2015

Information Preserving Processing of Noisy Handwritten Document Images

Jin Chen
Lehigh University

Information Preserving Processing of Noisy Handwritten Document Images

by

Jin Chen

A Dissertation
Presented to the Graduate Committee
of Lehigh University
in Candidacy for the Degree of
Doctor of Philosophy
in
Computer Science

Lehigh University
May 2015

Approved and recommended for acceptance as a dissertation in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Jin Chen

Date

Accepted Date

Daniel Lopresti, Dissertation Director, Chair
(Must Sign with Blue Ink)

Committee Members

Daniel Lopresti (Chair)

George Nagy (RPI)

Xiaolei Huang

Brian Davison

Huaigu Cao (Raytheon BBN)

Acknowledgments

Graduate school can be stressful. This dissertation would have been impossible if not my advisor, colleagues, friends and family throughout the years.

First of all, I am most indebted to my advisor, Professor Dan Lopresti, who has been what a Ph.D student can possibly expect to work with: knowledgeable, dedicated, responsible, and strict. I will never forget his dedication of substantial time and effort to instruct me, iteratively, to become a qualified researcher in critical thinking, scientific writing, academic presentation, time management and pedagogical methodology. This type of fundamental cultivation and training will reward me in my whole career. It's simply a privilege to work with and learn from him throughout these years.

Second, I am grateful to admirable Professor George Nagy, who is a true pioneer in the field of document image analysis and remains enthusiastic even if he is retired. Every meet-up with him has been a building block in my research and our friendship, which I will cherish forever. I wish him to continuously enjoy the fun of ski and the peace of his retirement.

I received tremendous amount of help from the committee members, Professor Xiaolei Huang, Professor Brian Davison, and Dr. Huaigu Cao, for their invaluable time in guiding me to accomplish a qualified dissertation. Professor Huang shared her insights on research with me from the very beginning of my Ph.D. Professor Davison's list of advices for graduate school success has been a constant consulting resource to me. Dr. Cao's expertise and guidance during my internships significantly expanded my horizon of conducting solid research.

I am grateful to the colleagues I had the privilege to work with: Dr. Prem

Natarajan, Dr. David Doermann, Dr. Xujun Peng, Dr. Fabian Monrose, Dr. Lucas Ballard, Dr. Ergina Kavallieratou and Dr. Bart Lamiroy.

I enjoyed Professor Henry Baird's stimulating lectures and numerous discussions with him in classrooms, offices, seminars and conferences. I was also benefitted from my lab mates at Lehigh, Pingping Xiu, Sui-yu Wang, Chang An, Mike Moll, Wen Cheng, Dawei Yin, and Tao Sun, who offered all sorts of help in my research and life. My student life would've been a lot harder if not Bryan Hodgson's constant technical assistance and the sorts of anecdotes in the history of computer industry.

Moreover, thanks to my dearest friends, who have always been unconditionally supportive when I was stressed out and down: Shuai Yuan, Chuanming Wei, Chen Chen, Miao Zhang, Peize Han, Qian Wu, Qian He, Zikang Wu, Le Guang and many others. Thanks to Sabrina for her truthful support and care in the early years.

Special thanks to Qingyu, who has provided profound encouragement to help me sustain the positive outlook that is necessary to overcome the difficulties of completing the Ph.D.

I appreciate the financial support from NSF and Raytheon BBN Technologies for the research assistantship during my graduate school. Also, I enjoyed all the three summer internships offered by BBN and the friendship with Bing, Yingbo, Le, Rex, Jayant, Xiaodan and Tim since then.

Lastly, I give the most gratitude to my parents. If there is any achievement that I've made in my career or personal life, it is the direct result of their unwavering love, cultivation, and trust.

I feel I am a fortunate man.

Dedication

This dissertation is dedicated to my parents
and those years in my 20s.

Contents

List of Tables

List of Figures

Abstract

Many pre-processing techniques that normalize artifacts and clean noise induce anomalies due to discretization of the document image. Important information that could be used at later stages may be lost. A proposed composite-model framework takes into account pre-printed information, user-added data, and digitization characteristics. Its benefits are demonstrated by experiments with statistically significant results. Separating pre-printed ruling lines from user-added handwriting shows how ruling lines impact people's handwriting and how they can be exploited for identifying writers. Ruling line detection based on multi-line linear regression reduces the mean error of counting them from 0.10 to 0.03, 6.70 to 0.06, and 0.13 to 0.02, compared to an HMM-based approach on three standard test datasets, thereby reducing human correction time by 50%, 83%, and 72% on average. On 61 page images from 16 rule-form templates, the precision and recall of form cell recognition are increased by 2.7% and 3.7%, compared to a cross-matrix approach. Compensating for and exploiting ruling lines during feature extraction rather than pre-processing raises the writer identification accuracy from 61.2% to 67.7% on a 61-writer noisy Arabic dataset. Similarly, counteracting page-wise skew by subtracting it or transforming contours in a continuous coordinate system during feature extraction improves the writer identification accuracy. An implementation study of contour-hinge features reveals that utilizing the full probabilistic probability distribution function matrix improves the writer identification accuracy from 74.9% to 79.5%.

Chapter 1

Introduction

Since its invention back in the second century BC in China, paper has been an important tool to record history and to foster civilization.[112] Although there are multiple means of information transmission today, paper documents are still among the most popular ones.

"A document is a material substance (as a coin or stone) having on it a representation of thoughts by means of conventional mark or symbol." – Merriam-Webster.[2] Such conventional marks or symbols refer to specific script glyphs and line arts, and may also present in different formats and styles, such as **bold** vs. *italic*, block vs. cursive handwriting, etc.

1.1 A Brief History of Document Image Analysis

Document image analysis (DIA) is the task of converting document images to a symbolic form for modification, storage, retrieval, reuse, and transmission.[175] It is a subfield of pattern recognition and computer vision where a number of engineering

techniques related to document images are proven to be successful in practice.

Historically, DIA dates back to before the invention of digital computers.[99] Early optical character recognition (OCR) was intended to expand telegraphy and create reading devices for the blind. In 1913, the *optophone*, used by the blind, was invented by Dr. Edmund Fournier d'Albe who used selenium photosensors to detect printed characters and converted them into audible output that the blind can interpret.[99]

Although DIA has been in use for decades especially in the banking business where numeric check numbers are read by computers, it is in the late 1980s that the DIA area has grown rapidly.[99] Hardware advancement enables processing of scanned document images at a more reasonable speed and cost. Nowadays, a typical personal computer is well capable of handling images of business letters scanned at 600 DPI, which is $5100w \times 6600h$.[58]

In the 1950s, researchers treated OCR as a task of pattern classification, thus there was a large amount of effort on hand-crafting features and exploiting machine learning techniques, given carefully prepared input data. In addition, due to limited computational resources, researchers focused on clean black/white isolated class images in the early days. One example is a handwritten digit dataset called the Modified National Institute of Standards and Technology (MNIST[141]), which was introduced in 1998 as a subset of the original NIST dataset.[4] Over the years, researchers have successfully pushed the error rates down to 0.23%, using large-scale Convolutional Neural Networks (CNNs).[65]

DIA, however, usually faces the input of scanned document images rather than clean and isolated characters or digits. For example, various document components

such as text lines, line art, and graphs may present within a document page. In addition, multiple scanning artifacts may be introduced during scanning as well, such as binarization, clutter noise, page rotation, page warping, etc. One known example is Google's book search project which aims to digitize books from all the world's scripts and languages discoverable via Internet.[241] Although OCR on machine-printed individual characters is considered a solved problem,[175] there are a number of challenges when dealing with books from such diverse origins.[241] For example, language/script identification is usually carried out so that a language specific OCR engine can be selected for higher OCR accuracy.[224,231] In addition, it is not a straightforward task of automatically deskewing, segmenting, and cleaning up scanned book pages, given the fact that historical books have ink/pigment properties and printing styles different from those in newly printed ones.

In general, documents can be divided into two major categories: text-mostly and graphics-mostly.[175] Figure 1.1 shows several examples of document images that DIA researchers have been working on. Text-mostly examples include business documents, forms, handwritten notes, as shown in the first row of Figure 1.1. Graphics-mostly examples include music scores, engineering drawings, maps, as shown in the second row of Figure 1.1. In DIA, document characteristics are so distinct that no techniques have, or perhaps will ever, claim to solve arbitrary document images.[30]

1.2 Workflow of Document Image Analysis

A typical workflow of document image analysis is shown in Figure 1.2. This diagram shows a perspective from computer vision and image processing when dealing with

Figure 1.1: Sample documents of several document categories: envelopes, forms, music scores, and engineering drawings.

document images. For example, pre-processing is used to help clean and/or normalize input images, followed by black/white thresholding. Then, document component detection and segmentation are conducted during layout analysis, followed by text recognition, non-text (e.g., tables, forms, graphs, drawings) interpretation, and document understanding. In general, there are three major strategies for layout analysis: *top-down* approaches that attempt to use white space for segmenting document images into homogeneous regions, *bottom-up* ones that recursively group homogeneous document primitives from basic components or foreground pixels, and

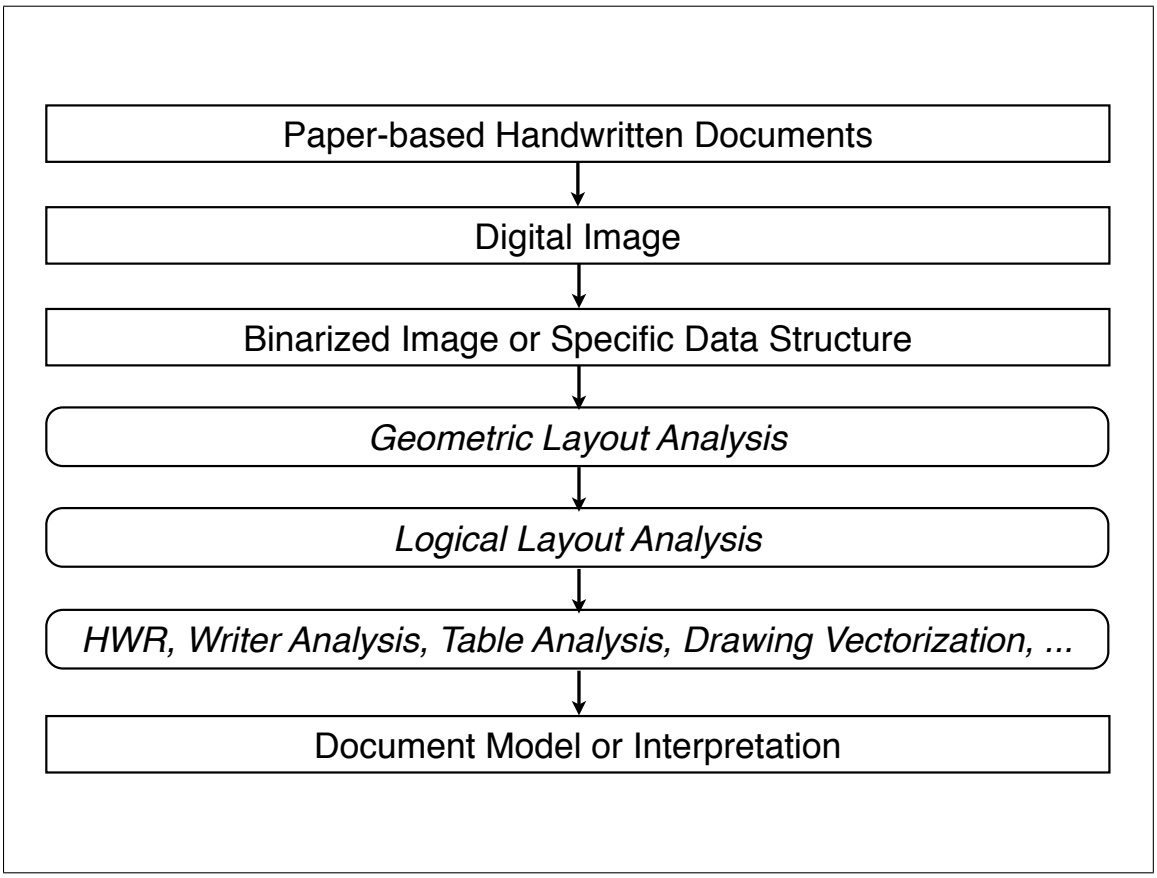


Figure 1.2: A diagram showing the workflow of handwritten document image analysis. Squared rectangles represent data items and rounded ones represent processing modules.

hybrid ones that combine the two.[117]

DIA covers a wide range of documents such as technical articles,[179] business letters and faxes,[69] tables/forms,[109,260] postal addresses,[92,262] musical scores,[67] maps,[259] and engineering drawings.[76,77,173] The underlying techniques include and are not restricted to printed character recognition, hand-printed character recognition, language/script identification, font identification, musical score recognition, map interpretation, engineering drawing vectorization and interpretation, graphic drawing

interpretation, and table/form recognition and interpretation.

In many of the attempts to present a general method for handling technical document images, Jain and Yu define a framework which includes *geometric* layout analysis, *logical* layout analysis, and application-specific processing.[117] Geometric layout analysis specifies the physical structure of the maximal *homogeneous* regions and the spatial relations of these regions.[96] A region is homogeneous if its area is of one type: text, table, figure, drawing, etc. In geometric layout analysis, page segmentation or decomposition has been a central task in the DIA area since OCR relies on a homogeneous textual region as input.

On the other hand, logical layout analysis determines the type of the page, assigns functional labels (title, logo, footnote, caption, signature, table, figure, etc.) to each zone of the page, and organizes text blocks according to their reading order. Common DIA applications include OCR, table understanding, drawing vectorization, and image compression.[117]

1.3 Motivation

Traditional document image analysis consists of a pipeline of processing stages where each stage makes assumptions about the nature of the input image. These assumptions amount to pre-conditions that must be satisfied for the procedure in question to function as expected. For example, a layout analysis module (e.g., the "X-Y" cut) may assume that the input page image contains no page skew. In general, these pre-conditions are satisfied by providing a preceding module that detects violations

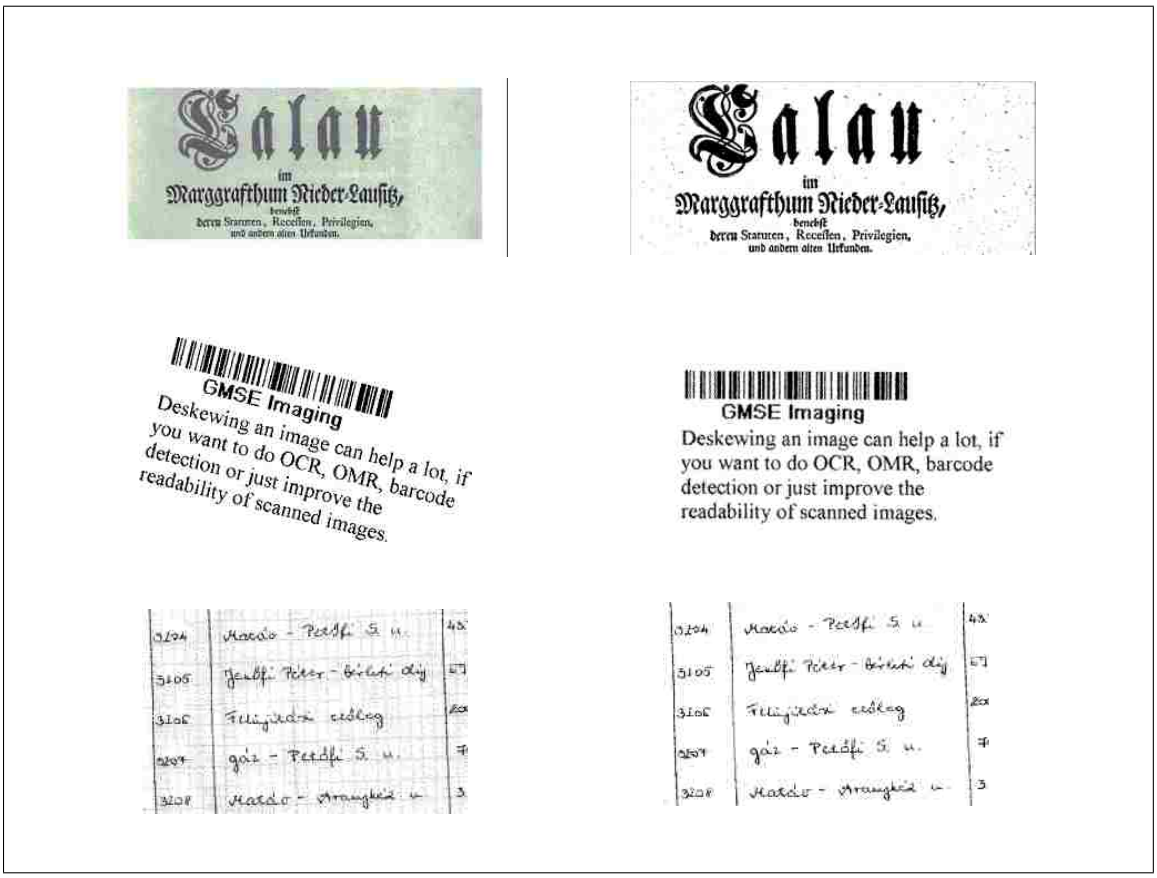


Figure 1.3: Several examples of pre-processing in document image analysis. Left-hand side are the original images and right-hand side are the processed ones.

of the assumption and corrects for it. For example, rotating the image bitmap is usually applied to counteract the presence of page skew due to the page being scanned at an angle. As a result, the original image is modified in an irreversible way as it is passed along the document processing pipeline. While this type of normalization may make it easier to design a particular module (e.g., text line segmentation is simpler if the lines are assumed to be horizontal and mostly parallel), important information that could be useful at later stages may be lost along the way.

