} \text{, then } \lambda_{i,j} = 1 \tag{4.5}$$

$$\text{If } d(\mathbf{x}_j, \boldsymbol{\beta}_i) > \frac{\max\{w_j, h_j\}}{2} \text{, then } \lambda_{i,j} = 0 \,. \tag{4.6}$$

The result in (4.5) shows that if the straight line $\boldsymbol{\beta}_i$ passes through the symbol $\mathbf{s}_j$, $\lambda_{i,j} = 1$, which means that the location of the symbol $\mathbf{s}_j$ contributes to the estimation of $\boldsymbol{\beta}_i$. On the contrary, as shown in (4.6), if the straight line $\boldsymbol{\beta}_i$ does not cut through the symbol $\mathbf{s}_j$, $\lambda_{i,j} = 0$, which means the symbol $\mathbf{s}_j$ is not considered to belong to the text line with $\mathbf{s}_i$; and the location of $\mathbf{s}_j$ will not contribute to the estimation of $\boldsymbol{\beta}_i$.

We design an alternating optimization scheme to solve (4.3). Multiple initial conditions are applied to handle the non-convexity of the cost function; and the matrix transform strategy in [39, 54] is applied to improve the computational efficiency. Due to space limitations, we do not describe the details of the optimization here. After we obtain $\hat{\boldsymbol{\beta}}_i$, we calculate the direction of the local text line containing the symbol $\mathbf{s}_i$ as

$$\hat{\theta}_i = \arctan(\frac{\hat{\beta}_{i,2}}{\hat{\beta}_{i,1}}) \,. \tag{4.7}$$

4.3.2 Text line construction

After we obtain $\hat{\boldsymbol{\beta}}_i$ and $\hat{\boldsymbol{\lambda}}_i$, $i = 1, \ldots, N$, we set up a graphical model $G = (V, E)$ to cluster the symbols into text lines. Here, the term V and E denote the set of nodes and the set of edges, respectively. Each of the symbols is considered as a node in the

graph. For any two symbol nodes $\mathbf{s}_i$ and $\mathbf{s}_j$, The probability that the symbol nodes $\mathbf{s}_i$ and $\mathbf{s}_j$ belong to the same text line is calculated as

$$p_{i,j} = \lambda_{i,j}\lambda_{j,i}\delta(\theta_i, \theta_j)\,. \tag{4.8}$$

In order to obtain a very efficient method, we here design $\delta(\theta_i, \theta_j)$ to have a binary output.

$$\begin{aligned}\delta\left(\theta_i, \theta_j\right) &= 1, \text{if}\,|\theta_i - \theta_j| \leq \theta_{\max} \\ &= 0, \text{otherwise}\,.\end{aligned} \tag{4.9}$$

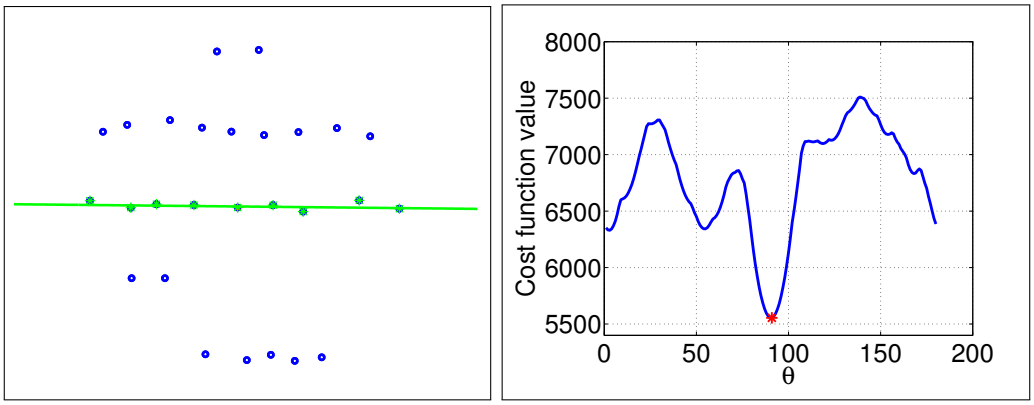
Empirically, we set $\theta_{\max} = \frac{\pi}{6}$. At the cost of greater computation, a continuous output model for $\delta(\theta_i, \theta_j)$ with a more sophisticated graph inference methods, such as loopy belief propagation or expectation propagation [27, 28, 55] could improve the robustness of our method.

With the graph G constructed, we construct text lines using the following procedure.

1. Mark all the symbol nodes as unclustered.
2. Pick any unclustered symbol node in the graph G as the source node, and find all the reachable symbol nodes using the breadth-first search (BFS) algorithm [56]. These reachable symbols and the source symbol are considered to belong to the same text line.
3. Mark these symbols as clustered.
4. Repeat Step 2 and Step 3 until all the symbols are clustered.

4.4 Experimental results

We conducted experiments with a variety of document images to demonstrate the effectiveness of our text line detection method. We also demonstrate that the proposed method can be applied to improve the accuracy of a state-of-art text detection method [53].



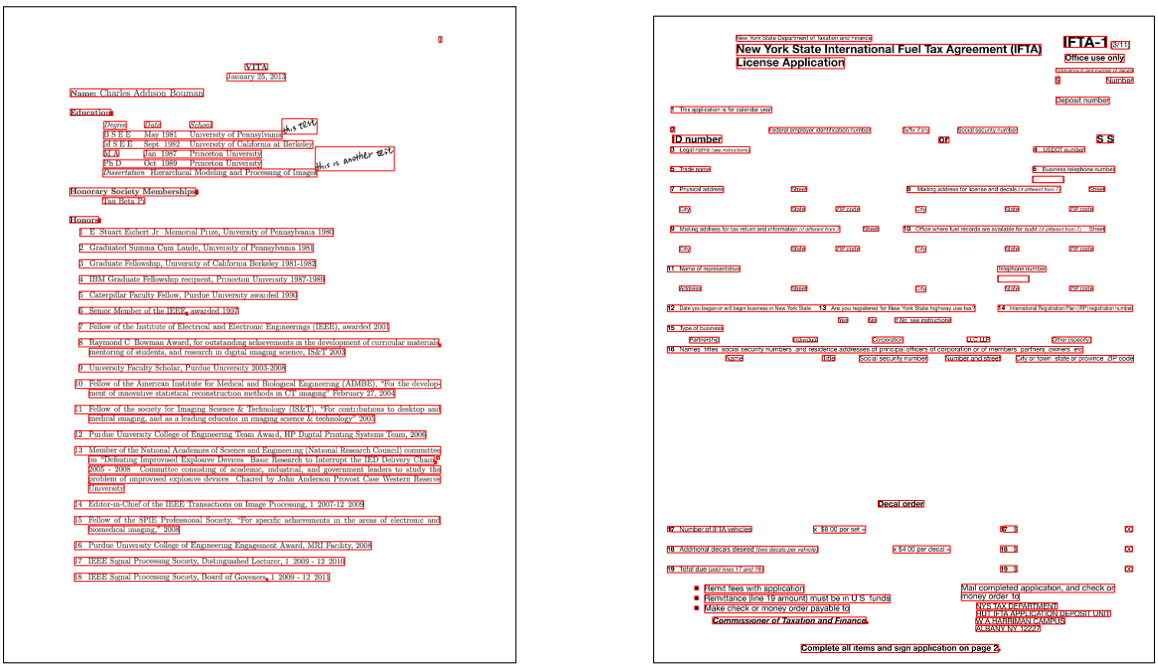
(a) Symbol centroids in local region (b) Cost value in (4.3) with different $\boldsymbol{\beta}$ values

Fig. 4.2. Example of the local text line direction estimation. In subfigure (a), each of the dots represents the centroid of a symbol in the local region shown in Fig. 4.1. Fixing $\boldsymbol{\beta}_i$ to have different values, we minimized the cost function in (4.3) over $\boldsymbol{\lambda}_i$, and show the result in subfigure (b). As indicated in the figure, the best value for θ_i is 90.6°, corresponding to the green line in subfigure (a).