Common pre-processing techniques include binarization, dewarping, deskewing,

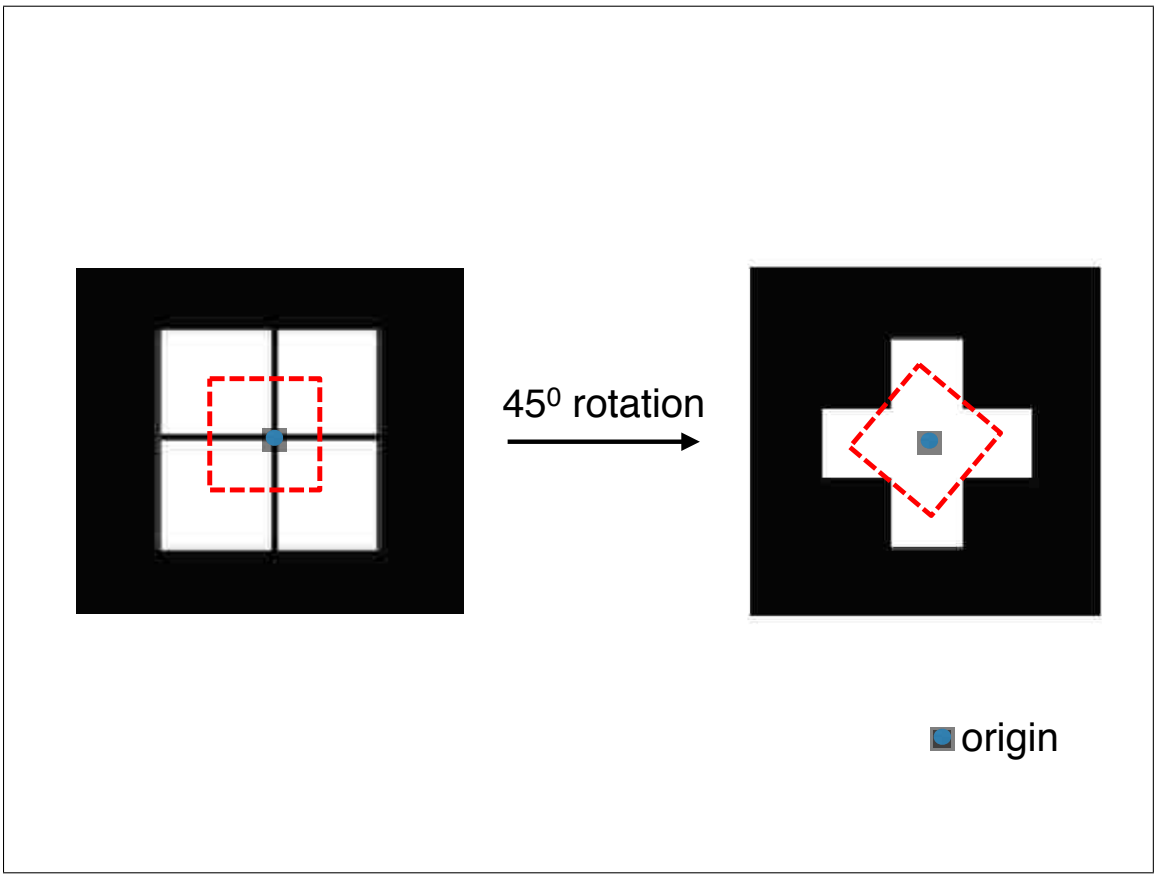


Figure 1.4: A 2×2 digital square is rotated by $45°$ and the result digital square is unexpected. Figure redrawn from the example in.[176]

and noise/artifact removal. DIA specific processing techniques include geometric layout analysis, text line segmentation, word/character segmentation and slant correction, etc. These types of image cleaning are usually termed as *pre-processing*, as exemplified in Figure 1.3. The first row of this figure shows binarization which converts color or gray-level images into black/white ones. The second row represents page deskewing which rotates the page against the page skew so that the result document image is upright. The third row shows noise removal which excludes scanning noise and/or other artifacts.

Although pre-processing largely reduces the variations of input images for feature extraction, it modifies the original image. Thus, what commonly happens is that useful information might be discarded during pre-processing.

To see how pre-processing may modify a bitmap, let us consider a typical image processing: rotation. Image rotation is a linear transform in image processing which has a transform matrix as follows:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{1.1}$$

In this equation, (x', y') and (x, y) are coordinates of one pixel before and after the image rotation transformation.

Although this transformation is mathematically sound, the issue of quantization arises when implementing the algorithm on a digital grid.[176] The most dramatic example is to rotate a square by 45°, as shown in Figure 1.4. The result image becomes a cross shape after the rotation, because rotated pixels at the corners are resampled onto a grid. Consequently, several image features might be computed wrong, e.g., moment features.

Originally, the moment feature $M_{i,j}$ is computed as:

$$\sum_i \sum_j x^i y^j I(x, y) \tag{1.2}$$

Suppose the center of the top-right square in the 2×2 tuple is (1,1) from the origin. After rotation by 45°, this distance becomes $\sqrt{2}$. Now if we look at $M_{2,0}$, for the original image we have $\sum\sum(1-0)^2 = 4$, while in the result image this

feature value becomes $\sum\sum(\sqrt{2}-0)^2$=8. As Nagy points out, not only rotation but also non-linear shape normalization,[240] mapping pen coordinates for feature extraction,[248] and document defect model[31] are usually implemented in resampling and thus error-prone.[176] Nagy also advocates normalizing image features using exact and reversible transformed coordinates rather than extracting features from resampled bitmaps. His idea partially motivates our work in this dissertation.

Another motivation of my thesis work is that the hierarchical document structures proposed in the literature seem to be over-simplified in the sense that they only represent in a tree structure without drawing connections between layers. For example, geometric layout modeling attempts to separate different components from a document image into groups or layers like noise/artifacts, machine-printed text, handwriting, etc., without drawing connections between these components. It is also worth noting that pre-printed information usually indicates document source characteristics, which are potentially useful for document image analysis and knowledge extraction at the corpus level. Thus, it may be worth a second thought when dealing with them during pre-processing, and as shown in Chapter 5, we may be benefited to exploit such interaction between document components.

1.4 An Evaluation Example

We illustrate by evaluating a ruling line removal algorithm in writer identification. Details are present in our previous work,[57] but the major points are echoed here for the sake of completeness.

Pre-printed ruling lines are designed to help people write neatly but become artifacts after scanning for document image analysis. For human perception, ruling lines are usually not a problem. For automatic document image analysis, however, they introduce challenges in extracting discriminative features for tasks like handwriting recognition and writer identification. Much of previous work aims to remove these ruling lines while having minimum impact on handwriting.[5,21,44]

1.4.1 The Approach

One traditional way of handling ruling lines is to detect and remove them first, and then to recover the possible broken handwritten strokes. To detect and remove ruling lines, the horizontal projection profiles (HPPs) are computed and the page skew is determined by finding the minimal entropy in the HPPs. Next, the position of the ruling lines are detected by finding the peak in the corresponding HPP. Finally, the line thickness is estimated by computing the histogram of vertical run-lengths,[128] and the bin with top vote is selected as the estimation.

The next step is to recover broken handwritten strokes after removing ruling lines. Following the strategy from Cao et al.,[46] there are three sub-steps: broken stroke reconnection, thinned stroke recovery, and "U-shape" pattern detection and stroke regeneration. Broken strokes are recognized by computing the distances between segments above the ruling line and those below them. Thinned strokes are handled by drawing extra ink pixels column by column in the direction of the ruling line. "U-shape" segments are connected by drawing an artificial straight line at the middle part of two segments and partial ellipses at the ends to make the artificial strokes look natural.

Table 1.1: Datasets used in our experiments.

Dataset	Sample Size (text lines)		
	Training	Testing	Total
Ruling-line-only (RLO)	2,700	900	3,600
Ruling-line-free (RLF)	20,700	6,900	27,600
Mixed (M)	3,600	1,200	4,800

1.4.2 Experimental Evaluation

The Arabic dataset for evaluation is provided by the Linguistic Data Consortium (LDC).[1] Sixty native Arabic writers contributed samples of handwriting on paper sheets. To avoid biased sampling, we split each writer's handwritten lines into four disjoint subsets to conduct four-fold cross-validation. For each fold, the data was equally divided into four subsets, each of which in turn served as a testing set and the remaining three as a training set. The results were then reported as the average accuracy of all four folds. A breakdown of all three datasets is shown in Table 1.1.

We used Support Vector Machines (SVMs) on different dataset conditions: ruling-line-only (RLO), ruling-line-free (RLF), and mixed (M) datasets. The results are summarized in a 3×3 matrix shown in Table 1.2. In this table, bold figures indicate statistically significant improvements when ruling lines are removed (see Section 1.7.) For convenience in the following discussion, we use the notation $\mathcal{E}(train/test)$ to represent an experiment that trains on some dataset and tests on another (or the same) dataset.

Looking at Table 1.2, it is clear that removing ruling lines for writer identification never improved the accuracy of writer identification. This observation motivated us

Table 1.2: Writer identification accuracy after applying a ruling line removal algorithm. The figures below are the mean of 4-fold cross validation.

	Training/Testing Subsets		
	RLO	RLF	M
Before Removal	62.5%	74.7%	62.0%
After Removal	58.0%	74.7%	61.0%

to investigate the traditional methodology of handling pre-printed ruling lines and to further propose a hopefully better approach of handling such artifacts.

1.5 Research Objectives

We propose a composite-model framework for analyzing semi-structured handwritten document images that consists of three major sources, namely *pre-printed information*, *user added data*, and *digitization characteristics*. In fact, these three sources can be viewed in the order they are added into a document image.

Pre-printed information includes anything that belongs to the document template during printing, such as business logos, header/footnote text, machine-printed text, pre-printed rulings, pre-printed tables/forms, etc. As well as these primitives, it includes all types of meta-data and derived knowledge of this corresponding pre-printed information, such as the underlying machine-printed text model that decides the font size and styles, the pre-printed ruling model that controls the position and inter-ruling spacing, and the table/forms model that governs row/column/cell specifications and their logical relations.

User-added data includes information that is added by human beings, via a

writing instrument or manipulation of the paper. Typically, the first part refers to handwriting, signatures, annotations/correction, hand-drawn sketches, etc. Moreover, the authorship model of user-added data may govern modalities ranging from handwriting, signatures to hand-drawn sketches, which usually serve as a fundamental hint for individuality, and is widely used in writer identification or verification in forensics and biometric security applications.[52]

In addition, although not a focus of this dissertation, physical manipulation of the document page is part of user-added data as well, e.g., folding, tearing, aging, and stapling. Folding and tearing documents may add, discard, and modify information contained in the document before scanning. Aging is common for historical documents where the original ink and paper materials might degrade substantially, thus different processing techniques might be required. Another common degradation of document images, usually referred as *bleed-through*, is that text/ink on the verso of the page comes through the recto. This effect reduces readability of the document and thus requires proper handling. Stapling is one common way to organize documents in office. Thus, stapling marks provide clues of clustering individual documents in a corpus, which make it useful for DIA at the corpus level.[208]

In this dissertation, the term *digitization characteristics* includes several aspects of converting a physical document page into an image file on computer. First, the digital array generated from a given document will be different each time it is scanned, even on the same scanner.[155,252] For example, the paper can be placed in a different position relative to the platen which causes translation, skew, missing corners or edges. Even if the paper is not moved, its relative displacement to the scanning grid constantly changes because of the inexact mechanical motion of the

scanning head, giving rise to random phase noise.[220] Furthermore, both the illumination and the CCD or CMOS sensor sensitivity change with aging and with source voltage fluctuations.

Second, digitization requires spatial sampling and amplitude quantization, thus an approximation.[207] As a result, the amplitude value of an image pixel does not necessarily correspond to the reflectance of the document at the pixel coordinates because of scanner limitations or defects. For example, line thickness cannot be accurately estimated from a bitonal setting because thin lines will be broken in places while complete in others. Even when properly calibrated, different scanners will give different pixel maps because some operating characteristics such as point spread function (PSF) cannot be adjusted.

Third, we would like to incorporate scanning settings such as color mode, brightness, contrast, and resolution because these settings decide the input for document image analysis. For example, scanning resolution decides the size of the result document image and its connected components. In general, high resolutions capture more precise information. However, they require more expensive equipment and more computational resources to process. For example, 200-600 DPI are commonly seen in current public datasets in the DIA area.[58] In addition, it is not rare for a document page to contain color information, such as a colorful business title or logos. During scanning in DIA, however, it is a common practice to scan using the black/white mode which eliminates such useful information for document analysis and knowledge extraction. Other situations include handwritten annotations that are usually present in a different color than the other text, which are common in notebook pages and manuscripts.[58]

We should clarify that there can be iterations or different orders of compositing these three sources. For example, in a printed form we could consider a composite model of 1. pre-printed information, 2. user-added handwriting, 3. defects due to scanning, 4. defects due to printing out the scanned form for signatures, 5. user-added signatures, and 6. defects added during the final scan, which might be on a different scanner or with different scanning settings.

We propose this composite-model framework as an attempt to unify several existing document image analysis models, such as page geometric (or physical) modeling and logical modeling. Our composite-model is more general because it distinguishes between sources of document components and addresses digitization characteristics in addition, rather than addressing different image objects, e.g., handwriting, machine-printed text, logo, signatures, clutter noise, etc. By separating pre-printed information and user-added data, we can examine how one impacts another and more importantly, how they should be exploited for information extraction from groups of documents.

At the implementation level, it is easier to view this paradigm as diagrammed in Figure 1.5. As shown in the literature of document image analysis, traditional pre-processing techniques attempt to clean up images by deskewing page, removing ruling lines, removing scanning noise, etc. In this irreversible procedure, the bitmap is modified without keeping a record of the image processing. Thus, the follow-up modules in the DIA pipeline have lost that part of the original image permanently.

On the contrary, our proposed paradigm ensures image integrity by detecting and storing pre-print information, user added data, and digitization characteristics in a

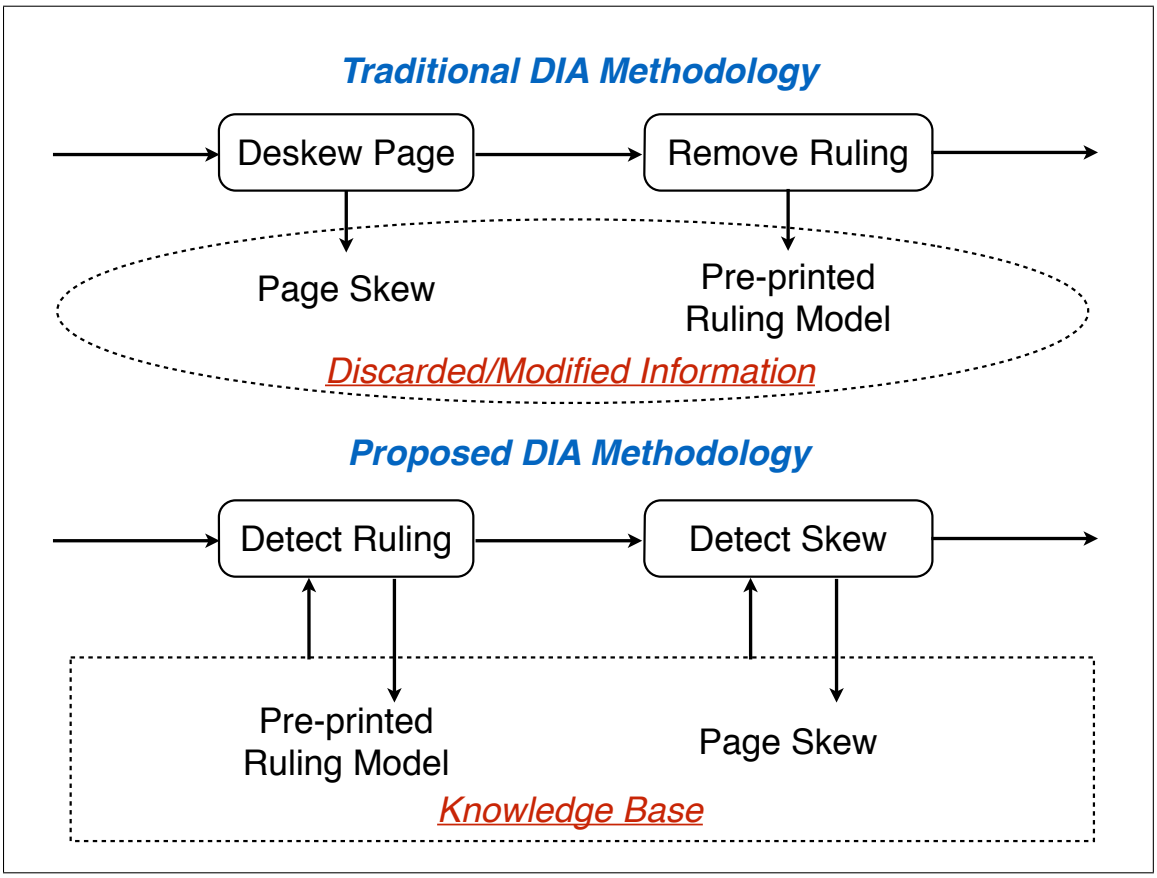


Figure 1.5: A comparison of a typical DIA pipeline and our proposed one. Arrows represents that information flows between pipeline modules.

knowledge base, which is accessible for all the follow-up DIA modules. The knowledge base is a structured collection of extracted document attributes, structures, artifacts, and the knowledge derived from them. Examples include color/gray-level information and the corresponding histograms of channel or gray-scale intensity, pre-printed ruling lines and their attributes, pre-printed tabular structures and their specifications, etc. In this way, no original image data is discarded and we reserve the possibility of making use of them throughout the pipeline.

1.6 The Arabic Handwritten Dataset

Experimental evaluation of this dissertation includes a noisy Arabic handwritten document dataset that contains secret police files of March 1991 uprisings in northern Iraq.[169,170] These files, which record evidence of the Anfal genocide in Iraqi Kurdistan during the 1980s, have been available to the academic community with restrictions since 1998.[170] The Linguistic Data Consortium (LDC)[1] prepared part of this corpus as one evaluation dataset for the Multilingual Automatic Document Classification and Translation (MADCAT) project.[3] One important characteristic of this corpus is that it was not prepared specifically for DIA research, thus it has complicated page layouts, cursive handwriting, artifacts, and digitization defects. We used the form subset of this corpus for our form structure analysis in Chapter 4.

To facilitate Arabic handwriting recognition, LDC also collected a number of handwritten documents by instructing multiple native Arabic speakers with geographic and background diversity. These Arabic subjects wrote text transcriptions on paper sheets of their own choice, using their own writing instruments. We used a subset which contain pre-printed ruling lines in Chapter 3 and Chapter 5.

1.7 Performance Evaluation and Comparison

For binary classification errors,[80] we normally have:

- Type I (*False Positive*): detecting a class that is not present.
- Type II (*False Negative*): failing to detect a class that is present.

Often we need to compare one algorithm with another on the accuracy of a classification problem. According to Dietterich's study on five different types of statistical significance tests, McNemar's test is considered to have low probability of incorrectly detecting a difference when no difference exists.[72]

Suppose there are two algorithms baseline $\mathcal{A}_1$ and proposed $\mathcal{A}_2$. We denote the accuracy of these two algorithms as $\mathcal{F}_1$ and $\mathcal{F}_2$, respectively. In short, McNemar's test is formulated as the following:

$$Z^2 = \frac{(|n_{01} - n_{10}| - 1)^2}{n_{10} + n_{01}}. \tag{1.3}$$

where we first divide misclassified samples into two groups, and then state the hypothesis test:

- n_{01}: number of samples misclassified by the proposed algorithm, but not by the baseline.
- n_{10}: number of samples misclassified by the baseline, but not by the proposed algorithm.
- Null Hypothesis $\mathcal{H}_0$: $\mathcal{F}_1 = \mathcal{F}_2$.
- Alternative Hypothesis $\mathcal{H}_1$: $\mathcal{F}_1 < \mathcal{F}_2$.

The test statistic Z^2 *approximately* follows the χ^2 distribution with 1 degree of freedom. Looking this up in the χ^2 table, we can verify the hypothesis test by claiming whether $\mathcal{A}_2$ is statistically significantly better than $\mathcal{A}_1$. As a rule of thumb, we say one algorithm outperforms another significantly with a confidence level of 95%. The test value Z^2 corresponding to this 95% confidence is 3.84. Although this statistic test is approximate, it is proven to be effective in detecting accuracy

differences between algorithms.[72]

Throughout this dissertation, we claim significant accuracy improvements based on McNermar's test with a confidence level of 95%. The phrase "significant" in this dissertation also indicates the accuracy improvements in questions are verified by this statistical significance test.

1.8 Contributions

We present the following principal contributions:

- A unified framework of modeling document image analysis in which three sources of information are added over time: pre-printed information, user-added data, and digitization characteristics.
- An algorithm for model-based pre-printed ruling line detection which guarantees a global optimal solution given regularly spaced ruling lines.
- An algorithm for ruling-based tabular structure detection which invokes an optimization procedure for selecting plausible rulings that constitute the tabular structure in noisy handwritten documents.
- A study on how to compensate for pre-printed ruling lines during feature extraction rather than pre-processing, and how to exploit them in the task of writer identification.
- An alternative way of handling digitization characteristics for detecting and compensating for them during feature extraction, instead of eliminating artifacts/noise during pre-processing.

1.9 Organization of the Dissertation

We begin with a discussion of related modeling work on document image analysis in Chapter 2. Then, in Chapter 3, we further investigate pre-printed ruling lines and introduce a model-based approach given global least square errors for accurate detection of such pre-printed information in handwritten paper sheets. In Chapter 4, we study how to make use of such pre-printed ruling lines for tabular structure analysis in noisy handwritten documents. We investigate the relations between pre-printed ruling lines, handwriting, and page skew within the framework of the composite-model and demonstrate the benefits of ensuring image integrity in writer identification in Chapter 5. Finally, we conclude with an evaluation summary and discussion on future research directions in Chapter 6.

Chapter 2

Related Work

Our thesis statement is to model document images according to the meta-data framework: pre-printed information, user-added data, and digitization characteristics. Within this framework, we are able to separate information from different sources and thus may be able to either exploit the impact of one component over another or find alternatives for processing information properly.

We survey in the literature the related work on each of these three components and several attempts to model a mixture of them in a document. In addition, we will discuss how our proposed framework unifies these existing attempts to model document images.

2.1 Pre-printed Information

In our composite-model, pre-printed information includes machine-printed text, pre-printed business logos, ruling lines, tables/forms, music scores, legends of engineering drawings, etc. It also includes corresponding meta-data, e.g., font style/size and

language for machine-printed text, spacing, length and thickness for ruling lines, row/column/cell dimensions for tables/forms, etc.

2.1.1 Optical Character Recognition

We have found a large amount of work in the literature for modeling, detecting, and recognizing pre-printed information. OCR is perhaps the most commonly-known task in DIA that has been studied even before digital computers were invented.[175] In this dissertation, we distinguish machine-printed character recognition from hand-written character recognition, as the latter belongs to user-added data and requires significantly different techniques.

Since the 1970s, OCR has been a mature technology with many companies involved in the manufacture and marketing of systems.[99] With hardware advances in optical disks, tape cartridge and scanner sensors since the 1980s, OCR has gone through Latin shape analysis, oriental shape analysis, linguistic context, and global classification.[175]

It helps to clarify scripts and languages for our discussions on multi-lingual OCR. In general, a *script* is one way to write in a specific *language*. It is possible that multiple languages may share the same or similar script. For example, most Western-European languages share variations of Latin script, some Eastern-European languages share variations of Cyrillic script, many Eastern languages share variations of the syllabic script, and many languages based on Islamic culture share variations of Arabic script.

OCR on Latin script is dominated by English OCR. Elastic pixel and curve

matching has been proven to be useful for OCR on binary and gray-level characters and line drawings.[43] Orthogonal Zernike moments, which are affine-invariant features, turn out to be more powerful than the regular moments and the Hu-moments.[127] In addition to various classifier improvements[22,23] and fusion techniques,[103,104] Kahan et al. conduct a benchmark evaluation on multi-font OCR.[120] Baird and Nagy later introduce a self-correcting 100-font classifier which allows a multi-font recognizer to specialize itself automatically to a specific unknown font by examining a few pages as it is reading.[33] Xiu and Baird later delve into this idea of self-correcting OCR and propose a methodology for whole book recognition.[250,251]

Chinese and Japanese Kanji characters differ from Latin ones in that they contain a large number of classes and a complex structure of ideographs.[175] Therefore, it is common to see a hierarchical structure of classification paradigm where the higher-level classifiers perform coarse classification to reduce the character candidates, so that the lower-level classifiers may perform more precise classification.[48,111,174]

Arabic OCR is complicated because character shapes depend on their position within a word. Many characters have four distinct shapes: isolated, initial, medial, and final. In addition, *diacritics* including dots and zigzags constantly confuse the feature extraction because they are isolated from the major body of the characters and thus can be easily treated as scanning noise. The common methodology is to isolate these diacritics first by projection-profile analysis,[10] but errors may occur since they are distinguishing parts between Part-of-Arabic-Words (PAWs) or words.[11–13,82,160] While OCR in Latin, Han/Kanji, and Arabic seems to be a mature technology, research has also extended to other scripts including Indian (Devnagari) script[75,184] and Hangul script.[143]

2.1.2 Script Identification

There has been a growing interest in multi-lingual OCR, which is useful in digitizing business document exchange across countries, and for multilingual countries like India where Hindi, Tamil, Telugu, and English may be used in the same document. To implement multi-lingual OCR, *script identification* is usually performed in order to select a specific OCR engine.

There are two categories of script identification techniques: page-wise and word-wise. Page-wise techniques assume a single script in a whole page while word-wise techniques work on individual words. In general, Asian scripts such as Chinese, Japanese, and Korean are easy to distinguish from Latin scripts due to their distinct structures of ideographs.[224] Also, identification of these three Asian scripts is successful when several text lines are available for training. Hochberg et al. propose a template-based technique to identify 12 scripts and three dozen languages by clustering *textual symbols* that occur frequently in the scripts.[105] Spitz makes use of the fact that Asian scripts are more uniformly distributed in the vertical direction while Latin ones turn to cluster around the baseline.[224] Tan uses rotation-invariant textual analysis, i.e., Gabor filtering based features, to identify six scripts including Chinese, English, Greek, Russian, Persian, and Malayalum.[231]

Word-level techniques have become popular in recent years where a document contains multi-lingual text, as in a Chinese-English dictionary. One straightforward approach to handle multi-script documents is to segment text lines into smaller units such as words, and then extract discriminative features for training and identification. Ma and Doermann investigate this problem on bilingual dictionaries using Gabor filtering based features and compare performance using kNNs, SVMs, and

GMMs.[159] Dhandra and Hangarge introduce a morphological opening based approach which reconstructs images in different directions and computes local descriptors for identifying Kannada, Telugu, Devnagari, and English in Indian documents.[70]

2.1.3 Other Work

Logos, part of pre-printed information, have gained significance in building document image analysis and information retrieval systems. Logo detection and recognition is one critical module in such a system. Much of the former work focuses on the recognition part where knowledge of logo region is assumed.[73,74,180,229] Logo detection can be handled by a top-down, hierarchical manner where the X-Y cut divides a document into segments and then features are computed in each segment for logo and non-logo classification.[213] It can also be solved by a feature-based approach where logos are represented by a bag-of-words model.[205] In order to overcome the drawback of losing spatial arrangement by the bag-of-word method, Rusinol and Llados use an opening operator to find features in the clusters. The combined methodology is also feasible. Zhu et al. propose a multi-resolution framework of logo detection and recognition.[269] They use a Fisher classifier to initially classify connected components at the coarse level, and then each candidate logo blob is further verified by a cascade of simple classifiers. Instead of the multi-resolution approach, Wang and Chen introduce another way of combining logo components where small feature rectangles are detected and they gradually grows until the final rectangle is obtained.[245] Le et al. demonstrate the performance gains of their SIFT-based logo spotting approach over several former methods[147,192,269] on the benchmark Tobacco dataset.[138]

Semi-structured documents like tables and forms are also common examples of pre-printed information. Tables are used to *present* information while forms are designed to *collect* data from users. Although the idea of tables/forms seems to be well understood, it turns out to be difficult to define,[84,153] ground-truth,[108] and evaluate them.[110] The problem of table analysis can be divided into two sub-tasks: table detection and table recognition. Table detection attempts to locate the regions for tabular structures, either using clues like table ruling lines[95,96,247] or regularly-spaced text.[56,109] Table recognition usually assumes identified table regions and the goal is to find the physical structure and the logical structure of a table.[50,66,95,96,202,247] We notice that many existing techniques are evaluated on datasets where rulings are usually salient and sole, meaning no other lines will distract the algorithms for table analysis. This is, however, not the case for the datasets in our experiments where we need to handle severely broken lines and/or false alarms. We will discuss more details in Chapter 4 for related work and our new approach of handling noisy handwritten documents where tabular structure is mingled with other pre-printed information or artifacts.

Another type of pre-printed information that attracts substantial interest is pre-printed ruling lines, which are designed to help people write neatly but people's handwriting constantly overlaps them.[5,21,135,266] The majority of existing work follows a paradigm of detecting and excluding pre-printed ruling lines during pre-processing, treating them as another type of noise and artifacts. It is, however, possible to view this problem from a different angle in our proposed composite-model framework where pre-printed ruling lines are detected and passed along the processing pipeline for follow-up processing, e.g., feature extraction. We will discuss

this more in Chapter 3.

2.1.4 Remarks

Research on pre-printed information is active and fruitful in document image analysis. OCR and script identification are in general considered solved problems given reasonable clean and clear input. Logo detection works well when the connected components of logos and their displacement differ largely from the plain text. Table/form detection has been a popular research topic for decades and continues to be an active area because of their complexity in physical layouts and added complexity of overlapping handwriting and digitization characteristics.

2.2 User-added Data

Within our discussion, the most common type of user-added data might be handwriting, which is the form of writing peculiar to a person. Handwriting is developed in ancient time as a way of expanding human memory and facilitating communication. The reason that handwriting persists in the age of digital computer is the convenience of paper and pen as compared to keyboards for daily situations.[195] Handwriting modalities include normal handwriting, signatures, and hand-drawn sketches. In terms of handwriting applications, there has been substantial research on modeling handwriting recognition, handwriting generation, signature verification, and writer identification. Handwriting can also be divided into *on-line* and *off-line*, with the former being associated with time stamps of each ink pixel while the latter only the static ink image. In this dissertation, we restrain our discussions

on off-line handwriting analysis only because our techniques do not make use of time stamps of on-line handwriting.