4.4.1 Local direction estimation

In this subsection, we show an example to demonstrate our local text line direction estimation. The local region used in this subsection is shown in Fig 4.1; and the centroids of the symbols in this region are shown in Fig 4.2. (a).

In our experiment, we obtained the value of the cost function in (4.3) with different $\boldsymbol{\beta}_i$ using the following scheme. We gradually changed the value of θ_i from 1° to 180°. For each fixed $\boldsymbol{\beta}_i = [\cos\theta_i, \sin\theta_i, \beta_{i,3}]^T$, we minimized (4.3) over $\boldsymbol{\lambda}_i$. The minimum values of the cost function given different θ_i values are shown in Fig. 4.2. (b). As indicated in the figure, when $\theta = 90.6°$, the overall minimum value was obtained, which corresponds to the straight line shown as the green line in Fig. 4.2 (a).



(a) Test case 1

(b) Test case 2

Fig. 4.3. Text line detection example. Each of the the text lines in the image is marked with a red rectangle.

4.4.2 Text line detection in complex document image

In this subsection, we conduct experiments with complex document images to demonstrate the robustness of our algorithm. Some of the challenging cases are shown in Fig. 4.3. The text lines detected using our algorithm are marked with red rectangle. Figure. 4.3 (a), shows a difficult case for smearing methods. Since the smearing methods keep trying to find the closest symbols to merge, the closely located hand written text line and the printed text line tend to be considered as the same text line. But as shown in Fig. 4.3 (a), our algorithm successfully separated the hand written text line from the machine printed text lines. This is because our method connects symbols according to the local text line direction information. Fig. 4.3 (b) shows that our method can detect text lines in very complicated layout environment.

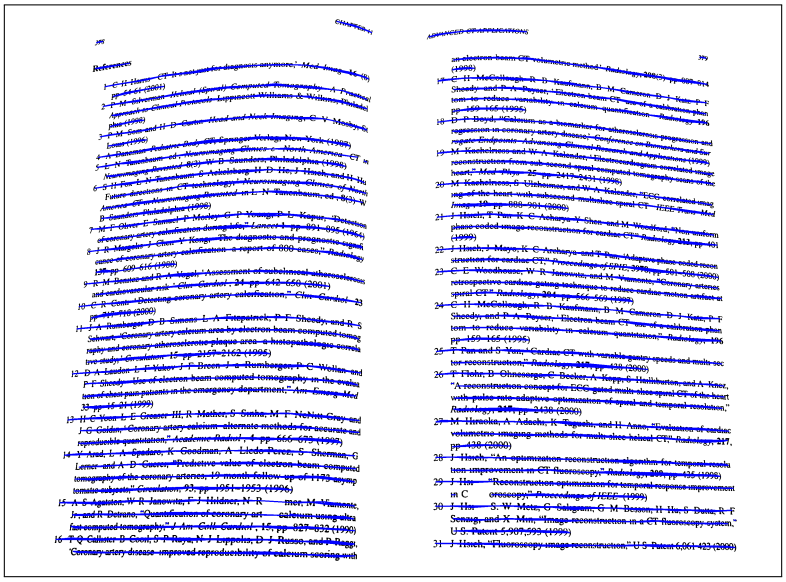


Fig. 4.4. Text line detection example. Text lines are tracked by blue lines.

Another example is in Fig. 4.4, which shows that our method can detect text lines that are not straight in a complex layout structure, while a typical Hough-based method cannot.

4.4.3 Text detection in natural images

In this subsection, we demonstrate that our text line detection can be used to improve the accuracy of the the text detection in natural images. In our experiment, the natural images were first fed into the cost optimized segmentation (COS) algorithm in [53]. COS output a binary image, in which the text symbols are detected with very low mis-detection rate. In [53], a training-based connected component classification (CCC) method is applied to reduce the false alarm in the output of COS. Both the COS and CCC results are show in Fig. 4.5, and Fig. 4.6. We applied our text line detection method to the output of COS, and only consider the connected components

(a) Natural image 1

(b) COS result

(c) CCC result

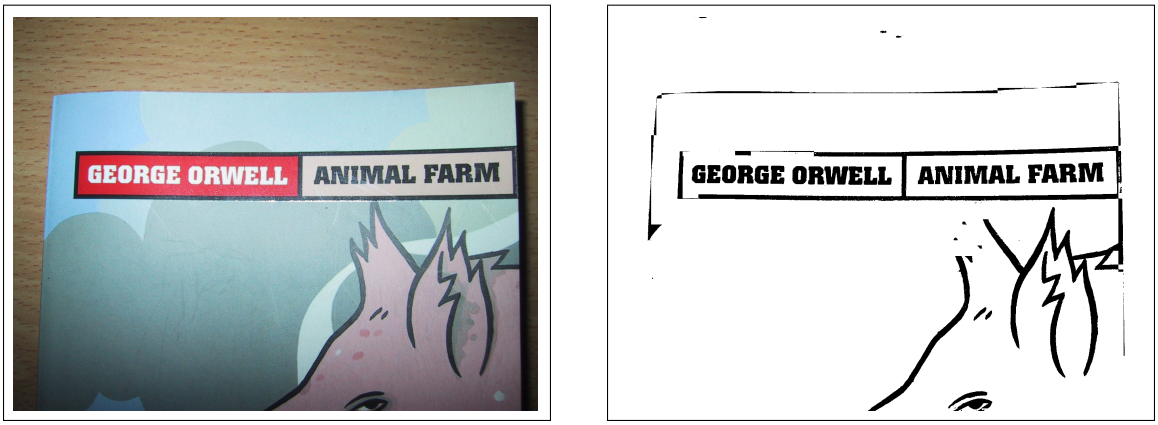
(d) Our algorithm

Fig. 4.5. Performance in text detection in natural image I.

with a line structure as text symbols. The results are shown in the forth column of Fig. 4.5, and Fig. 4.6.

4.4.4 Text detection in image registration

In this subsection, we present a novel image registration method designed for binary document images. The reference images we are using are obtained by rastering the original pdf image, called template image. The sensed images are obtained by



(a) Natural image 2

(b) COS result

(c) CCC result

(d) Our algorithm

Fig. 4.6. Performance in text detection in natural image II.

scanning the printed image, called query image. The deformation in the query image includes skew, a bit wrapping, extra information, white noise, etc. The basic procedure is as follows. First, we detect text lines for both the template and the scanned images. Then, for each of the text lines, construct a feature vector for each of the text lines. The feature vector consists of the size, position, and pixel value information of the corresponding text line. Then we match the text lines between the query and template image. Figure 4.7 demonstrates our idea.