No only does handwriting belong to user-added data, inexplicit information introduced by users is also included. For example, physical manipulation of the document page is also included, e.g., folding, tearing, aging, and stapling. For example, aging is common for historical documents where the original ink and paper materials might degrade significantly.

2.2.1 Handwriting Recognition

With the development of OCR techniques, it is a straightforward desire to extend OCR from on machine-printed text to handwritten text. Handwriting recognition, however, is more difficult than OCR in that:

- Handwriting has more variations in the structural ideographs, compared to font/style based machine printed text.
- Traditional text recognition relies on segmented characters or words, but segmentation of handwritten text into lines, words, and characters is considered one of the most challenging tasks in document image analysis.
- People's handwriting may differ across populations, thus it is more difficult to design style-invariant image features.

While OCR is considered a solved problem in DIA,[175] free-form handwriting recognition is still far from becoming a mature technique. Handwritten digit recognition, however, has gained its acceptance in the community based on the obtained high accuracy of over 99% on the challenging MINST dataset.[4,65,141]

As summarized in Plamondon and Srihari's classic survey paper, structural and rule-based methods of handwriting recognition techniques suffer from the difficulty in formulating general and reliable rules to compare textual shapes in a large database of characters and words.[195] Statistical methods, on the other hand, rely on extraction of discriminative image features to train a classifier such that an input image can be recognized based on its statistical characteristics.

Research interest in handwriting recognition has been shifting from character/-word level recognition to line level recognition. Since the 1980s, researchers have focused on designing powerful image features and classifiers for segmentation based character/word recognition.[161,217,235,242] Although segmentation-based approaches may vary in specific implementation details, such as features and classifiers, they follow the high-level processing pipeline below:

- Extract text lines and correct slant of text lines.
- Extract words from text lines and segment them into primitives such as characters or subwords (e.g., PAWs in Arabic).
- Concatenate recognized primitives to recognize words.
- Employ language models to re-rank plausible word candidates.

The word and/or character segmentation in handwriting is challenging because handwritten glyphs have various shapes depending on the writer's handwriting style and the neighboring words or characters. Common methodologies include contour analysis,[129] project profile analysis,[140] run-length analysis,[38] and disjoint box segmentation.[129] In general, these approaches are heuristic local shape analysis, and are difficult to generalize for different glyph shapes or scripts.

Alternatively, segmentation-free methods are proposed to avoid *explicit* segmentation of words and/or characters, such as using Kohonen self-organized feature maps (SOFM),[149] homeomorphic subgraph matching,[204] convolutional time-delay neural networks (TDNN),[93] and Hidden Markov Models (HMMs) framework.[59,83,167] It is arguable that HMM-based approaches have attracted the most attention in the DIA area.

HMMs are a framework in which an underlying stochastic processing is unobservable, and signals can only be observed through another stochastic process that emits a sequence of observations.[83] The transition between hidden states is guided by the transition probability, while the observation consists of a sequence of outputs emitted from a set of hidden states according to some probability distribution function (PDF). Although HMMs assume conditional independence of observation given the state sequence, which limits their application, they are still considered a successful technique for various pattern classification tasks like speech recognition, language modeling, handwriting recognition, and part-of-speech (POS) tagging.

HMMs are appropriate for modeling handwritten text lines or words because character HMMs can be concatenated into word HMMs and further into a line HMM (HMM network). To avoid explicit character or word segmentation, the input text line image is first over-segmented into narrow image frames using a sliding window approach. Then the HMM decoding algorithm finds an optimal alignment of a sequence of image frames with a sequence of states underlying the HMM network (e.g., Viterbi decoding[243]). Thus, the character or word hypothesis is generated by visiting this optimal state path and its corresponding class labels. Finally, language models such as n-grams can be used to re-rank the n-best hypotheses generated

from the HMM decoding.[37,42,83,132,163]

2.2.2 Authorship Analysis

One important piece of information a user incorporates into a handwritten document is her authorship, which has been widely used in forensics for authorship testimony and security applications. Srihari et al. investigate this issue and observe positive proof to support the hypothesis of handwriting individuality.[225] In their experiments, the authors collect 1500 samples from a wide range of populations in terms of gender, age, ethnic groups, geographic conditions, etc. Characteristics such as line separation, slant, and character shapes are validated with a high degree of confidence by machine learning approaches. This evidence serves as positive proof to aid the community of forensics and biometric security.

Writer identification is a task in which given a query input and a database of identified writers, the system outputs the identity of the handwriting. In general, the result given by an identifier is a list of identified names with associated confidence scores in descending order.[35,41,145,206,211,270] Sometimes, a rejection option is available as well.[210] *Writer verification*, on the other hand, is a task which given a query input and a claimed identity, tells if this input comes from the identity as claimed.[116,253] Therefore, writer identification is a 1:N problem and writer verification is a 1:1 problem.

Although writer identification and verification are inherently different problems, they are similar in data acquisition, data interpretation, and solution methodologies. If any text content has been used for identity establishment, this task of

identification or verification is usually called *text-dependent*, otherwise it is *text-independent*. Although on-line and off-line writer identification and verification are both heavily studied areas, we focus on off-line writer identification and verification methods,[35,41,206,211,270] where only spatial information is available. Plamondon and Lorette publish a classic survey paper on writer identification and verification techniques by 1989.[194] Later, Leclerc and Plamondon update this survey with some applications of neural network classifiers.[139]

There has been extensive work in advancing writer identification and verification from the perspectives of classifier and feature extraction. Representative work in classifier design includes Artificial Neural Networks,[270] k-Nearest Neighbors,[100] Weighted Euclidean Distance,[206] Hidden Markov Models,[210] Gaussian Mixture Models,[209] Kohonen Self Organizing Maps,[211] and Support Vector Machines.[119] In terms of feature extraction, there are approaches based on connected-component contours,[211] allographs,[35] Gabor filtering,[206] projection-profiles,[270] chain code,[218] and several others.[100] For a more detailed survey on writer identification and verification, refer to our technical report on this topic.[52] In recent ICDAR conferences, competitions on a 250-writer dataset have pushed forward state-of-the-art techniques in writer identification.[156,157] In the ICDAR 2013 competition, one method based on the contour gradient features computed on character-like segments produces the winning performance.[118]

2.2.3 Remarks

Handwriting recognition has been a popular research topic for decades, with the trend shifting from segmented digit or character recognition to segmentation-free

handwritten text line recognition. Numerous techniques have been proposed to address this task and HMMs are so far the most thoroughly studied approach. Thanks to the closely related speech recognition community, substantial innovations have been made in terms of HMM model parameter estimation (e.g., maximum likelihood, maximum posteriori, maximum mutual information, and minimum phoneme error), HMM state *tying* (e.g., state-level, character-level, and hybrid level), and feature transforms (e.g., linear discriminative analysis, heteroscedastic discriminant analysis, maximum likelihood linear transformation, and region-dependent feature transforms). Thus, HMM-based handwriting recognition can and should make use of such advances in the speech recognition community.

Another trend is that more open competitions have been conducted to investigate how different methodologies might work on the same datasets.[71,91,219,257] These competition datasets usually contain elicited handwriting and/or require curation after data acquisition, thus they are not the exact scenario of people's handwriting in practice. Common curation practice includes removing sample images from the corpus which contain unexpected handwriting or are badly scanned. Nevertheless, they provide substantial motivation for researchers to better understand the problem and thus push forward state-of-the-art handwriting recognition techniques.

Writer identification can also be considered a mature area in the sense that numerous macro and micro features are proven to be powerful with a number of classifiers. Carefully prepared datasets such as IAM[164] and Firemaker[212] are commonly used in large scale writer identification research and competitions. These datasets are explicitly designed for research purposes, thus contain elicited handwriting and require curation. In practice, however, all kinds of artifacts may be

present such as pre-printed ruling lines, which may impact on writer identification techniques substantially. In later chapters of this dissertation, we will investigate how to deal with these pre-printed artifacts when addressing writer identification, and further how viewing separately pre-printed information and user-added data may help benefit authorship analysis like writer identification.

2.3 Digitization Characteristics

Effects of digitization are inevitably introduced during scanning before automatic document image analysis is conducted. From the historical perspective, DIA research is derived from image processing, and various pre-processing techniques are proposed to segment and normalize the input document image such that OCR or handwriting recognition can work on input that is clean and/or more coherent. For example, it has been a common practice for DIA researchers to work on black/white document images. In addition to its conceptual simplicity, other reasons include limited scanning sensor technology and poor computational power on handling a color-encoded image in the old time. As a result, the majority work of feature extraction for OCR, handwriting recognition, writer identification, etc., focuses on bi-level images.

In general, document degradation refers to geometric distortion introduced during photocopying or scanning, and perturbation during the optical scanning and digitization process.[125] Geometric distortions mostly refer to page skew and scaling. Perturbation includes scanning noise such as *salt-and-pepper*, blur, jitter, bleed-through, clutter, etc. It is known that the resulting images differ even if one presses

the scan button twice while leaving the same document in the scanner.[155,268] Various defect models have been proposed to help understand how image quality correlates to performance of downstream DIA tasks.[31,125,146]

2.3.1 Binarization

Binarization has been a challenging problem for document images with low contrast, variable background reflectance, significant noise, and complicated textual patterns. Therefore, no single thresholding techniques provide satisfactory binarization results on a variety of document images.[232] Instead, various local adaptive techniques are proposed to preserve the text information as completely and clearly as possible. A series of experiments[232,234,236,238] have been conducted to examine and evaluate existing local adaptive binarization techniques, including Bernsen,[36] Chow and Kaneko,[62] Eikvil,[81] Mardia and Hainsworth,[162] Niblack,[181] Taxt,[232] Yanowitz and Bruckstein,[255] Parker,[186] White and Rohrer,[249] and Trier and Taxt.[237] The comparison evaluation shows that Niblack's method combined with the post-processing of Yanowitz and Bruckstein's method outperforms the others.

Early evaluation completely depended on visual judgment due to lack of ground-truth.[152,236] Quantitative measurements, however, rapidly gain scientific reputation. There are two common types of evaluation for binarization techniques: pixel-/region-level ground-truth based[63,98,228] and down-stream task performance based, e.g., error rates in OCR[188] or character segmentation.[158] It is straightforward to evaluate by checking whether a pixel is correctly classified as foreground, and vice versa. The ground-truthing part is, however, tedious, time-consuming, ambiguous, and error-prone. Thus, data synthesis[227] is also widely used to obtain pixel-level ground-truth

at low cost, although it is arguable whether synthesized datasets truly reflect the challenges in real-world datasets. Recently, binarization competitions have become a new interesting venue of comparing different approaches and pushing state-of-the-art performance.[89,197–199]

2.3.2 Page Skew Estimation

Another subtask in layout analysis is to determine the page skew, which is introduced during photocopying or scanning. Several approaches are proposed for skew estimation: projection profile analysis, Hough transform, connected component clustering, and cross correlation.[49]

The assumption of projection profile based methods is that text is organized in straight lines.[24,25,64,115,187,196] A projection profile is a histogram of black ink pixels accumulated in the horizontal or vertical direction. Thus, for a document page where text lines are vertically isolated by space, it is expected to see in projection profiles periodic peaks with widths similar to the height of text lines. To determine the page skew, a range of expected skew angles is tested, and the angle that generates the maximum variances in the bin heights is selected as the page skew. Various improvements are proposed to optimize the precision and the computation speed.

Hough transform is a classic method for finding straight lines in computer vision and has been successfully extended to find text lines in document image analysis.[101,137,166,185,223,226,261] It assumes that text lines are parallel to each other. In these methods, each black ink pixel is transformed into a curve in the polar plane parameterized by the distance to the origin, ρ, and the angle, θ. Thus, pixels on the same straight line will intersect in the polar plane and thus the peaks indicate the

skew angle. To reduce the computation complexity, people select a representative subset of ink pixels[261] or apply the transform in a subregion of the entire page.[166]

Nearest neighbor clustering assumes proximity of text that is aligned and close to each other.[97,182,185,221] The generic approach is bottom-up, where connected components or point representatives are extracted, and distance measures are computed for nearest neighbor clustering. For example, in O'Gorman's docstrum algorithm,[182] centroids of connected components are used to find k nearest neighbors, where both the Euclidean distance and the angles between centroid pairs are considered during the search.

Cross correlation takes advantage of the fact that horizontally spanned text lines have small variances at the local scope.[9,90,254] Therefore, a document is divided into fixed-width vertical stripes and horizontal projection profiles are computed along the horizontal direction. The skew angle is estimated by computing the shift of projection profiles or the accumulated correlation of pairs of vertical lines.

2.3.3 Degradation Modeling

Document images are inevitably degraded in the course of printing, photocopying, faxing, and scanning.[30] In general, *degradations*, or *defects*, refer to less-than-ideal properties of real document images, including global effects like geometric deformations, coarsening due to low digitizing resolution, and local effects like ink/toner drop-outs and smears, thinning and thickening, etc.[30] Even if these defects are visually unnoticeable, they may cause significant decline in OCR accuracy.[200,201] Degradation models can be exploited to conduct controlled experiments for studying breakdown points for OCR, create large scale synthetic datasets for sufficient

classifier training, design optimal noise removal algorithms, and predict OCR performance.[124]

There are in general two different methodologies in the literature of degradation modeling. The first is to model physics of the apparatus in detail, where apparatus used in printing and imaging includes some human actions, such as placing a document onto the scanner bed.[30] The other is to model both global perspective deformation and local perturbation during photocopying and optical scanning.[125] The literature mainly focuses on machine-printed document images, but it is possible to extend the modeling to handwritten document images as well.

Baird conducts a series of studies with the first modeling methodology where a single-stage parametric model of per-symbol and per-pixel defects is proposed to model the physics of printing and imaging.[26–29] In this modeling, global distortions such as rotation, scaling, and translation are applied first, then for each pixel, the jitter determines the centers of each pixel sensor, for which the blurring kernel and per-pixel sensitivity noise are applied in order. Finally, each pixel's intensity is binarized to give the output image. In practice, the values of these parameters are decided by a pseudo-random number generator for each symbol.

While Baird's methodology builds on the character and pixel level, Kanungo et al. propose a page level distortion model for perspective and degradation.[125] Perspective distortion occurs during photocopying or scanning thick and bound documents. Degradation refers to perturbation in the digitization process: speckle, blurr, jitter, etc. To validate the document degradation model, Kanungo et al. introduce a statistical, non-parametric approach in which a two-sample permutation test is used.[123] They incorporate the power function to choose distance functions,

and design the validation process to be model-independent such that it can be used for any other degradation model.[121,122,124] Alternatively, Li et al. introduce a task performance oriented validation measure: a degradation model is validated if the OCR errors introduced by the model are indistinguishable from the errors occurring in real-world documents.[146]

2.3.4 Remarks

Binarization has been a common pre-processing module in document image analysis. A number of techniques are proposed to preserve and enhance text information while excluding noise, however, it is still fair to say that no universal binarization approach is able to provide satisfactory results for an arbitrary input document.

Document degradation modeling aims for a systematic way of understanding noise and degradation during document photocopying, scanning, and faxing. Perhaps the biggest impact of these degradation models is in the data synthesis used in extensive classifier training. So far, there are still discussions on whether synthetic document images are comparable to real-life ones, and how human users might be helpful in calibrating parameters in such degradation models.[61]

2.4 Document Layout Analysis

So far, we have seen several DIA tasks that can be described in an individual composite-model component. In the rest of this chapter, we will see several other DIA tasks that involve multiple components in the composite-model framework. Document image analysis aims to build a document hierarchy that captures the

physical structure and the logical meaning of document entities.[148] These entities may range from the document itself to primitives such as characters or isolated connected components. For example, a glyph may be associated with a label, such as a character, and its attributes such as the corresponding ASCII/Unicode value, font style, the geometric position of this character within the page, authorship, etc.

Document layout analysis, or geometric layout analysis, is the task of determining the physical structure of a document, which consists of single connected components, e.g., speckle noise, dots, dashed lines, touching characters, and alternatively groups of connected components or blocks, e.g., a word, a text line, a paragraph.[182] Logical analysis assigns functional labels or data types such as abstract, title, graphics, etc., to each block generated from layout analysis. In our discussion, we focus on document layout analysis because it is straightforward to project corresponding literature work onto our composite-model framework.

In general, document layout analysis can be conducted in a top-down manner where a document page is first divided into one or more column blocks which are further split into text paragraphs and text lines.[32,60,87,94,114,131,178,179] Alternatively, it can be done in a bottom-up manner where primitives such as connected components are grouped into characters, words, text lines, paragraphs, etc.[14,117,130,182,244,246]

2.4.1 Top-down Approaches

The top-down methodology solves the layout analysis task from an image processing perspective, without considering specific types of document components. At the higher level, many approaches concentrate on processing background pixels, or using the white space to identify homogeneous regions within a page.[117]

For example, the X-Y tree based methods[94,134,178] assume upright document orientation and well-separated document components so that alternating horizontal and vertical cuts may work properly. The X-Y tree is a nested decomposition of rectangular blocks into rectangular blocks where at each level, the cut is applied in either a horizontal or vertical direction.[134] The leaves of the X-Y cut represent document image primitives, such as characters, punctuation marks, graph figures, logos, etc. Although the original X-Y cut algorithm works at the pixel level, modifying it to work on connected components results in much faster computation.[94]

Another example of white space analysis is based on thinning background of a page.[8] One advantage is that these approaches rely only on the background area analysis and barely assume any shape constraints on the textual area. In other words, these methods are capable of segmenting pages with non-rectangular blocks as well as with various angles of page skew.[131] First, background thinning gives a representation of connected thin lines or chains for the white space area of any shape. Next, the task of document layout analysis is converted to finding loops enclosing the foreground regions. This procedure is proposed for machine-printed documents and is usually referred as *Voronoi tessellation* in the literature. Several variants and improvements and improvement have been made to handle handwritten documents. The Voronoi++ approach[7] takes into account the fact that degraded handwritten documents usually miss textual components, and contain various sizes of textual components. This approach adapts to local variations in the orientation and distance of document components, in addition to the originally sole consideration of the component size. Further improvement makes use of the content type of components and combines component relationship, local textual patterns, and context features

for page decomposition.[8]

Breuel proposes a method of *whitespace cover* that finds maximal empty rectangles in a document page, with a quick and simple implementation.[39] The idea is analogous to the branch-and-bound method. Given several *obstacles* (pixels or rectangles) within a page, the algorithm selects one as a pivot so that the whole area is divided into four sub-rectangles. Within each region, an upper bound of a quality function is evaluated such that the sub-rectangles along with their obstacles and quality values are inserted into a priority queue. This algorithm ends when the first obstacle-free rectangle appears at the beginning of the queue. This procedure guarantees a global optimal solution.

Lee and Ryu introduce a parameter-free approach in which a pyramidal quad-tree structure is constructed for multi-scale analysis.[144] First, a quad-tree structure is computed by iteratively reducing the resolution of an input image as long as its size is over 50×50. Next, bounding boxes of connected components are extracted and the periodic attributes of each region in the horizontal and vertical direction are computed. If a region fails the periodic test, it is split into two. This procedure continues until all regions are single periodic ones. Further, each homogeneous region is identified as text, ruling lines, tables, or images based on a series of heuristic rules.

In general, top-down approaches have the advantage of making use of the global page structure to conduct layout analysis quickly.[183] However, if the page does not contain a linear bound or the figures are intermingled with text, these approaches can fail. Examples include magazine pages where line art and graphical figures are usually mixed with the text region, and also handwritten pages where pre-printed ruling lines make the white space less salient to detect and trace.

2.4.2 Bottom-up Approaches

In contrast to top-down approaches, bottom-up ones attempt to detect and cluster primitives (pixels, connected-components) into homogeneous regions until all blocks are found on the page. This methodology has the advantages over the top-down one in that it is able to handle complicated layouts such as intermingled graphics and text, non-Manhattan layouts, etc. On the other hand, however, its downside is that it is computationally more expensive.[183]

Docstrum is a bottom-up approach that uses k nearest neighbor clustering to group characters into text lines and blocks.[182] First, connected components are extracted from the page image and the k nearest neighbors are computed according to their Euclidean distances. Next, each pair of components is described by a tuple of corresponding Euclidean distance and the angle between the centroids of the two components. Thus, a pair of characters on the same text line usually has small distance and close to zero skew. On the other hand, between-line pairings have larger distances than within-line ones and the angles are approximately 90°. After plotting the 2-tuple values from all pairings, a transitive closure is performed on within-line pairings to extract the connected components on the same time line. Then, a line fitting algorithm is applied on the centroids of connected components to obtain the representation of text lines. This algorithm has an advantage of being skew insensitive, and is able to estimate the page skew as a by-product, as discussed in Section 2.3.2.

Fisher et al. introduce a rule-based method for segmenting document page image into text and non-text regions.[86] First, a run length smoothing algorithm (RLSA[244]) is applied after binarization and skew correction during pre-processing. Next, the

horizontal smoothed image is logically *ANDed* with the vertically smoothed one, giving the RLSA image. Then, connected components are extracted and their bounding boxes are computed to calculate several important statistics such as aspect ratio, black pixel density, Euler numbers, perimeter length, perimeter to width ratio, etc. Finally, text and non-text blocks are classified based on these feature statistics.

The above mentioned methods either only discriminate between foreground and background, or simply treat foreground as text and non-text. To further facilitate document image understanding, Jain and Yu propose a hierarchical document structure model where more specific document components are detected and labeled.[117] They construct the hierarchical document representation based on a bottom-up approach where block adjacency graph (BAG) nodes are extracted to constitute connected components, horizontal and vertical lines, generalized text lines (GTLs), region blocks, and so on. The proposed primitive, a BAG node, is a foreground block of several horizontal run lengths stacked and aligned at both left and right end. After BAG node extraction, connected components are computed by finding connected BAG nodes in a graph. Based on a set of rules, a document representation is organized in a top-down manner where a journal document page consists of text regions and non-text regions including tables, halftone images, drawings, and ruling lines.

Lee et al. propose a pure rule-based method for page decomposition in technical article pages.[142] The knowledge rules are divided into region segmentation ones and identification ones. Region segmentation starts with page skew correction and connected component extraction. Then, the document is segmented into columns based on analysis of the projection profiles, followed by an iterative segmentation in

the vertical direction. Similar to the X-Y cut method,[134,178] this approach assumes Manhattan layouts. Making use of the knowledge that technical article pages usually consist of text regions such as titles, abstracts, and author information, as well as non-text regions such as photos, tables and drawings, the authors design a series of rules to help classify each region. In general, there are 40 rules with a total of 44 threshold values that are designed specifically for IEEE T-PAMI article pages. Applying this approach for other types of document pages would require adjustment.

Liang et al. introduce an algorithm for Latin-character document layout analysis using a Bayesian framework.[148] The document structure is also organized as a hierarchical tree where the document page resides at the root and characters are at leaf nodes labeled as the glyphs. This Bayesian framework assigns and updates the probabilities during the page segmentation process and iteratively finds the segmentation solution that maximizes the probability of the extracted document structures. This method is generic and can be applied to document hierarchy construction at any level, e.g., words, lines, text blocks, etc. The authors validate their modeling in the task of text line extraction where they follow a bottom-up strategy to extract connected components. Then a left and a right connected component are grouped together to compute their likelihood of being in the same text line. Next, the baseline direction is estimated and corrected if the page skew is larger than a threshold. Finally, a homogeneous labeling and grouping adjustment is repeated to ensure the maximum probability. The experimental evaluation on UW-III dataset shows an accuracy over 99% from a total of 105K text lines.

2.4.3 Remarks

Document layout analysis or page decomposition is a heavily studied area in document image analysis. In addition to the representatives mentioned above, several competitions aim to evaluate the state-of-the-art performance.[15–20] One trend is that document page layout analysis is in transition from modern documents to historical ones, the latter of which is considered to contain more noise and thus are more challenging.

So far the methods of document layout analysis that we discuss focus on the perspective of image processing, in the sense that either they make use of the white space streams to segment the page image into regions in a top-down manner, or they extract primitives such as characters and connected components in a bottom-up manner. All the above mentioned methods assume some known page layout styles implicitly.

There are several papers, however, that explicitly use document style in the task of page decomposition and attempt to simulate the process of document generation.[126,222] For example, Spitz uses a GUI tool to generate a style sheet that defines document regions and their corresponding logical labels. This style sheet is used throughout the batch processing as a starting point of a search for document regions.[222] Kanungo and Mao use a stochastic framework for stimulating the document image generation processing, which consists of five models: a logical structure model that specifies the semantic relations among logical components, a language model that generates text for logical components, a physical style model that specifies the physical appearance and spatial relations, a typesetting model that converts the symbolic document into a noise-free image, and a document noise model that

mimics the document degradation.[126]

Previous papers of document layout analysis can be viewed in our proposed framework. First, both top-down and bottom-up methods deal with the tasks of segmenting pre-printed information into homogeneous regions such as characters/-words/lines, figures, tables, logo, in the presence of digitization characteristics such as page skew, document warping, and scanning speckle noise. Second, style-directed methods as in Kanungo and Mao's work[126] show more clearly that the logical structure model, the text language model, and the physical style model control the pre-printed text information, whereas the typesetting model and the document noise model control the digitization characteristics in the result document images. Finally, our composite-model separates the processing of pre-printed information and user-added data such as handwriting, and thus makes it possible to build and exploit the relation between these two components.

It is more demanding to analyze noisy handwritten documents. Handwriting exhibits more variations than machine-printed text in terms of its glyph shapes, displacement within page, and possibility of overlapping other components. In addition, the authorship of handwriting present on a page provides a tremendous amount of information for indexing and retrieving in a document corpus. Another important fact is that handwriting is often added after a page is printed out, so it is not rare to see handwriting overlaps other document structures or components on a handwritten page. Therefore, handwritten document image analysis may require a different methodology than machine-printed DIA.

A number of methods have been proposed to distinguish machine-printed and handwritten text.[190,265] In general, these methods model a handwritten document

page with three classes, machine-printed text, handwriting, and noise.[265] After applying several image features to classify blocks as initial results, Markov Random Fields (MRFs) are used to post-process initial classification results.

This task of separating handwriting and machine-printed text can be more complicated when they overlap.[34,189] This is possible since pre-printed information is generated before handwriting is added, according to our composite-model framework. Banerjee et al. adopt the methodology of removing the noise and then restoring broken text strokes using MRFs[34] while Peng et al. use finer units (sub-word patches) to segment overlapping parts and use a coarsening process to generate larger regions for follow-up feature extraction and classification.[189]

So far, we have discussed several DIA tasks and demonstrated how they relate to our proposed composite-model framework. We also claim that not only is it intuitive to model document images using our composite-model with information preserving individual models, but also beneficial to do so in order to ensure data integrity for the downstream tasks, such as document indexing, knowledge extraction, and information retrieval.

In the following chapters, we will see one important type of pre-printed information, i.e., ruling lines, that has drawn much interest recently in the research community and we will demonstrate how these ruling lines impact people's handwriting. Also, we will discuss how this pre-printed information affects traditional pre-processing procedures that aim to provide a clean image for analysis.

Chapter 3

Pre-printed Ruling Detection

Our composite-model framework requires detecting and recording document components rather than modifying them during pre-processing. Pre-printed ruling lines commonly present in notebooks are designed to help people write neatly. However, they introduce a series of problems because of overlap with handwriting or other document components. In this chapter, we will introduce a model-based ruling detection algorithm that makes use of vertical spacing regularity and guarantees a global-optimal estimation of the ruling line attributes. In the later chapters, we will demonstrate how to compensate for such artifacts and further exploit them for discriminative image features.

3.1 Introduction

Line processing is needed in various document analysis applications, e.g., forms/invoice processing,[267] table analysis,[102] engineering drawing processing,[78] music score

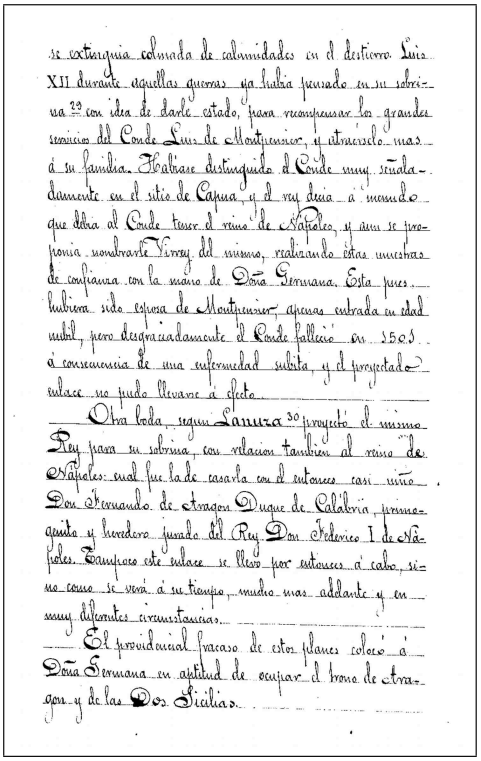
analysis,[47] and off-line handwriting analysis.[5] Many techniques work well on relatively clean images of good quality.[21,53] However, if lines are severely broken due to low image resolution or they are overlapped by other components, ruling lines may be missed or spurious rulings may be detected. For example, Cao and Govindaraju introduced a method of processing low-resolution noisy medical forms, where the authors modeled a degraded image using Markov Random Fields (MRFs) and made use of the contextual information of MRFs to infer missing text parts after removing ruling lines.[44]

For a particular application, prior knowledge is helpful in designing specific algorithms.[266] Unlike in the other applications, pre-printed ruling lines on paper sheets exhibit a simple but strongly correlated pattern:

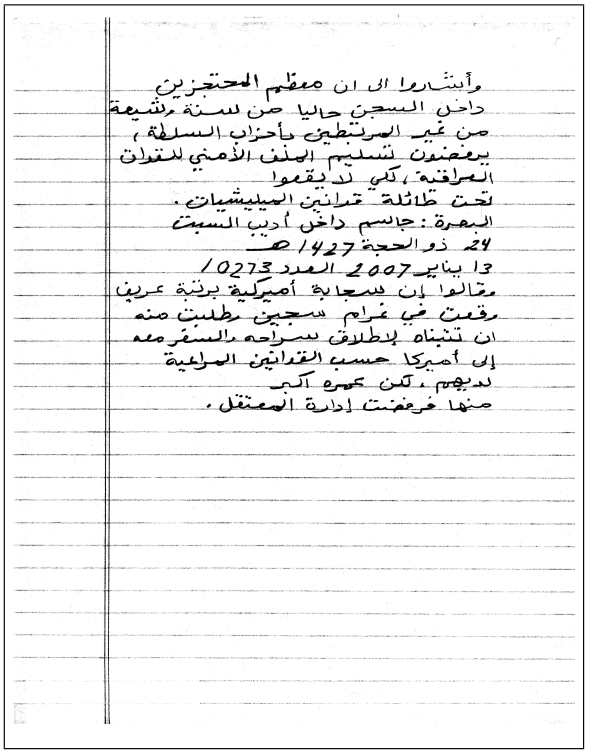
- Ruling lines are parallel straight lines.
- They have consistent spacing, length, and thickness.

On the other hand, since people usually make use of ruling lines when they are present, separating handwriting that overlaps ruling lines can be a significant challenge. Figure 3.1 shows two sample documents used in our experiments.

The protocol of performance evaluation uses either pixel-level metrics or object-level ones. Pixel-level metrics including *precision*, *recall*, and *F-score* are intuitive measurements for performance evaluation. However, ground-truthing at pixel level is difficult because pixel-level judgement is subjective and this situation becomes more severe when lines are degraded. On the other hand, researchers have presented several object-level metrics.[106,133,151,193,264] Although these compound metrics are designed to incorporate meaningful components, it can be difficult to show how



(a) A sample in Germana.