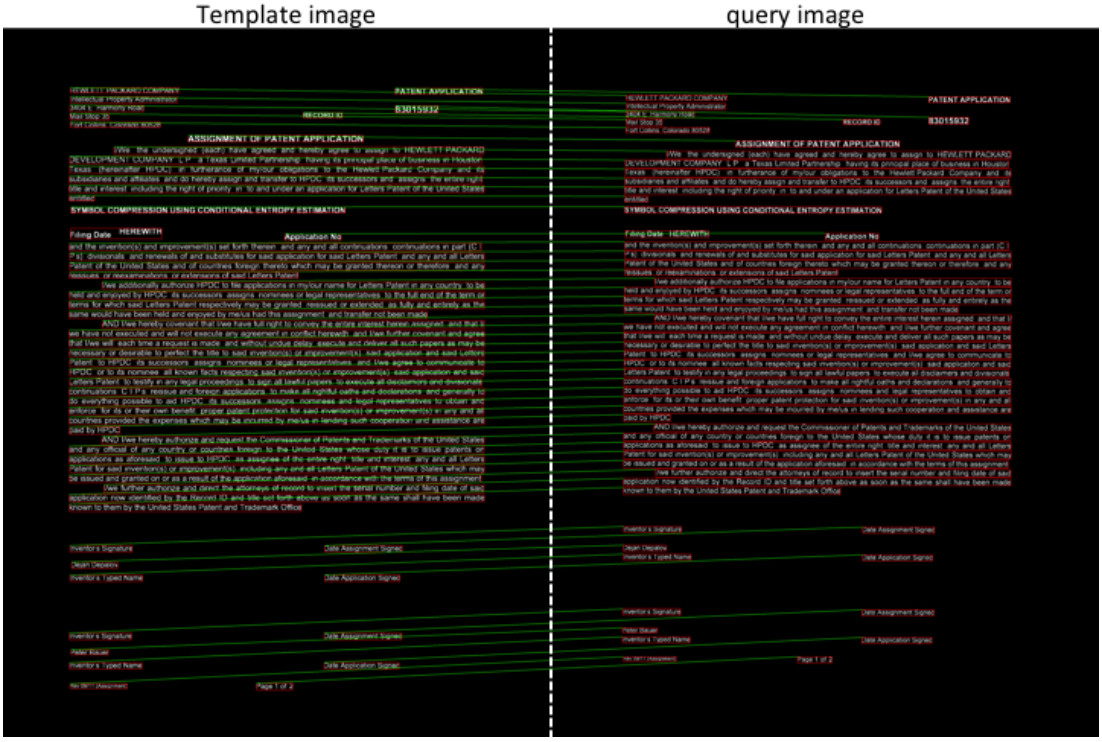


Fig. 4.7. Binary document image registration results.

4.4.5 Binary document image search results

We built up a template image database by collecting blank forms from USCIS, IRS Tax return, Purdue business and academic forms, and Test forms provided by HP. In total, we have 341 blank forms. Then, we collected 54 filled out forms by randomly selecting and letting lab-mates fill template forms. The basic procedure of our search system is shown in Figure 4.8. This is a small experiment just to illustrate the effectiveness of our method in document images. We can extend the image search scale by incorporating this to a larger indexing and searching system.

We achieved 100% accuracy in our database. Two search results are show in Figure 4.9 and Figure 4.10 as illustrations.

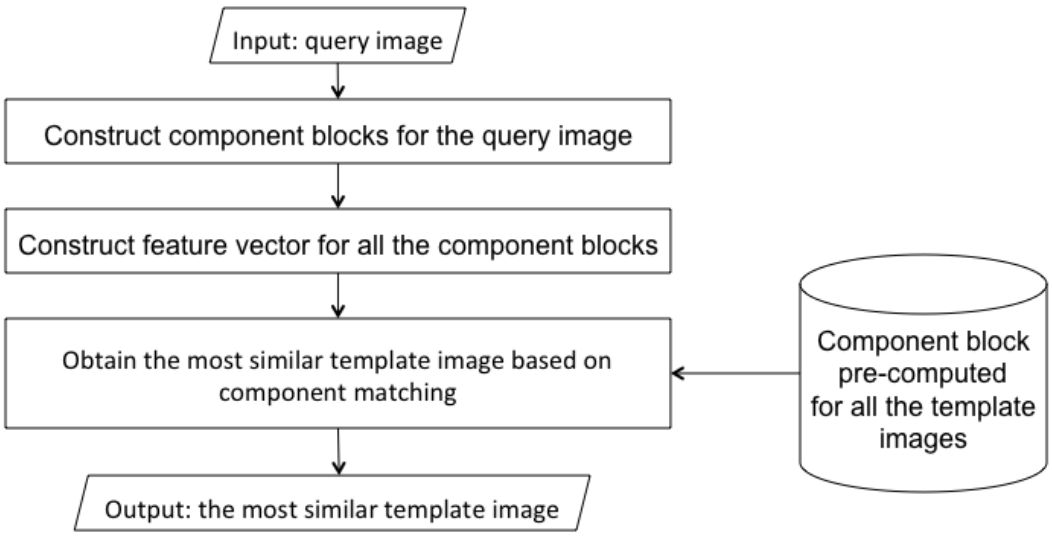


Fig. 4.8. Binary document image search flow chart.

4.5 Appendix

In order to solve the cost function (4.3), we set initial values for $\boldsymbol{\beta}_i$, then alternatively update $\boldsymbol{\lambda}_i$ and $\boldsymbol{\beta}_i$ until converge. Since this cost function (4.3) is not a convex problem, we start with different initial conditions for $\boldsymbol{\beta}_i$. For each of the initial condition, we obtain one estimate for $\boldsymbol{\beta}_i$ and $\boldsymbol{\lambda}_i$. The estimate with the smallest cost

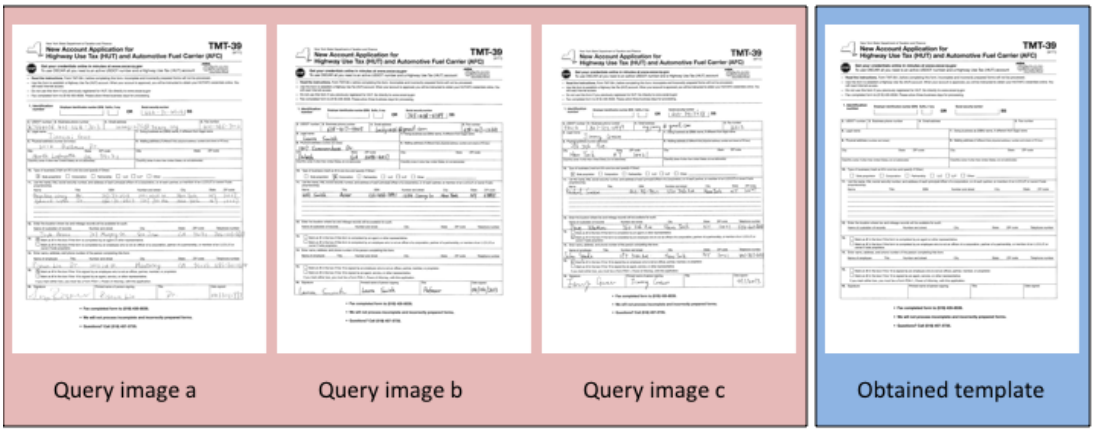


Fig. 4.9. Binary document image search result I.

function value is chosen to be the final estimate. In our experiments, we used 4 initial values for $\boldsymbol{\beta}_i$.

$$\boldsymbol{\beta}_i = [0, 1, -y_i]^T \tag{4.10}$$

$$\boldsymbol{\beta}'_i = [1, 0, -x_i]^T \tag{4.11}$$

$$\boldsymbol{\beta}''_i = [0.71, -0.71, -0.71 \times x_i + 0.71 \times y_i]^T \tag{4.12}$$

$$\boldsymbol{\beta}'''_i = [0.71, 0.71, -0.71 \times x_i - 0.71 \times y_i]^T \tag{4.13}$$

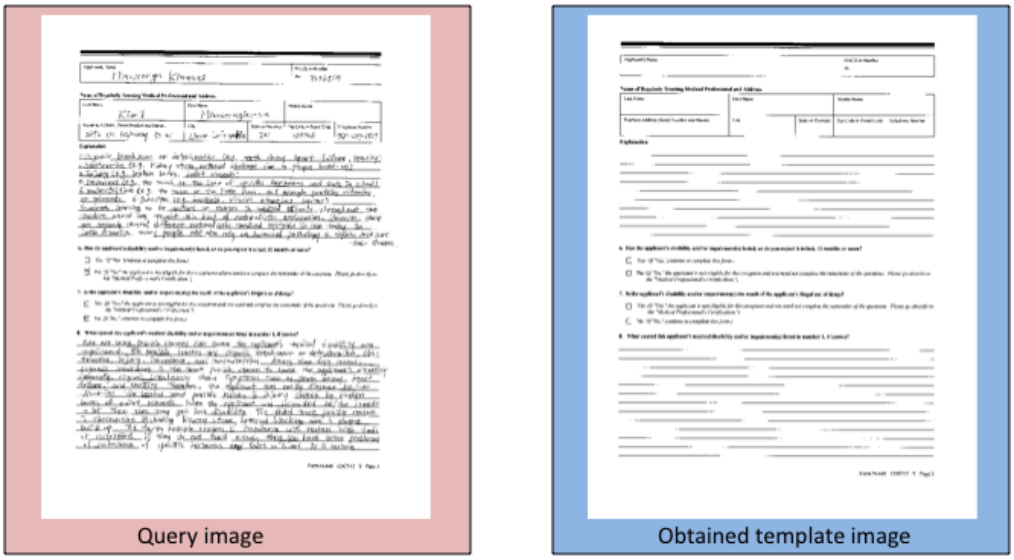


Fig. 4.10. Binary document image search result II.