(b) A sample in Madcat.

Figure 3.1: Sample documents used in our experimental evaluation.

significantly the performance differs among algorithms. For example, Liu and Dori design one object-level metric for evaluating performance for engineering drawing processing:[151]

$$Q_v(c) = [Q_{pt}(c) \cdot Q_{od}(c) \cdot Q_w(c) \cdot Q_{st}(c) \cdot Q_{sh}(c)]^{1/5}. \tag{3.1}$$

where the vector detection quality $Q_v(c)$ is the geometric mean of five factors: end-point quality $Q_{pt}(c)$, overlap-distance quality $Q_{od}(c)$, line-width quality $Q_w(c)$, line-style quality $Q_{st}(c)$, and line-shape quality $Q_{sh}(c)$. If two algorithms' Q_v-values

differ by 0.1, we still do not know how significant the differences are. Researchers have also used the performance of downstream applications for evaluation, such as *word error rates* (WERs) for handwriting recognition.[44,46]

We introduce a model-based ruling line detection algorithm that takes advantage of the model of ruling lines. Next, we present the framework of multi-line linear regression and derive a globally optimal solution under the Least Squares Error (LSE). Then we describe an effective Hough transform variant for extracting line segments and the adaptive Basic Sequential Algorithmic Scheme (BSAS clustering) to group line segments. The next step makes use of the ruling line properties to detect lines that are missed by the Hough transform. Finally, we use the multi-line linear regression to estimate the model parameters.

For performance evaluation, we choose to compute the error statistics of the model attributes individually, rather than defining a single metric. We consider this an effective way of showing different aspects of how the algorithm performs, and indicating what future improvement can be made. In addition, we evaluate performance by measuring the effort needed for a human subject to correct algorithmic errors. To do that, we provide a human subject with a GUI that enables him/her to interactively correct algorithmic errors. During this process, the GUI records the duration of editing, the number of clicks, and all adjustment operations. Then we show the editing time and the number of clicks required to correct any algorithmic errors on three public test datasets.

3.2 Related Work

3.2.1 Line Processing

In engineering drawing processing, three categories of methodologies exist: thinning based,[173,230] medial line extraction based methods,[76,78,168,172] and hybrid.[107] Thinning based algorithms usually use an iterative morphological *erosion* so that only an one-pixel wide skeleton remains in the image. Next, skeleton pixels are grouped into line segments and a polygonal approximation is used to convert these segments into vectors. However, line thickness is lost during the processing. In medial line extraction based methods, run-length is commonly used for preserving line connectivity and thickness, as well as for efficient line detection.[168] For example, Dori and Liu introduced a vectorization approach that first converted graphic objects in a raster image into vector forms.[78] They used a thinning-free Sparse Pixel Vectorization (SPV) algorithm that only visited a selected subset of the medial axis points. This procedure computed a crude polyline that was later refined through a polygonal approximation algorithm. The authors reported that their algorithm was more robust than the Orthogonal Zig-Zag (OZZ) algorithm.[76]

For music score analysis, *staff lines* are critical for recognizing notes and pitches.[68] In Roach and Tatem's work, they detected staff lines using a sliding window.[203] Within the window, the authors measured the angle of the run-length that started from the center of the window to its furthest black pixels. Using the angle information, they were able to identify horizontal staff lines. As a run-length based

approach, Carter and Bacon presented a Line Adjacency Graph (LAG) method.[47] Their algorithm was able to handle difficult situations where a symbol tangentially intersected with the staff lines. d'Andecy et al. attempted to segment music scores into four detectable layers.[68] They used the Kalman filter to separate these layers, which is robust to scaling, curvature, and noise in music score images.

Forms/Invoice processing consists of documents without handwriting[150,239] and those with handwriting.[45,256,258,260] In this discussion, we focus on the latter case where in general, handwritten form documents usually contain three components: form frames, preprinted information (titles, labels, instructions, logos, texts, etc), and handwriting. Yu and Jain presented a block adjacency graph (BAG) method to detect form frame lines.[260] Each node in the BAG represented a run length in the image, which was similar to the LAG method.[47] Each edge represented the adjacency between nodes. Horizontal frame lines were connected nodes with large *aspect* ratios. Ye et al. used the morphological *opening* operation with linear shape *structure elements* on foreground pixels to remove frame lines that were longer than a predefined length.[256] Then, to restore information that was removed by the line removal processing, they used a *closing* operation with a dynamic structure element for different orientations (90°, 45°, and 135°). Given a known form, Cao and Govindaraju applied a template matching method to mask the horizontal ruling lines on low-quality handwritten carbon forms.[45] Since handwriting constantly intersected with these ruling lines which were broken after masking, the authors used Markov Random Fields (MRFs) to restore the lost handwriting information.

Line processing is also necessary in image-based handwriting recognition. Arvind

et al. introduced a rule-based method that first detected the ruling lines within segmented handwritten blocks by computing the horizontal projection profiles.[21] To minimize the profile entropy, the authors computed the skew angle and detected the positions of ruling lines by investigating the peak positions in the horizontal projection file. They then performed run-length analysis to determine which pixels belonged to the ruling lines. Zheng et al. presented a stochastic model based ruling line detection algorithm that incorporated context to detect ruling lines systematically.[266] Using a vectorization based method called "Directional Single-Connected Chain" (DSCC), the authors separated most handwriting from a set of line segments.[267] Rather than treating the peaks on the projection profile as the line positions, they modeled the profile with a Hidden Markov Model so the constraints among lines could be incorporated. However, when dealing with ruling lines preprinted on paper sheets, they did not make use of other constraints such as consistent spacing, skew angle, and line length. Abd-Almageed et al. introduced a ruling line removal algorithm based on modeling rulings in linear subspaces.[5] Kumar and Doermann introduced a fast ruling line removal algorithm that takes advantage of integral images to compute line features and uses a re-sampling scheme to reduce the samples for training an SVM.[135] Most methods in the literature treat ruling lines as an artifact in documents that are removed for follow-up processing, so ruling lines are eliminated for all downstream tasks which might benefit from this information. Details are presented in Chapter 5.

3.2.2 Performance Evaluation

In the literature on line processing, several performance measures have been introduced in different applications in line processing, including pixel-level measures[5,21] and object-level measures.[106,133,264] Pixel-level measures include precision, recall, and F-score. They are easy to understand, but labeling at pixel level is tedious and usually subjective, especially when document images are degraded.

On the other hand, several object-level measures have been designed to measure certain characteristics of lines in different applications. For example, Kong et al. developed a performance measure for evaluating dashed line algorithms.[133] In addition to the overlap criteria including angle and distance between a ground-truthed line and a detected line, they also considered end point detection and line styles.

Hori and Doermann introduced a protocol for evaluating lines in CAD representations.[106] The line attributes consist of end points, line thickness, and line styles (dashed, dot, solid), in addition to other feature points that are useful in CAD applications, such as T-junctions, crossing points, and corner points.

Liu and Dori designed a compound measure for evaluating straight and circular line detection.[151] Their measure included matching-degree computation and a set of evaluation indices at both pixel and vector levels. By computing the geometric mean of five qualities, the authors produced a single metric to measure performance, as shown in Eq. 3.1.

Phillps and Chhabra designed a general protocol in evaluating graphics recognition systems.[193] They proposed evaluation algorithms for line-line matching, arc-arc matching, arc-line matching, arc-circle matching, circle-circle matching, and text-text matching.

Researchers have also used performance measures from downstream applications to evaluate line processing algorithms. For example, Cao et al. adopted the *word error rate* (WER) of handwriting recognition to evaluate the performance of their ruling line removal algorithm.[46]

For our purposes, we want to use measures that are reasonably simple while retaining enough information about the line attributes. At the same time, these measures should be useful in calculating the effort needed for a human user to correct any algorithmic errors.

3.3 Multi-line Linear Regression

Our ruling line detection algorithm builds upon the linear regression model under the Least Squares Error (LSE). In what follows, we first explain the single linear regression and then derive a variant for the ruling line detection.

3.3.1 Linear Regression

Consider the simplest case in linear regression: given a set of points in the plane $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, the linear model is defined as the following:

$$\beta_0 + \beta_1 x + \epsilon = y \tag{3.2}$$

or equivalently,

$$\boldsymbol{X\beta} + \boldsymbol{\epsilon} = \mathbf{y} \tag{3.3}$$

where $\mathbf{X} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}, \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}, \boldsymbol{\epsilon} = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}, \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$, and refer to $\mathbf{X}$ as the *design matrix*, $\boldsymbol{\beta}$ as the *parameter vector*, $\boldsymbol{\epsilon}$ as the *random error vector* with zero mean ($E(\epsilon) = 0$) and unknown variance σ^2, and $\mathbf{y}$ as the *observation vector*. The task then is to find the parameter vector $\hat{\boldsymbol{\beta}}$ such that $\mathbf{X}\,\hat{\boldsymbol{\beta}}$ approximates $\boldsymbol{y}$ *best*, meaning $\boldsymbol{\epsilon}$ has the smallest error variance. Using the Least Squares Estimation, we want to minimize the *residual function* $F(\boldsymbol{\beta})$:

$$F(\boldsymbol{\beta}) = \sum_{1}^{n} \epsilon_i^2 = \boldsymbol{\epsilon}^T \boldsymbol{\epsilon} = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^T (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) \tag{3.4}$$

Expanding $F(\boldsymbol{\beta})$, we get

$$\begin{aligned} F(\boldsymbol{\beta}) &= \boldsymbol{y}^T\boldsymbol{y} - \boldsymbol{\beta}^T\boldsymbol{X}^T\boldsymbol{y} - \boldsymbol{y}^T\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\beta}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} \\ &= \boldsymbol{y}^T\boldsymbol{y} - 2\boldsymbol{\beta}^T\boldsymbol{X}^T\boldsymbol{y} + \boldsymbol{\beta}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} \end{aligned} \tag{3.5}$$

Note that $\boldsymbol{\beta}^T\boldsymbol{X}^T\boldsymbol{y}$ is a scalar, so we take the derivative with respect to $\boldsymbol{\beta}$:

$$\frac{\partial F}{\partial \boldsymbol{\beta}} = -2\boldsymbol{X}^T\boldsymbol{y} + 2\boldsymbol{X}^T\boldsymbol{X}\hat{\boldsymbol{\beta}} = 0 \tag{3.6}$$

Rearranging the items, we obtain the *Least Squares normal equations*:

$$\boldsymbol{X}^T\boldsymbol{X}\hat{\boldsymbol{\beta}} = \boldsymbol{X}^T\boldsymbol{y} \tag{3.7}$$

Given that the column vectors in $\boldsymbol{X}^T\boldsymbol{X}$ are linearly independent, then $(\boldsymbol{X}^T\boldsymbol{X})^{-1}$ exists. Thus we can compute $\hat{\boldsymbol{\beta}}$ as:

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{3.8}$$

We adopt the Least Squares estimator approach since it has several important statistical properties. First of all, this is an *unbiased* estimator, *i.e.*, $E(\hat{\boldsymbol{\beta}}) = \boldsymbol{\beta}$. To see this,

$$\begin{aligned} E(\hat{\boldsymbol{\beta}}) &= E[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}] \\ &= E[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon})] \\ &= E[(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} + (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{\epsilon}] \\ &= E[\boldsymbol{I}\boldsymbol{\beta}] + (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T E[\boldsymbol{\epsilon}] \\ &= \boldsymbol{\beta} \end{aligned} \tag{3.9}$$

Furthermore, the *Gauss-Markov Theorem* establishes that the ordinary Least Squares estimator of $\boldsymbol{\beta}$ is BLUE (*best linear unbiased estimator*).[171] Finally, if we assume a normally distributed random error, then the Least Squares Estimation is also a Maximum Likelihood Estimation (MLE).

3.3.2 Multi-line Linear Regression

We derive a variant of the standard linear regression model for our work. The multi-line linear regression model assumes the following:

(i) The underlying geometric model is for straight lines.

(ii) All these straight lines preserve consistent spacing and skew angle.

(iii) The association of point-to-line is known.

(iv) The number of lines k is known.

(v) No lines are missing between the first and last lines.

It is important to clarify that properties (iii), (iv), and (v) are guaranteed by the first phase of our algorithm, and are not fundamental limitations of our formulation of the problem.

Next, since we assume that the number of lines k and the point-to-line association are known, the points set $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ can be further formulated as $\{(x_{i,j}, y_{i,j}) \mid i = 1, \ldots, k; j = 1, \ldots, n_i; \sum_{i=1}^{k} n_i = n\}$. In addition, all the ruling lines are parallel and have consistent spacing, so we rewrite the normal form of the line equation as:

$$\beta_0 + i\beta_1 + \beta_2 x_{i,j} + \epsilon_i = y_{i,j} \tag{3.10}$$

where $i = 1, \ldots, k$, $j = 1, \ldots, n_i$, and $\sum_{i=1}^{k} n_i = n$. $(\beta_0 + \beta_1)$ is the y-intercept of the first line, β_1 is the spacing between lines, and β_2 is the skew angle.

Now we rewrite the residual function as:

$$\begin{aligned} F(\beta) &= \sum_{i=1}^{k} \epsilon_i^2 \\ &= \sum_{i=1}^{k} \sum_{j=1}^{n_i} (y_{i,j} - \beta_0 - i\beta_1 - \beta_2 x_{i,j})^2 \end{aligned} \tag{3.11}$$

To minimize the residual, we take the partial derivatives of f with respect to $\beta_0, \beta_1, and \beta_2$, respectively.

$$\begin{cases} \frac{\partial F}{\partial \hat{\beta}_0} = -\sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n_i}} 2(y_{i,j} - \hat{\beta}_0 - i\hat{\beta}_1 - \hat{\beta}_2 x_{i,j}) = 0 \\ \frac{\partial F}{\partial \hat{\beta}_1} = -\sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n_i}} 2k(y_{i,j} - \hat{\beta}_0 - i\hat{\beta}_1 - \hat{\beta}_2 x_{i,j}) = 0 \\ \frac{\partial F}{\partial \hat{\beta}_2} = -\sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n_i}} 2x_i(y_{i,j} - \hat{\beta}_0 - i\hat{\beta}_1 - \hat{\beta}_2 x_{i,j}) = 0 \end{cases} \tag{3.12}$$

Rearranging these equations into the form of Eq. 3.3, we have

$$\begin{bmatrix} \sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n_i}} 1 & \sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n_i}} i & \sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n_i}} x_{i,j} \\ \sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n_i}} i & \sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n_i}} i^2 & \sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n_i}} i\ x_{i,j} \\ \sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n_i}} x_{i,j} & \sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n_i}} i\ x_{i,j} & \sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n_i}} x_{i,j}^2 \end{bmatrix} \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \\ \hat{\beta}_2 \end{bmatrix} = \begin{bmatrix} \sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n_i}} y_{i,j} \\ \sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n_i}} i\ y_{i,j} \\ \sum_{\boldsymbol{i=1}}^{\boldsymbol{k}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n_i}} x_{i,j}\ y_{i,j} \end{bmatrix} \tag{3.13}$$

The 3-by-3 design matrix is a symmetric matrix with positive entries. By multiplying the inverse of the design matrix on the left on both side, we get the normal equations as in Eq. 3.7. From physical considerations where the number of lines and the point-to-line association are available, we can expect a unique solution as $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$.

3.4 Model-based Ruling Line Detection

Our model-based ruling line detection algorithm assumes that documents preserve salient, although not necessarily continuous and complete, ruling line segments. We start by separating clutter noise, which is large background noise present around the boundary of a document image, by differentiating connected-component sizes at a low resolution scale of 25%. Then, we work on the image layer as follows without the clutter component.

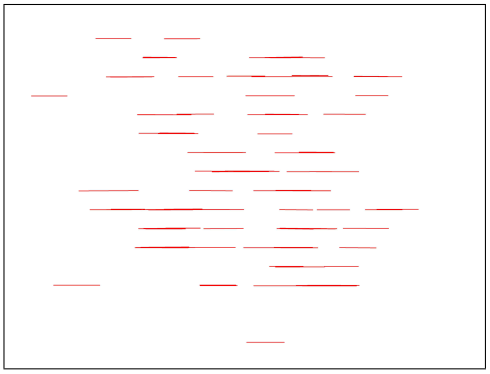
3.4.1 A Variant of The Hough Transform

The classical Hough Transform is a method for detecting predefined features in digital images.[113] As suggested by Duda and Hart,[79] researchers usually use the normal form to detect lines through collinear subsets of points $\mathcal{P} = \{(X_i, Y_i), i = 1, 2, \ldots, \mathcal{M}\}$. The transformation from Cartesian coordinate (x_i, y_i) to the polar coordinate (ρ, θ) is defined as follows:

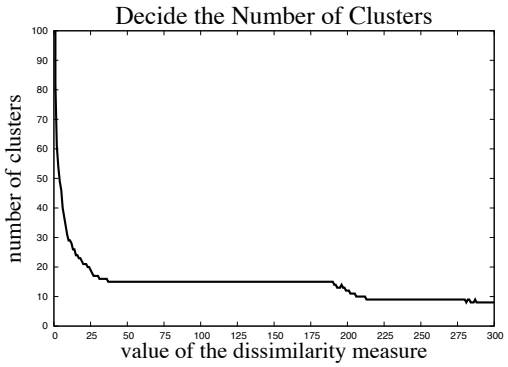
$$\rho = x_i \; cos\theta + y_i \; sin\theta \tag{3.14}$$

where $\theta \in [0, 2\pi)$. By this equation, each image point (x_i, y_i) is transformed into a set of points on a sinusoidal curve in the (ρ, θ) plane (the Hough Space). Given a set of points on a straight line, the intersections of the corresponding sinusoidal curves indicate the parameters of corresponding lines. Thus, the line-finding problem is transformed to a peak-detection problem in the accumulation matrix.

We use a simplified, yet efficient variant because broken lines are common in degraded document images.[165] First, in each iteration we select a point randomly



(a) Line segments generated from the Hough Transform variant.



(b) Dissimilarity measure $\boldsymbol{d}(\cdot, \cdot)$ against its corresponding number of clusters m.

Figure 3.2: An example showing how to decide the most probable clusters. By varying the dissimilarity values, we compute the number of clusters using the ordinary BSAS clustering, then select the largest flat area in the plot as the most probable clustering result.

from the remaining point set, and then compute its sinusoidal curve in the Hough space and update the accumulation matrix. If the current maximum votes is larger than the threshold, then we search in each direction from the current position for the end points of the line segment. Since ruling lines may be broken, short gaps are allowed during the search. Once the search stops, we record the coordinates of the end points of the line segment, remove all these points from the accumulation matrix, and proceed until no candidate points are left.

3.4.2 Sequential Clustering

After the Hough transform, we obtain a set of line segments specified by their end points, as shown in Figure 3.2(a). Since in many cases ruling lines are broken, we need to cluster those segments which belong to the same line. In our work, it is

straightforward to use the ρ-value distance between line segments as the dissimilarity metric, because ruling line segments are assumed to be parallel to each other.

Defining $\mathcal{Q}$ as the maximum number of clusters, we outline the "Basic Sequential Algorithmic Scheme" (BSAS[233]) in Algorithm 1. This algorithm favors compact clusters in the sense that line segments are tightly clustered with respect to their ρ-values. Only one pass is required over the dataset G. Since the total number of clusters m is expected to be much smaller than N, the algorithm operates in linear time complexity $\mathcal{O}(N)$. In our early work,[54] we adopted this approach because of its simplicity and efficiency.

Algorithm 1: Basic Sequential Algorithmic Scheme (BSAS)[233]

Input: $G = \{\mathbf{x_i} : i = 1, \ldots, N\}$: a set of points ($\rho$ values of line segments).
$\mathcal{T}$: value of the dissimilarity measure.
Output: m: number of clusters obtained.
begin
  $m = 1$;
  $C_m = \{x_i\}$;
  for $i = 2$ **to** N **do**
    Find $C_j : \boldsymbol{d}(x_i, C_j) = min_{1 \leq p \leq m} \boldsymbol{d}(x_i, C_p)$;
    if $\boldsymbol{d}(x_i, C_j) > \mathcal{T}$ **and** $(m < \mathcal{Q})$ **then**
      $m = m + 1$;
      $C_m = \{x_i\}$
    else $C_j = C_j \cup \{x_i\}$;
    ;

However, this algorithm has a few shortcomings.[233] First, the algorithmic output depends on the order of the input sequence. Second, it is hard to estimate the value of the dissimilarity measure $\mathcal{T}$ for all inputs because of varying spacing. Therefore, we adapt the BSAS algorithm by estimating the number of clusters and the corresponding value of the dissimilarity measure simultaneously.

Algorithm 2: Adaptive BSAS[233]

Input: $G = \{\mathbf{x_i} : i = 1, \ldots, N\}$: a set of points ($\rho$ values of line segments).
Output: m: number of clusters.
$\mathcal{T}$: corresponding value of the dissimilarity measure.
begin
  for $i = 1$ **to** $\mathcal{E}$ **do**
    for $j = 1$ **to** *100* **do**
      $G^{'} = \text{random\_shuffle}(G)$;
      $m^{'} = \text{BSAS}(G^{'}, i)$;
      accumulate in the histogram of $m : H_m$;
    $m^{'}$ = the index of $max\{H_m\}$;
    $\phi(i) = m^{'}$;
  m = y-value of the widest flat region in $\phi(\cdot)$;
  $\mathcal{T}$ = x-value of the widest flat region in $\phi(\cdot)$;

Let $\mathcal{E}$ denote the upper bound of the threshold of dissimilarity between clusters. This procedure is outlined in Algorithm 2. We plot the function $\phi(\cdot)$ in Figure 3.2(b). The rationale of selecting the widest flat region is that line segments within the same cluster (a straight line) are expected to be compact with respect to their ρ values. In addition, the minimum distance between clusters (r) is much larger than the maximum distances within clusters ($r1$): $r \gg r1$. In other words, all these clusters are well separated. Therefore, if we select $\mathcal{T}$ to be within $(r1, r - r1]$ and run the BSAS clustering, we obtain the same number of clusters. However, we need to set up a bounding value $\mathcal{E}$ for $\mathcal{T}$, otherwise a large enough $\mathcal{T}$ can always generate one single cluster with all line segments in it, rendering the clustering result useless.

After the BSAS clustering, we estimate the ruling line spacing s by building a histogram of spacing values between two consecutive clusters. This value is temporary – in a later stage we update it by re-computing the spacing globally and hence, more precisely.

3.4.3 Single Line Fitting

After combining close clusters, we now have a good idea of which line segments belong to which cluster. At this stage, we use the linear regression on each cluster, as formulated by Eq. 3.3. Then, we compute a temporary skew angle by averaging all β_2 values. Again, this skew angle is tentative and will be fine-tuned by the multi-line linear regression at a later stage. The ruling line thickness $\mathcal{H}$ is computed by first building a histogram of vertical run-lengths along each line. We then examine the histogram and select the most frequent bin as the thickness $\mathcal{H}$.

3.4.4 Reasoning About Missing Lines

For degraded documents, it is common for the Hough transform to miss light and/or broken lines. Thus, we traverse the clusters checking whether two consecutive clusters have a much larger spacing than the temporary spacing. In such cases, we hypothesize that there may be missing lines in between. We estimate the positions of missing lines by considering the spacing and the skew angle, then we scan along the missing lines to collect sample points for further regression. This procedure is outlined in Algorithm 3.

The subroutine *ScanZones* specifies either "North" or "South" for the scanning direction, and it uses either "strict" or "relaxed" criterion for collecting evidence. For the strict criterion, we decide that a ruling line exists only if we collect more than $\mathcal{T}_1$ sample points from the scanning areas. For the relaxed criterion, we do not set such a threshold. We apply the strict criterion at the topmost and bottommost lines on the page, and the relaxed criterion for all other lines. The rationale is that if we want to add missing lines to the top/bottom of the current candidate list,

we need strong evidence; otherwise we can derive their positions from the spacing between existing lines. In this way, we iteratively scan for missing lines until the pixel count is lower than the threshold or the process reaches the edge of the image.

Algorithm 3: Find Missing Lines.

Input: A list of lines: $lines_{old}[m]$. The temporary spacing: s.
Output: An updated list of lines: $lines_{new}[m']$.
begin
  $lines_{new}$ = ScanZones($lines_{old}[0]$, s, "North," "strict");
  for $i = 0$ **to** $m - 1$ **do**
    $space = lines_{old}[i+1].\rho - lines_{old}[i].\rho$;
    $count = space/s$;
    $local\_s = space/count$;
    if $abs(space) > 1.5\ s$ **then**
      $lines_{new}$ = ScanZones($lines_{old}[i]$, $local\_s$, "South," "relaxed") ;
  $lines_{new}$ = ScanZones($lines_{old}[m]$, s, "South," "strict");

3.4.5 Computing Model Parameters

At this stage, we have satisfied all prerequisites for Eq. 3.13 in Section 3.3. Solving this equation, we obtain the estimated parameter vector $\hat{\boldsymbol{\beta}}$. Next, we update the linear equation for each cluster. Then for each cluster, we scan the areas that extend from the leftmost and the rightmost point. Newly discovered sample points are collected as the new start and end points for that line. We use the maximum line length as the ruling line length $\mathcal{L}$. At the same time, we can determine the starting position of the first ruling line $\mathcal{P}(x_p, y_p)$.

Algorithm 3 relies on the local spacing estimate to find missing lines, so it may not be fully reliable. Hence, we run another round of missing line scanning using the global spacing and a new threshold $\mathcal{T}_2$. If there are additional ruling lines detected

at this stage, we update the corresponding model parameters, i.e., the number of ruling lines $\mathcal{K}$ and the starting position of the first line $\mathcal{P}(x_p, y_p)$.

To summarize, the model parameters determined by our algorithm are:

(i) starting point of the first line $\mathcal{P}(x_p, y_p)$

(ii) the length $\mathcal{L}$

(iii) the thickness $\mathcal{H}$

(iv) the skew angle $\boldsymbol{\beta_2}$

(v) the number of lines $\mathcal{K}$

(vi) the spacing $\boldsymbol{\beta_1}$

Thus, we denote the model parameters as $\Theta = (\mathcal{P}(x_p, y_p), \mathcal{L}, \mathcal{H}, \boldsymbol{\beta_2}, \mathcal{K}, \boldsymbol{\beta_1})$.

3.5 Experimental Evaluation

3.5.1 Data Preparation

We evaluated our algorithm on both synthesized and real datasets. First, as a simple test of correctness of algorithm, we synthesized a dataset that contained only ruling lines using predefined parameter settings. We ensured that no noise was added in these pages and that they were created in black and white only. One

subset consisted of 11 pages where each had 20 ruling lines and the same length, thickness, spacing, but each page had a different skew angle, ranging from $[-1.0°, 1.0°]$ with a step size $0.2°$. The other one consisted of 10 pages where each had the same length, thickness, skew angle, but each had a different number of lines: [10, 20), thus the spacing was different as well. The ground-truth for this dataset was generated directly from the predefined model parameters.

We also used another three real datasets for performance evaluation: Madcat,[1] Germana,[191] and Field.[266] The Madcat dataset was provided by the Linguistic Data Consortium (LDC). This is an Arabic handwriting dataset where all documents are scanned at a resolution of 600 DPI and then binarized. A common size for these scanned document pages is $5100w \times 6600h$. The Germana dataset originated from a Spanish manuscript of 1891, in which most pages contain cursive handwriting on sheets with ruling lines.[191] It has approximately $21K$ text lines manually marked and transcribed by paleography experts. All the pages were binarized and had a size of $1420w \times 2120h$. The Field dataset was used in Zheng et al.'s work[266] which contains 167 Arabic handwritten pages with various page dimensions and layouts. The pre-printed rulings in this dataset are usually broken and the scanning resolution is 200 DPI, making them difficult to detect.

For performance evaluation, we randomly selected 100 pages from Madcat, 86 from Germana, and 84 from Field as the testing datasets. For the training datasets that are required by Zheng et al.'s HMM training, we selected another 100 pages from Madcat, 87 from Germana, and the remaining 83 from Field. The Hidden Markov Model in their algorithm was trained using their annotation tool that enabled a user to label ruling lines on a line-by-line basis. A breakdown of the three datasets is

Table 3.1: A breakdown of datasets used in the experimental evaluation.

Dataset	# of Pages	# of Lines	Page Size	Spacing	Thickness	Source
synthetic-rotation	11 pages	210	$816w \times 1056h$	624 pixels	1 pixel	N/A
synthetic-spacing	10 pages	145	$816w \times 1056h$	vary	1 pixel	N/A
Madcat	100 pages	2,894	$5100w \times 6600h$	varying	varying	multi-writer
Germana	86 pages	2,057	$1420w \times 2120h$	90 pixels	1 pixel	single-writer
Field	84 pages	2,098	varying	varying	varying	multi-writer

listed in Table 3.1.

The ground-truth for our evaluation was generated using an annotation tool developed at Lehigh. Based on the ruling line model, the GUI allows users to label ruling lines efficiently with a few simple "point-click-and-drag" operations. Figure 3.3 shows the GUI of our annotation tool. As shown in this figure, pre-printed ruling lines within one page are sufficiently ground-truthed using three line handles: two horizontal ones and one vertical.

In our experiments, we empirically set $\mathcal{E} = 100$ for Germana and Field, and 300 for Madcat, since images on Madcat are larger than those in the other two. The idea is to set this spacing parameter to be ∼1.5 times the ruling line spacing, in order to get a robust estimation from Algorithm 2. In addition, we set $\mathcal{T}_1 = 0.2 \times page\_width$ for scanning missing lines using the local spacing. To rescan missing lines using the globally estimated spacing, we set $\mathcal{T}_2 = 0.03 \times page\_width$ for the heavily degraded dataset Germana and Field, and $\mathcal{T}_2 = 0.15 \times page\_width$ for Madcat. The idea here is to obtain high precision of true ruling line segments from any degraded image. All these parameters are empirically set after studying ∼5 documents independently from each dataset.

To ensure the efficacy of using the GUI, we conducted simple tests to find the

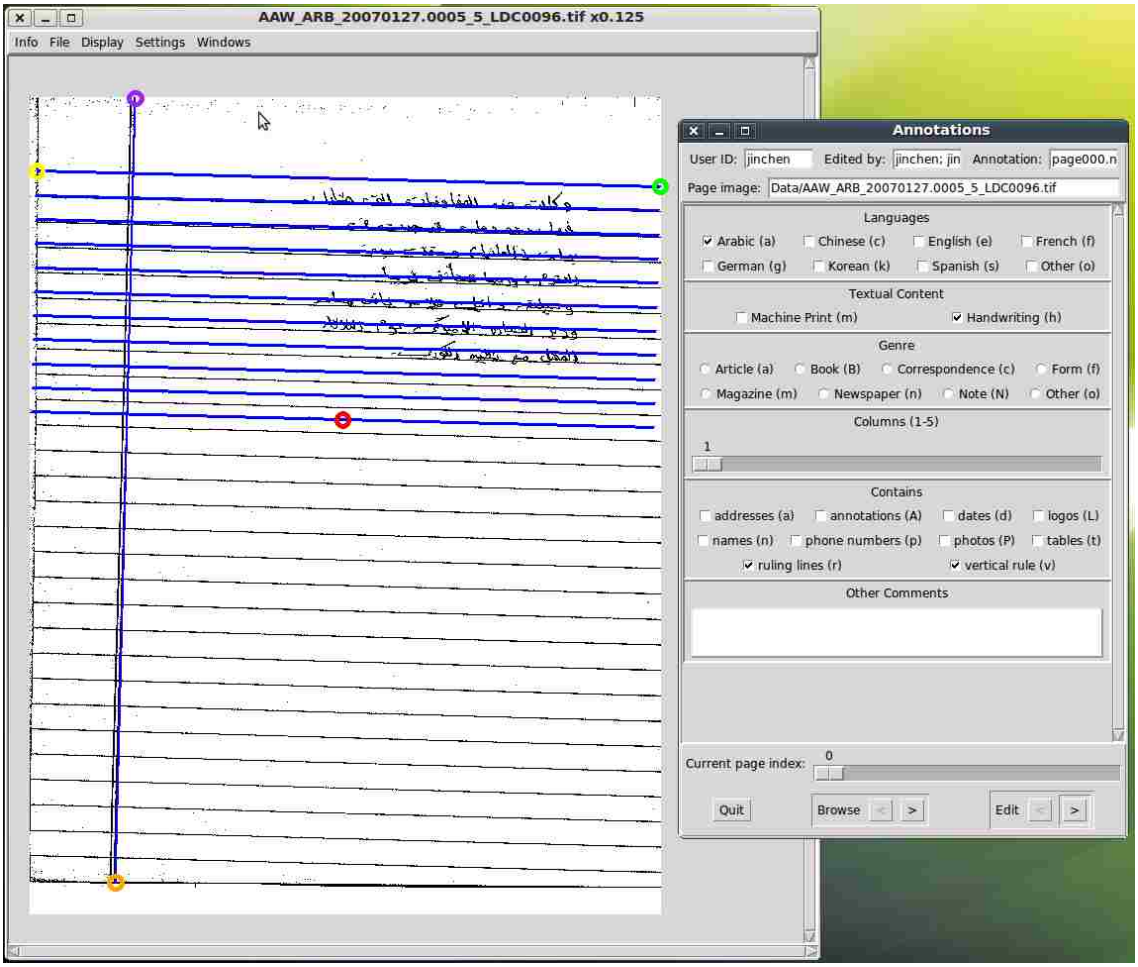


Figure 3.3: The annotation GUI for the annotation of pre-printed ruling lines.

variances of human subjects' annotation, which can serve as a metric for interpreting algorithmic results. Six college students were invited to participate in this study in which five pages each from synthetic-rotation, synthetic-spacing, Madcat, and Germana were selected for subjects to annotate.