The estimate of $\lambda_{i,j}, j \in \partial i$ with $\boldsymbol{\beta}_i$ fixed has been shown in (4.5) and (4.6). Here we show how to update $\boldsymbol{\beta}_i$ with $\boldsymbol{\lambda}_i$ fixed. Let X be the matrix of which the columns consists of $\mathbf{x}_i$ and $\{\lambda_{i,j}\mathbf{x}_j | j \in \partial i\}$, we have

$$\boldsymbol{\beta}_i^* = \underset{\boldsymbol{\beta}_i}{\operatorname{argmin}} \boldsymbol{\beta}_i^T X X^T \boldsymbol{\beta}_i + \sum_{j \in \partial i} \alpha_{i,j} |1 - \lambda_{i,j}| \tag{4.14}$$
$$\text{subject to } \beta_{i,1}^2 + \beta_{i,2}^2 = 1$$

Let $R = XX^T$. We decompose R into the following form using the matrix transform in [39, 54].

$$\begin{aligned} R &= \begin{bmatrix} R_{1,1} & R_{1,2} & R_{1,3} \\ R_{2,1} & R_{2,2} & R_{2,3} \\ R_{3,1} & R_{3,2} & R_{3,3} \end{bmatrix} \\ &= \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &\times \begin{bmatrix} \tilde{R}_{1,1} & 0 & \tilde{R}_{1,3} \\ 0 & \tilde{R}_{2,2} & \tilde{R}_{2,3} \\ \tilde{R}_{3,1} & \tilde{R}_{3,2} & R_{3,3} \end{bmatrix} \times \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{4.15}$$

where

$$\theta = \frac{1}{2} \arctan \frac{-2R_{2,1}}{R_{1,1} - R_{2,2}} \tag{4.16}$$

Without loss of generality, we can decompose the vector $\boldsymbol{\beta}_i$ as

$$\boldsymbol{\beta}_i = k_{i,1} \begin{bmatrix} \cos(\theta) \\ -\sin(\theta) \\ 0 \end{bmatrix} + k_{i,2} \begin{bmatrix} \sin(\theta) \\ \cos(\theta) \\ 0 \end{bmatrix} + k_{i,3} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{4.17}$$

Substituting (4.15) and (4.17) back to the cost function (4.14) results the update of $\boldsymbol{\beta}_i$.

5. HIGH DIMENSIONAL REGRESSION USING THE SPARSE MATRIX TRANSFORM (SMT)

5.1 Problem statement

Regression from high dimensional observation vectors is particularly difficult when training data is limited. More specifically, if the number of sample vectors n is less than dimension of the sample vectors p, then accurate regression is difficult to perform without prior knowledge of the data covariance.

In this paper, we propose a novel approach to high dimensional regression for application when $n < p$. The approach works by first decorrelating the high dimensional observation vector using the sparse matrix transform (SMT) estimate of the data covariance. Then the decorrelated observations are used in a regularized regression procedure such as Lasso or shrinkage. Numerical results demonstrate that the proposed regression approach can significantly improve the prediction accuracy, especially when n is small and the signal to be predicted lies the subspace of the observed signal corresponding to the small eigenvalues.

5.2 Introduction

Regression from high dimensional observation vectors is particularly difficult when training data is limited. Traditional regression methods that use the sample covariance, such as the ordinary least squares (OLS) approach, perform poorly in this situation. This is because, if the sample number n is less than data dimension p, then the sample covariance is singular, with at least $p - n$ of the smallest eigenvalues estimated to be zero. In this case, the sample covariance does not accurately charac-

terize any signal that falls in the subspaces corresponding to the smallest eigenvalues of the observations.

In the past decades, regression methods that adopt regularization have been introduced, such as ridge regression [57], subset selection, and Lasso [58]. More recently, there has also been increasing interest in replacing the sample covariance with some sparse estimates of the true covariance or its inverse for high dimensional regression problems [59, 60].

In this paper, we propose a novel regression approach that first decorrelates the high dimensional observation vector using the sparse matrix transform (SMT) estimate of the covariance [54]. To improve the prediction accuracy, model selection is then performed by regularizing the regression in the domain of the decorrelated data. In particular, we explore the use of both Lasso and shrinkage methods for this regularized regression step. While the technique we propose can be used with other estimates of the covariance, we have found that SMT covariance estimation results in relatively good estimates particularly when $n < p$ [54, 61]. The SMT covariance estimate achieves this improved accuracy by imposing a constraint that the eigenvector transformation should be formed by a product of sparse rotations.

Our numerical results demonstrate that the both the SMT-Lasso and SMT-Shrinkage regression methods can significantly improve the prediction performance when $n < p$, and that, for our experiments, the SMT-Lasso method yields better results than the SMT-Shrinkage method, but at the cost of greater computational cost.

5.3 Regression Model

Without loss of generality, let $y \in \Re^{n\times 1}$ be a vector of n i.i.d. zero-mean Gaussian random variables which we would like to estimate. Our observations are $X \in \Re^{n\times p}$, a matrix containing n independent zero mean Gaussian random row vectors, each of

dimension p. The minimum mean square error (MSEE) estimate of y given X has the form

$$\hat{y} = Xb \tag{5.1}$$

where b is a vector of regression coefficients given by

$$b = R_x^{-1}\rho \ , \tag{5.2}$$

where $R_x = \frac{1}{n}\mathrm{E}[X^t X]$ is the covariance of the observations X, and $\rho = \frac{1}{n}\mathrm{E}[X^t y]$ is the correlation between the observations X and y.

Of course, in practice R_x and ρ are often unknown, so that b must be estimated from training data (y, X). This problem has been widely studied over the years, and most recently has become of particular interest in the challenging case when $n < p$. The traditional method for solving the regression problem is ordinary least squares (OLS). However, OLS becomes ill-posed when $n < p$, so partial least squares (PLS), ridge regression [57], and the Lasso [58] have been proposed as alternatives.

5.4 SMT Regression for High Dimensional Data

Our approach will be to estimate y based on the assumption that we can accurately estimate the covariance R_x. Our approach is motivated by a variety of new methods for estimating high dimensional covariance matrices through the use of sparsity constraints [54, 62]. In particular, we will use the recently introduced SMT covariance estimation method, which has been shown to produce substantially more accurate covariance estimates for certain physical data through the introduction of a covariance model [54, 61]. Importantly, the SMT covariance estimate can accurately produce all the eigenvalues of the covariance even when $n < p$, and the resulting estimate is typically full rank.

Perhaps surprisingly, we will demonstrate that even when the exact value of the covariance is used in Eq (5.2), the resulting regression coefficients may yield estimates that are inferior to established methods. Intuitively, this is because the correlation

ρ must also be accurately determined. Therefore, having an accurate estimate of R_x does not insure success.

Therefore, our proposed regression approach is based on two steps. In the first step, we decorrelate the observations using the estimate of R_x. In the second step, we estimate the regression coefficients in this decorrelated domain. We propose three possible methods for estimating these regression coefficients. The first method, which we refer to as SMT-Lasso, applies the Lasso regression method in the decorrelated domain. The second method, which we refer to as SMT-Shrinkage, shrinks the regression coefficients toward zero; and the third method, which we refer to as SMT-subset selection, simply selects the coordinates which are most correlated with the y.

For all the three methods, we use the SMT covariance estimate to decorrelate the observation data in the first step. Let $\hat{R}_x$ be the covariance estimate, and let the eigen decomposition of the covariance estimate be given by

$$\hat{R}_x = \hat{E}\hat{\Lambda}\hat{E}^t \ , \tag{5.3}$$

where $\hat{E}$ is the orthonormal matrix of eigenvectors and $\hat{\Lambda}$ is a diagonal matrix of eigenvalues. Using these estimated quantities, we can approximately decorrelate and whiten the observed data using the transformation

$$\tilde{X} = X\hat{E}\hat{\Lambda}^{-\frac{1}{2}} \ . \tag{5.4}$$

For the SMT covariance estimate, the entries of the diagonal matrix $\hat{\Lambda}$ are generally positive, so the matrix is invertible.

Using the whitened observations $\tilde{X}$, the estimate of y can now be expressed as

$$\hat{y} = \tilde{X}\beta \ . \tag{5.5}$$

Since $\tilde{X}$ is a linear transformation of the observations X, it does not change the OLS estimate. However, it can change other regression results based on nonlinear estimators of the regression coefficients.

An important special case occurs if the observations are perfectly whitened. In this case, $\frac{1}{n}E\left[\tilde{X}^t\tilde{X}\right] = I$, and the regression parameters for MMSE estimation are given by

$$\beta = \frac{1}{n}\mathrm{E}[\tilde{X}^t y] \ . \tag{5.6}$$

The question remains of how to compute an effective estimate of β. For this purpose, we propose three methods.

5.4.1 SMT-Lasso

The first method we propose for estimating β, which we call SMT-Lasso, is based on the use of least squares minimization with an L_1 norm constrain on β [58]. The SMT-Lasso estimate is given by

$$\hat{\beta} = \arg\min_{\beta}\left\{\parallel y - \tilde{X}\beta \parallel^2 + \lambda \parallel \beta \parallel_1\right\} \ . \tag{5.7}$$

Notice that the solution to (5.7) depends on the specific whitening transformation used in (5.4) because the L_1 norm is not invariant to orthonormal rotations. Therefore, SMT-Lasso will produce a different solution from conventional Lasso performed in the native coordinates of X. Since the columns of the matrix $\tilde{X}$ are not exactly orthogonal, the SMT-Lasso regression coefficients are computed as the solution to a quadratic programing problem. As with conventional Lasso, this optimization problem can be computed using a variety of efficient techniques [58, 63].

5.4.2 SMT-Shrinkage

The second method we propose for estimating β, which we call SMT-Shrinkage, is based on the approximation that the columns of $\tilde{X}$ are orthogonal, and have an L_2 norm of n. More specifically, we assume that

$$\frac{1}{n}\tilde{X}^t\tilde{X} \approx I \ . \tag{5.8}$$

In fact, the form of the SMT covariance estimate ensures that $\frac{1}{n}\mathrm{diag}\left\{\tilde{X}^t\tilde{X}\right\} = I$. So the columns of $\tilde{X}$ have a constant norm of n. In addition, the columns are

approximately orthogonal because SMT covariance estimation attempts to minimize correlation between columns subject to a constraint on the required number of sparse rotations.

Using this approximation, the solution to (5.7) can be computed by first solving the unconstrained optimization problem to yield

$$\hat{\beta} = \frac{1}{n}\tilde{X}^t y \ , \tag{5.9}$$

and then applying the soft shrinkage operator to yield

$$\hat{\beta}_{\gamma i} = \text{sign}(\hat{\beta}_i)(\mid \hat{\beta}_i \mid -\gamma)^+ \ , \text{for } i = 1, 2, \ldots, p \ , \tag{5.10}$$

where the operator $(\cdot)^+$ returns the positive portion of the argument. Notice that this soft shrinkage operator has the same form of wavelet shrinkage for image denoising.

5.4.3 SMT-Subset Selection

The third method we propose for estimating β, which we call SMT-Subset Selection, is similar to SMT-Shrinkage, but it selects a subset of decorrelated components for the regression, rather than shrinking the components toward zero.

For SMT-Subset selection, we first compute $\hat{\beta}$ of (5.9). We then apply the following operation to each component of $\hat{\beta}$

$$\hat{\beta}_{\gamma i} = \begin{cases} \hat{\beta}_i & \text{if } \mid \hat{\beta}_i \mid > \gamma \\ 0 & \text{otherwise} \end{cases} \ . \tag{5.11}$$

Notice that this operation selects the components that have the largest correlation with the observations to be predicted.

5.5 Numerical Experiments

Let $X \in \Re^{n \times p}$ be a matrix containing n independent observations of a p-dimensional vector. These observations are formed as

$$X = y \cdot \tau^t + W \ . \tag{5.12}$$

(a) (b) (c)

Fig. 5.1. (a) Simulated color IR view of airborne hyperspectral data over Washington DC Mall [64]. (b) Ground-truth pixel spectrum of grass. (c) Ground-truth pixel spectrum of water.

where $\tau \in \Re^{p\times 1}$ is a deterministic but unknown signal, $y \in \Re^{n\times 1}$ is a vector of n independent Gaussian random variables to be estimated, and each row of $W \in \Re^{n\times p}$ is an independent p-dimensional Gaussian random vector of correlated noise or clutter. Without loss of generality, we assume that W, y, and X are all zero mean, and that elements of y have unit variance.

To demonstrate the effectiveness of the proposed SMT regression methods, numerical examples of high dimensional regression are performed. In these examples, the signal and the clutter have a dimension of $p = 191$. The "true" covariance of the clutter $R_w = \frac{1}{n}\mathrm{E}[W^t W]$ is computed from real hyperspectral data [64]; and we know $R_x = R_w + \tau\tau^t$. Spectra of some ground-truth grass and water samples are shown in Fig. 5.1. Different cases of τ and W will be investigated.

Performance of different regression models is tested on another independently generated dataset of 300 observations using the achieved signal-to-noise ratio (SNR),

$$SNR = \frac{\| y \|^2}{\| y - X\hat{b} \|^2} \ . \tag{5.13}$$

Next, we first use two specific cases of t to demonstrate the SMT regression methods in detail. More aggregate results are then provided. Among all these experiments, 3-fold cross validation was used to determine the values of the regularization parameters.

5.5.1 When τ Is a Small Eigenvector

Here, we investigate the case that the signal τ falls in small eigenvector subspaces of R_w. We choose τ to be the 170-th eigenvector of R_w with $\| \tau \|^2 = 3^2$. Here, R_w is computed using hyperspectral grass data [64]. A total number of $n = 100$ training samples is used.

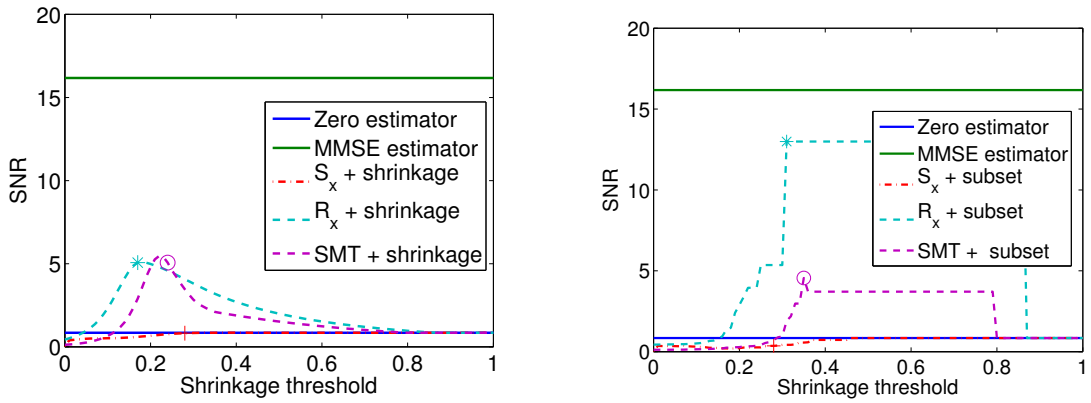
Figure 5.2(a) shows the SNR as a function of shrinkage threshold (γ in (5.10)) after the data were whitened using different covariance estimates; Similarly, Fig. 5.2(b) shows the SNR as a function of subset selection threshold (γ in (5.11)). The marker points represent the achieved SNRs using the threshold values chosen by cross validation. The SNRs using the zero estimator and the optimal MMSE estimator are also listed for comparison.

Notice that both shrinkage and subset selection significantly improve the regression performance after the data were whitened using the true covariance or the SMT estimate. However, if the sample covariance estimate is used, neither shrinkage nor subset selection improves the regression performance. This is because the signal exists in the small eigenvector subspaces that can not be represented by the sample covariance.

5.5.2 When τ Is a Random Signal

Here, τ is generated as a Gaussian random vector with zero mean and covariance I. Again, R_w is the "true" covariance of the hyperspectral data of grass class. A total number of $n = 100$ training sample is used.

Figure 5.3 shows the results for a specific example of τ: Fig. 5.3(a) shows the SNR as a function of shrinkage threshold after the data were whitened using different covariance estimates; Fig. 5.3(b) shows the SNR as a function of subset selection threshold. It may be surprising to notice that the SMT-Shrinkage regression even works slightly better than the true covariance-shrinkage as shown in Fig. 5.3(c).



(a) Shrinkage

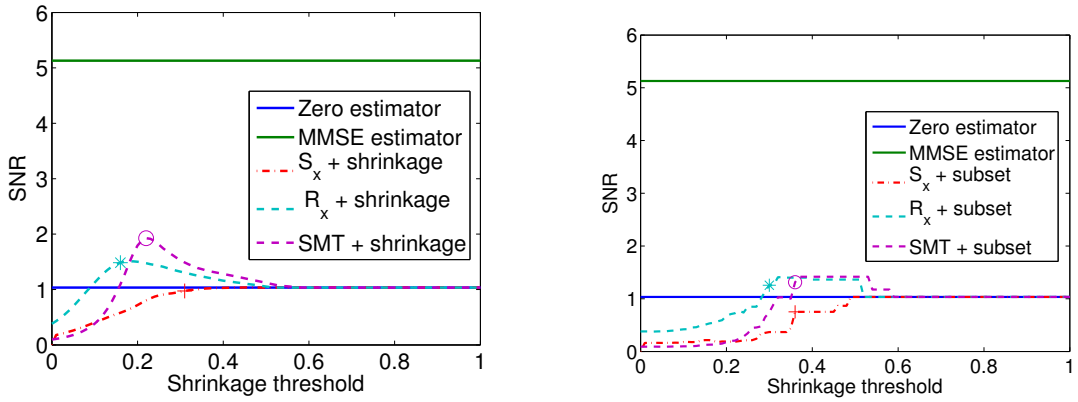
(b) Subset selection

Fig. 5.2. Results when t is the 170-th eigenvector of R_w: (a) SNR versus shrinkage threshold. (b) SNR versus subset selection threshold. The marker points represent the achieved SNRs using shrinkage thresholds chosen by cross validation.

5.5.3 Aggregate Results

Here, more aggregate experiments are performed for further assessment of different regression methods. Traditional Lasso and ridge regression are included for comparison. Here, τ is generated as a Gaussian random vector as above. The amplitude of t is scaled in order to investigate different SNR cases. R_w is computed as the "true" covariance matrix of hyperspectral data of grass and water, respectively. $n = 50, 100, 200$ and 300 training samples are used for building the regression models, respectively. For each case, 30 repeated experiments are run, and the average SNR is calculated. For each run, τ and W are re-generated.

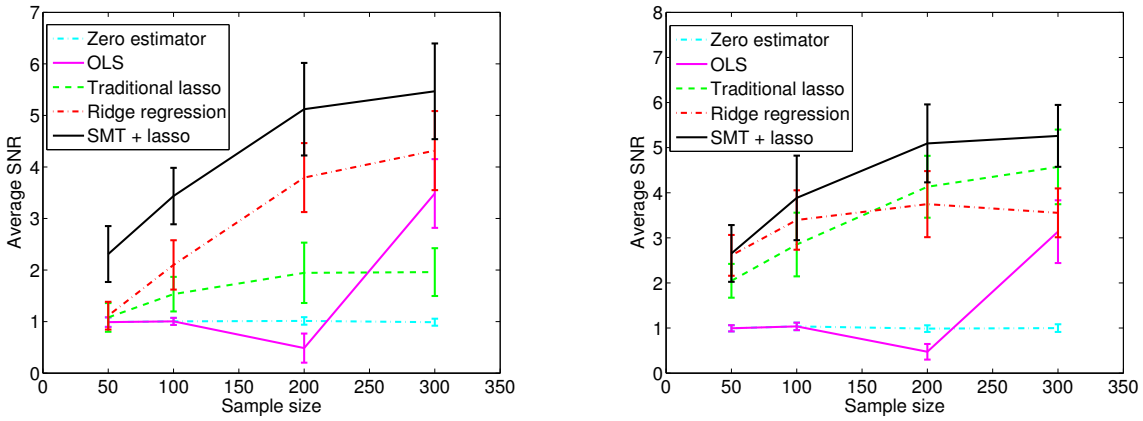
Figure 5.4 shows the plots of the average SNRs. Notice that the SMT-Lasso regression results in significantly higher SNRs, especially in the range of $n < p$. Figure 5.5 shows the SNRs of SMT-Lasso, SMT-Shrinkage and SMT-Subset selection, respectively. SMT-Lasso performs best, but are much more computationally expensive.



(a) Shrinkage

(b) Subset selection

Fig. 5.3. Results when t is a generated random Gaussian signal: (a) SNR versus shrinkage threshold. (b) SNR versus subset selection threshold. The marker points represent the achieved SNRs using shrinkage thresholds chosen by cross validation.



(a) grass (b) water

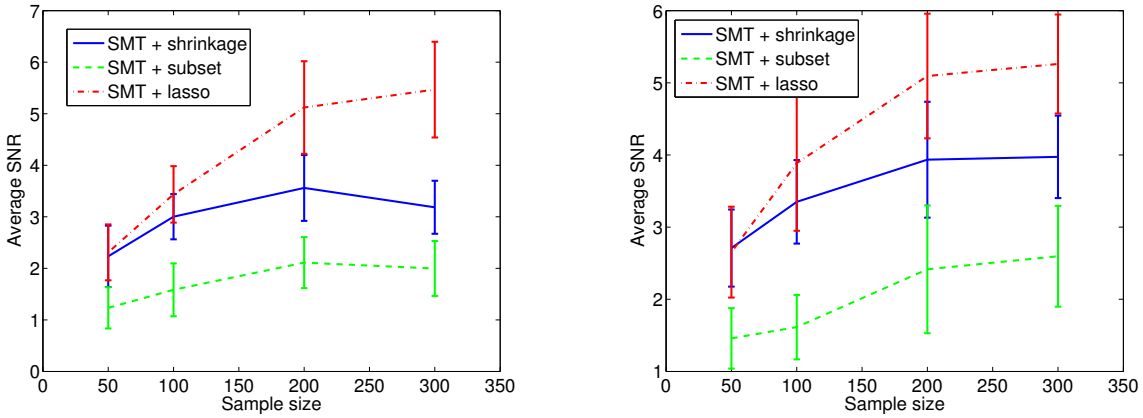
Fig. 5.4. Plots of average SNRs using different regression methods. Notice that SMT-Lasso regression results in highest SNRs, especially in the range of $n < p$ compared to the other regression methods. (a) Clutter W is generated using hyperspectral grass data. (b) Clutter W is generated using hyperspectral water data.

5.6 Conclusions

In this paper, we proposed a novel regression approach for high dimensional data. In this approach, the SMT covariance estimate is computed and used to decorrelate the data, and then different model selection methods are used to obtain good estimates of regression coefficients in the decorrelated data domain. Numerical examples show that the proposed approach can significantly improve the SNR of the regression models, especially for 'small n, large p' problems.

5.7 Alternative Regression Methods

This section lists out some alternative methods for regression. We will compare our proposed regression approaches with these methods in the numerical simulations.



(a) grass (b) water

Fig. 5.5. SMT-Lasso versus SMT-Shrinkage versus SMT-Subset. SMT-Lasso works best, but are much more computationally expensive. (a) Clutter W is generated using hyperspectral grass data. (b) Clutter W is generated using hyperspectral water data.

5.7.1 Ordinary Least Squares Regression

In practice we do not usually know the true covariance and the correlation coefficient, therefore, the sample covariance $S = \frac{1}{n}X^tX$ and sample correlation $\frac{X^ty}{n}$ are generally used to estimate the regression coefficient vector b, which is called ordinary least squares (OLS) regression. OLS can be formulated as the unconstrained optimization problem

$$\hat{b} = \arg\min_b \parallel y - Xb \parallel^2 , \tag{5.14}$$

and the solution is given by

$$\hat{b} = (X^tX)^{-1}\left(X^ty\right) . \tag{5.15}$$

We know the OLS estimator is unbiased, and it has lowest variance among all linear unbiased estimators [57]. However, its variance can be very high for high-dimensional data, compared to regularized estimators that are usually biased. Especially, if $p > n$, the sample covariance is singular and non-invertible.

5.7.2 Ridge Regression

Ridge regression imposes a L_2 penalty on the magnitude of the regression coefficients. It can be formulated as the constrained optimization problem

$$\hat{b} = \arg\min_b \left\{ \parallel y - Xb \parallel^2 + \lambda \parallel b \parallel^2 \right\} . \tag{5.16}$$

5.7.3 Traditional Lasso Regression

Traditional Lasso regression is usually performed in the native coordinates of X with L_1 regularization on the regression coefficients [58]

$$\hat{b} = \arg\min_b \left\{ \parallel y - Xb \parallel^2 + \lambda \parallel b \parallel_1 \right\} . \tag{5.17}$$

REFERENCES

REFERENCES

[1] "JBIG2 Final Draft International Standard," *ISO/IEC JTC1/SC29/WG1N1545*, Dec. 1999.

[2] R. B. Arps and T. K. Truong, "Comparison of international standards for lossless still image compression," in *Proc. of IEEE*, vol. 82, 1994, pp. 889–899.

[3] O. Fumitaka, R. William, A. Ronald, and C. Corneliu, "JBIG2 - the ultimate bi-level image coding standard," in *Proc. of IEEE Int'l Conf. on Image Proc.*, 2000.

[4] P. G. Howard, F. Kossentini, B. Martins, S. Forchhammer, and W. J. Rucklidge, "The emerging JBIG2 standard," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 8, pp. 838–848, 1998.

[5] A. Ortego and K. Ramchandran, "Rate-distortion methods for image and video compression," *IEEE Signal Proc. Magazine*, vol. 15, pp. 23–50, 1998.

[6] M. J. Ruf and J. W. Modestino, "Operational rate-distortion performance for joint source and channel coding of images," *IEEE Trans. on Image Processing*, vol. 8, pp. 305–320, 1999.

[7] M. J. J. Holt, "A fast binary template matching algorithm for document image data cmpression," in *Proc. of IEEE Int'l Conf. on Pattern Recognition*, 1988, pp. 230–239.

[8] W. Pratt, P. Capitant, W.-H. Chen, E. Hamilton, and R. Wallis, "Combined symbol matching facsimile data compression system," in *Proc. of the IEEE*, 1980, pp. 786–796.

[9] M. Figuera, C. A. Bouman, and J. Yi, "A new approach to JBIG2 binary image compression," in *Proc. of SPIE*, vol. 6493, 2007, p. 649305.

[10] Q. Zhang and J. M. Danskin, "Entropy-based pattern matching for document image compression," in *Proc. of IEEE Int'l Conf. on Image Proc.*, vol. 2, 1996, pp. 221–224.

[11] Q. Zhang, J. M. Danskin, and N. E. Young, "A codebook generation algorithm for document image compression," in *1997 IEEE Data Compression Conf.(DCC)*, vol. 2, 1997, pp. 300–309.

[12] Y. Ye and P. C. Cosman, "Dictionary design for text image compression with JBIG2," *IEEE Trans. on Image Processing*, pp. 818–828, 2001.

[13] Y. Ye, D. Schilling, P. C. Cosman, and H. H. Ko, "Symbol dictionary design for the JBIG2 standard." in *Data Compression Conference'00*, 2000, pp. 33–42.

[14] Y. Ye and P. C. Cosman, "Fast and memory efficient text image compression with JBIG2," *IEEE Trans. on Image Processing*, pp. 944–956, 2003.

[15] M. Elad, R. Goldenberg, and R. Kimmel, "Low bit-rate compression of facial images," *IEEE Trans. on Image Processing*, vol. 16, pp. 2379–2383, 2010.

[16] O. Bryt and M. Elad, "Compression of facial images using the k-svd algorithm," *J. Vis. Comun. Image Represent.*, vol. 19, no. 4, pp. 270–282, May 2008. [Online]. Available: http://dx.doi.org/10.1016/j.jvcir.2008.03.001

[17] G. Qiu, X. Gu, Z. Chen, Q. Chen, and C. Wang, "An information theoretic model of spatiotemporal visual saliency," in *ICME*, 2007, pp. 1806–1809.

[18] Y. Guo, X. Gu, Z. Chen, Q. Chen, and C. Wang, "Adaptive video presentation for small display while maximize visual information," in *Advances in Visual Information Systems*. Springer, 2007, pp. 322–332.

[19] Q. Qiu, V. M. Patel, and R. Chellappa, "Information-theoretic dictionary learning for image classification," *CoRR*, vol. abs/1208.3687, 2012.

[20] K. Skretting and K. Engan, "Image compression using learned dictionaries by rls-dla and compared with k-svd." in *ICASSP*. IEEE, pp. 1517–1520.

[21] Y. Guo, D. Depalov, P. Bauer, B. Bradburn, J. P. Allebach, and C. A. Bouman, "Binary image compression using conditional entropy-based dictionary design and indexing," in *Proc. SPIE 8652, Color Imaging XIII: Displaying, Processing, Hardcopy, and Applications*, vol. 8652, 2013, p. 865208.

[22] P. Schmid-Saugeon and A. Zakhor, "Dictionary design for matching pursuit and application to motion-compensated video coding." *IEEE Trans. Circuits Syst. Video Techn.*, vol. 14, no. 6, pp. 880–886, 2004.

[23] M. Elad and D. Datsenko, "Example-based regularization deployed to super-resolution reconstruction of a single image," *Comput. J.*, vol. 52, no. 1, pp. 15–30, Jan. 2009. [Online]. Available: http://dx.doi.org/10.1093/comjnl/bxm008

[24] P. Jin, E. Haneda, and C. Bouman, "Implicit Gibbs prior models for tomographic reconstruction," in *the 46th Asilomar Conference on Signals, Systems and Computers*, Nov. 2012.

[25] R. Rubinstein, A. M. Bruckstein, and M. Elad, "Dictionaries for sparse representation modeling," *Proc. of the IEEE*, vol. 98, pp. 1045–1057, 2010.

[26] G. Parisi, *Statistical Field Theory*. Addison-Wesley, 1988.

[27] T. P. Minka, "A family of algorithms for approximate bayesian inference," Ph.D. dissertation, 2001, aAI0803033.

[28] Y. Qi and Y. Guo, "Message passing with l_1 penalized kl minimization," in *Proceedings of the* 30*th International Conference on Machine Learning, and Journal of MLR*, vol. 28, Atlanta, Georgia, USA, 2013.

[29] "Standardization of Group 3 Facsimile Apparatus for Document Transmission," *CCITT Recommend. T.4*, 1980.

[30] "Facsimile Coding Schemes and Coding Control Functions for Group 4 Facsimile Apparatus," *CCITT Recommend. T.6*, 1984.

[31] "Progressive Bi-level Image Compression," *CCITT Recommend. T.82*, 1993.

[32] R. Hunter and H. Robinson, "International digital facsimile coding standards," *Proc. of the IEEE*, vol. 68, pp. 854–867, July 1980.

[33] C. Constantinescu and R. Arps, "Fast residue coding for lossless textual image compression," in *1997 IEEE Data Compression Conf.(DCC)*, ser. DCC '97, 1997, pp. 397–406. [Online]. Available: http://dl.acm.org/citation.cfm?id=789085.789584

[34] P. G. Howard;, "Lossless and lossy compression of text images by soft pattern matching," in *1996 IEEE Data Compression Conf.(DCC)*, 1996, pp. 210–219.

[35] "Application Profiles for Recommendation T.88: – Lossy/lossless Coding of Bi-level Images (JBIG2) for Facsimile," *ITU-T recommendation T.89*, 2001.

[36] Y. Ye and P. C. Cosman, "Fast and memory efficient JBIG2 encoder," in *Proc. of IEEE Int'l Conf. on Acoust., Speech and Sig. Proc.*, vol. 3, 2001, pp. 1753–1756.

[37] M. Figuera, "Memory-efficient algorithms for raster document image compression," Ph.D. dissertation, Purdue University, West Lafayette, IN, USA, 2008.

[38] Y. Guo, D. Depalov, P. Bauer, B. Bradburn, J. P. Allebach, and C. A. Bouman, "Dynamic hierarchical dictionary design for multi-page binary document image compression," in *Proc. of IEEE Int'l Conf. on Image Proc.*, 2013, pp. 2294 – 2298. [Online]. Available: + http://dx.doi.org/10.1117/12.711693

[39] G. Cao, Y. Guo, and C. A. Bouman, "High dimensional regression using the sparse matrix transform (SMT)," in *Proc. of IEEE Int'l Conf. on Acoust., Speech and Sig. Proc.*, 2010, pp. 1870–1873.

[40] S. J. Inglis, "Lossless document image compression," Ph.D. dissertation, University of Waikato, New Zealand, 1999.

[41] H. Chen, S. S. Tsai, G. Schroth, D. M. Chen, R. Grzeszczuk, and B. Girod, "Robust text detection in natural images with edge-enhanced maximally stable extremal regions," in *Proc. of IEEE Int'l Conf. on Image Proc.*, 2011.

[42] C. Lu, J. Wagner, B. Pitta, D. Larson, and J. Allebach, "SVM-based automatic scanned image classification with quick decision capability," in *Proc. SPIE 9015, Color Imaging XIX: Displaying, Processing, Hardcopy, and Applications.* to be published, 2014.

[43] L. Fletcher and R. Kasturi, "A robust algorithm for text string separation from mixed text/graphics images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 10, no. 6, pp. 910–918, Nov. 1988.

[44] A. H. Laurence Likforman-Sulem and C. Faure, "A hough based algorithm for extracting text lines in handwritten documents," in *ICDAR*. Washington, DC, USA: IEEE Computer Society, 1995, pp. 774–777. [Online]. Available: http://dl.acm.org/citation.cfm?id=839278.840242

[45] Y. Pu and Z. Shi, "A natural learning algorithm based on Hough transform for text lines extraction in handwritten documents," in *Proc. of the 6 Int'l Workshop on Frontiers in Handwriting Recognition*, 1998, pp. 637–646.

[46] Z. Shi and V. Govindaraju, "Line separation for complex document images using fuzzy runlength," in *DIAL*. IEEE Computer Society, 2004, pp. 306–313.

[47] F. L. Bourgeois, H. Emptoz, E. Trinh, and J. Duong, "Networking digital document images," in *ICDAR*. IEEE Computer Society, 2001, pp. 379–383.

[48] G. Louloudisa, B. Gatosb, I. Pratikakisb, and C. Halatsisa, "Text line detection in handwritten documents," *Pattern Recognition*, vol. 41, pp. 3758–3772, 2008.

[49] G. Louloudis, B. Gatos, and C. Halatsis, "Text line detection in unconstrained handwritten documents using a block-based hough transform approach," in *ICDAR*, vol. 2, Los Alamitos, CA, USA, 2007, pp. 599–603.

[50] Y. Li, Y. Zheng, and D. Doermann, "Detecting text lines in handwritten documents," in *Proc. of IEEE Int'l Conf. on Pattern Recognition*, vol. 2, Los Alamitos, CA, USA, 2006, pp. 1030–1033.

[51] C. Yi and Y. Tian, "Text string detection from natural scenes by structure-based partition and grouping," *IEEE Trans. on Image Processing*, vol. 20, no. 9, pp. 2594–2605, 2011.

[52] R. Grzeszczuk, V. Chandrasekhar, G. Takacs, and B. Girod, "Method and apparatus for representing and identifying feature descriptors utilizing a compressed histogram of gradients," May 20 2010, wO Patent App. PCT/IB2009/007,434. [Online]. Available: https://www.google.com/patents/WO2010055399A1?cl=en

[53] E. Haneda and C. A. Bouman, "Text segmentation for mrc document compression," *IEEE Trans. on Image Processing*, vol. 20, no. 6, pp. 1611–1626, 2011.

[54] G. Cao and C. Bouman, "Covariance estimation for high dimensional data vectors using the sparse matrix transform," in *Advances in Neural Information Processing Systems*. MIT Press, 2008, pp. 225–232.

[55] B. Frey and D. MacKay, "A revolution: Belief propagation in graphs with cycles," in *Advances In Neural Information Processing Systems*, 1998, pp. 479–485.

[56] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. The MIT Press, 2001.

[57] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer, 2001.

[58] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B*, vol. 58, no. 1, pp. 267–288, 1996.

[59] X. J. Jeng, Z. J. Daye, and Y. Zhu, "Sparse covariance thresholding for high-dimensional variable selection with the lasso," Purdue University, Statistics Department, Technical Report TR 08-07, 2008.

[60] D. M. Witten and R. Tibshirani, "Covariance-regularized regression and classification for high dimensional problems," *Journal of the Royal Statistical Society: Series B*, vol. 71, no. 3, pp. 615–636, 2009.

[61] G. Cao, C. A. Bouman, and J. Theiler, "Weak signal detection in hyperspectral imagery using sparse matrix transformation (SMT) covariance estimation." Grenoble, France: First Workshop on Hyperspectral Image and Signal Processing (WHISPERS), August 2009.

[62] P. J. Bickel and E. Levina, "Regularized estimation of large covariance matrices," *Annals of Statistics*, vol. 36, no. 1, pp. 199–227, 2008.

[63] S. Alliney and S. Ruzinsky, "An algorithm for the minimization of mixed l(1) and l(2) norms with application to bayesian estimation," *IEEE Transactions on Signal Processing*, vol. 42, no. 3, pp. 618–627, 1994.

[64] D. A. Landgrebe, *Signal Theory Methods in Multispectral Remote Sensing.* New York: Wiley-Interscience, 2005.

VITA

VITA

Yandong Guo received his B.S. and M.S. degree in electrical and engineering from Beijing University of Posts and Telecommunications, China, in 2005 and 2008, receptively. Since January 2009, he has been pursuing his Ph.D. in electrical engineering at Purdue University. He joined Microsoft Corporation as an associated researcher in January 2014. He is a member of the Institute of Electrical and Electronics Engineers (IEEE) and a member of the international society for optics and photonics (SPIE). His research focuses on statistical image models, inverse problems, machine learning, and computer vision, especially image compression, reconstruction, understanding, and search.