3.5.2 Performance Evaluation Metric

Instead of proposing a compound metric that attempts to combine all parameters into a single value, we chose to measure directly the discrepancies between the computed parameters and the ground-truth where $\Theta = (\mathcal{P}(x_p, y_p), \mathcal{L}, \mathcal{H}, \boldsymbol{\beta_2}, \mathcal{K}, \boldsymbol{\beta_1})$:

$$D[i] = M_{algorithm}[i] - M_{ground-truth}[i] \tag{3.15}$$

where $D[]$ is the error vector and $M_{(.)}[]$ contains the algorithmic output or the ground-truth. Then we can compute the mean and the standard deviation of errors for each dataset.

3.5.3 Obtaining Results from an Existing Algorithm

Zheng et al.'s algorithm relies on several pre-processing steps on the input images. First, the clutter noise around the border was removed. Second, handwritten text was filtered out in order to compute ruling positions more precisely. Third, all input images were normalized to [1,000, 2,000) for image width and height. In addition, the discrete HMM model was suggested to make fully use of the training samples[1].

Since their algorithm is a supervised approach, they also provided us with the annotator GUI for ground-truthing training and evaluation datasets. Figure 3.4 shows a snapshot of their GUI system. The GUI system supported optional text filtering before annotating line segments. An embedded line finding algorithm was used to find primary line segments. The end points of each line segment were

[1] All these pre-processing and training strategies are kindly suggested by Yefeng Zheng.

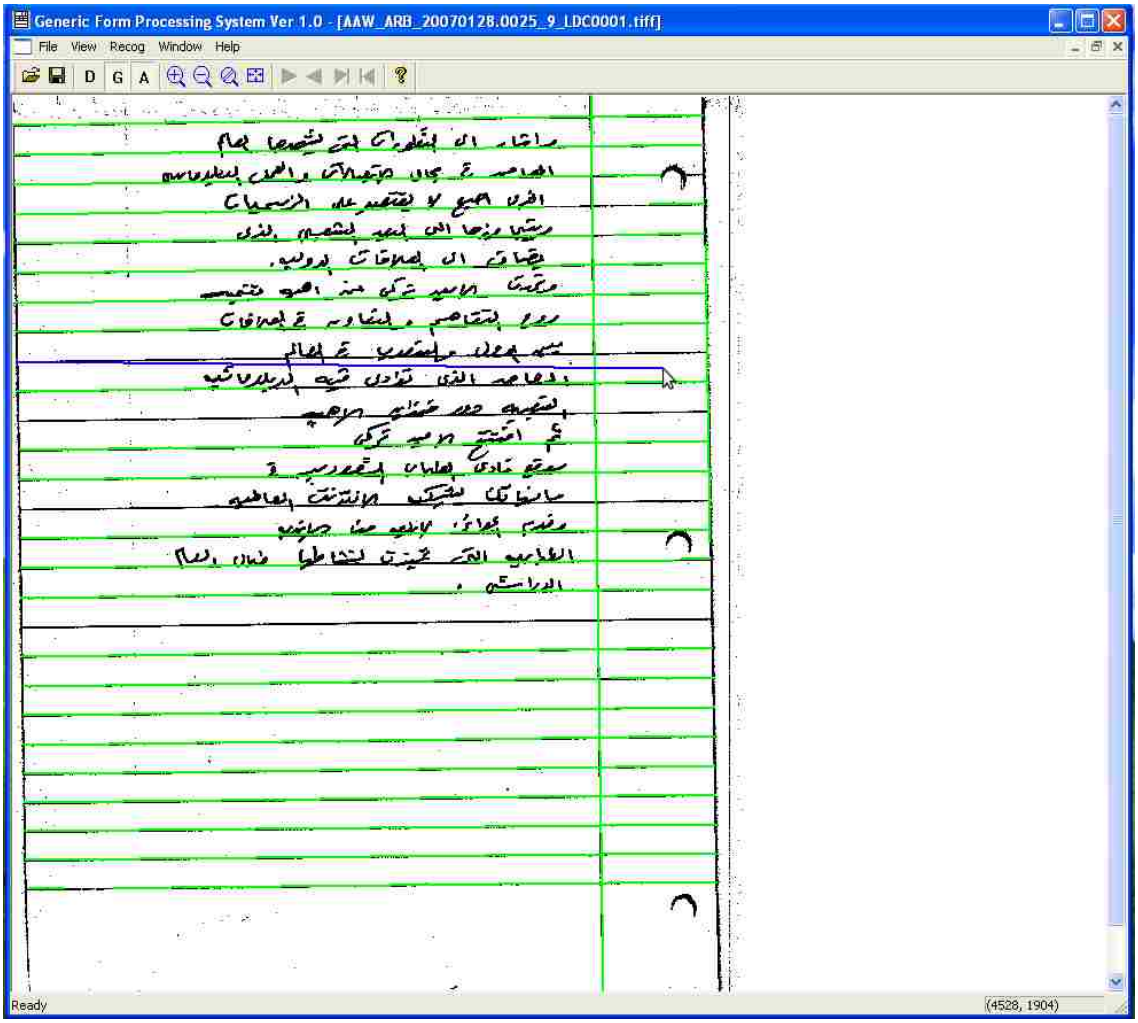


Figure 3.4: The annotator GUI for the annotation of lines in Zheng et al.'s work.

manually adjusted to fine-tune the ground-truth.

The output of their algorithm is a list of line segments specified by their end points, thus it is necessary to derive the model attributes for evaluation. The number of lines $\mathcal{K}$ was taken to be the list size. The spacing $\boldsymbol{\beta_1}$ was computed the same way as the temporary spacing s in Section 3.4.2. The skew angle $\boldsymbol{\beta_2}$ was the average for all line segments. The thickness $\mathcal{H}$ was obtained as described in Section 3.4.3.

Finally, $\mathcal{P}(x_p, y_p)$ and $\mathcal{L}$ were computed as described in Section 3.4.5.

3.6 Experimental Results

3.6.1 Observations on Using GUI

Six human subjects participated in the annotation tests. Each of them was asked to take some time with the Lehigh GUI and then start to label ruling lines. We show the average of standard deviations $\overline{\sigma}$ for different datasets in Table 3.2. $\overline{\sigma}$ is computed as:

$$\overline{\sigma} = \frac{1}{m} \sum_{i=1}^{m} \sqrt{\sum_{j=1}^{n} \frac{(V_j - \overline{V})^2}{n-1}} \tag{3.16}$$

where m is the number of samples to label, n is the number of human subjects involved, V_j means an attribute's value in Subject j's annotations, and $\overline{V}$ denotes the mean value for that attribute across all subjects' annotations.

The annotation variances are small on synthetic datasets. This is probably because all the synthetic images are free of noise and their ruling lines have regular patterns. One indication is that although the granularity of *X-position* $\mathcal{P}.x$, *Y-position* $\mathcal{P}.y$, and *Length* $\mathcal{L}$ is one pixel, the average standard deviation of human subjects' annotations is less than 1.00. The relatively large mean standard deviation of line length (27.38) on Madcat is because of the dimension of page images ($5100w \times 6600h$). In other words, the mean standard deviation of line length is within 1.00% of the page width. Based on the error statistics in Table 3.2, we determine that the

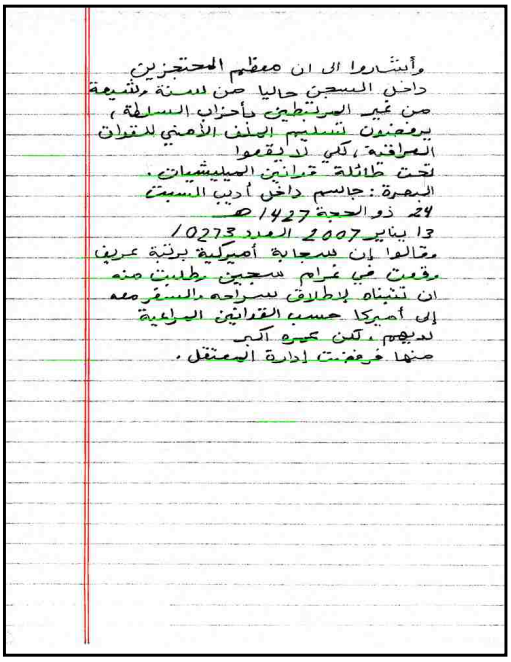
Table 3.2: Experimental results of tests on using a GUI. We compute the average standard deviations of the errors in each model attribute as follows. $\overline{\sigma}$ below is defined in Eq. 3.16.

Statistics	**X-** $\mathcal{P}.x$	**Y-** $\mathcal{P}.y$	**Length** $\mathcal{L}$	**# of lines** $\mathcal{K}$	**Spacing** β_1	**Skew** β_2	**Thickness** $\mathcal{H}$
	Synthetic Dataset						
$\overline{\sigma}$	0.49	0.27	0.95	0.00	0.03	0.04	0.04
	Madcat Dataset						
$\overline{\sigma}$	5.55	2.22	27.38	0.00	0.17	0.08	3.30
	Germana Dataset						
$\overline{\sigma}$	3.77	0.75	5.50	0.00	0.21	0.18	0.51

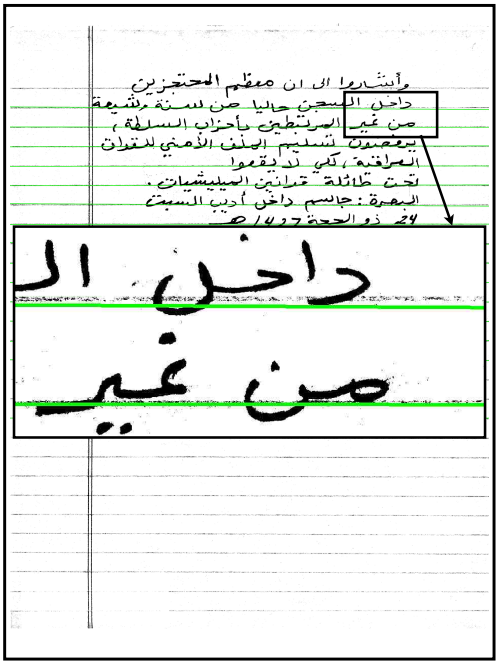
GUI is effective in annotating ruling lines and these error statistics can be used for interpreting the algorithmic evaluation results.

3.6.2 Ruling Line Detection

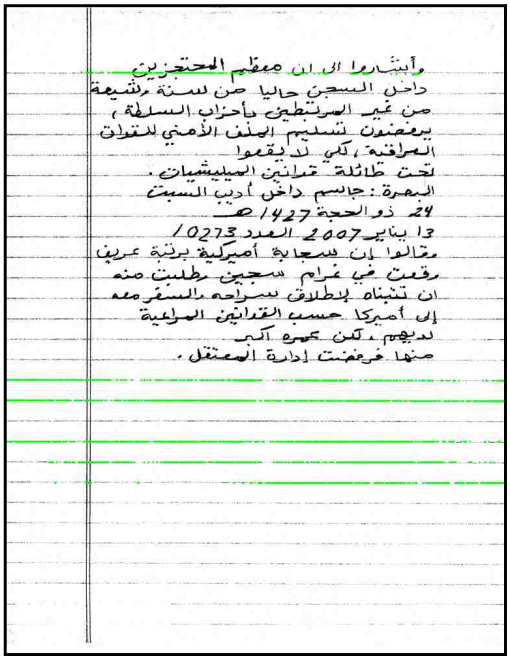
We plot some intermediate results of our model-base ruling line detection algorithm in Figure 3.5, generated using a Madcat sample. Since the ruling lines are clearly printed and are scanned at a high resolution, a majority of the ruling lines were present in the scanned images. First, salient line segments were successfully extracted by the Hough transform variant, although several light and broken parts were missing. Next, after the single-line linear regression, we obtained an estimation of the skew angle and consistent spacing. Then, making use of the spacing, we probed at the top and the bottom of the page for missing lines. We found several light and broken lines in the image, as shown in Figure 3.5(c). Finally, we estimated the skew and the spacing using the multi-line linear regression and checked again for missing lines. The output image is shown in Figure 3.5(d). Note that several missing lines at the bottom of the page were successfully detected. Figure 3.5(b)



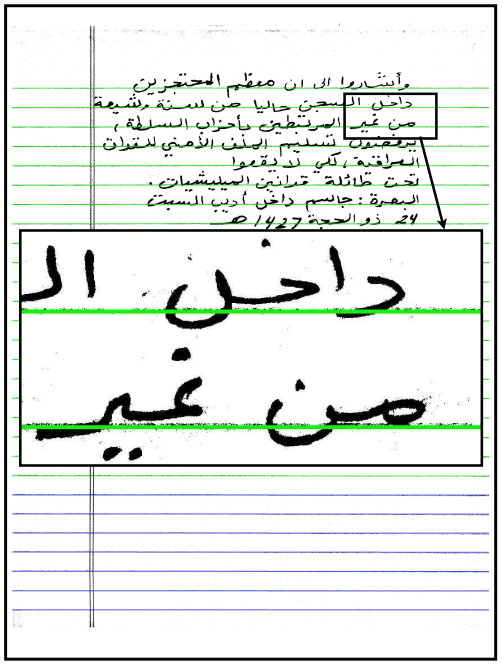
(a) Detected clusters of line segments.



(b) Single line linear regression.



(c) Detected missing lines.



(d) Multi-line linear regression.

Figure 3.5: Intermediate results of our algorithm on a Madcat sample.

shows that single-line linear regression may not be accurate for finding other pixels on the line. Our multi-line linear regression, however, takes advantage of the model to find ruling lines, and thus more precisely (see Figure 3.5(d)).

Among all model parameters, the number of lines $\mathcal{K}$ may be the most straightforward measure for performance evaluation. Our algorithm correctly detected 95% of the rulings on Madcat, 79% on Germana, and 62% on Field, while Zheng et al.'s algorithm detected 81% on Madcat, 48% on Germana, and 57% on Field. The performance improvements are statistically significant according to the McNemar test (Section 1.7), with a confidence level of 95%. Errors on Germana and Field are mainly due to the degradation introduced during scanning; most ruling lines outside the text region are heavily eroded and only few isolated pixels presented. Although the performance difference on Field seems small, an examination on the failure cases for each method shows that the majority of our algorithm's mistakes occur at the top and the bottom boundaries of the ruling area, while Zheng's approach misses ruling lines in regions with clustered text.

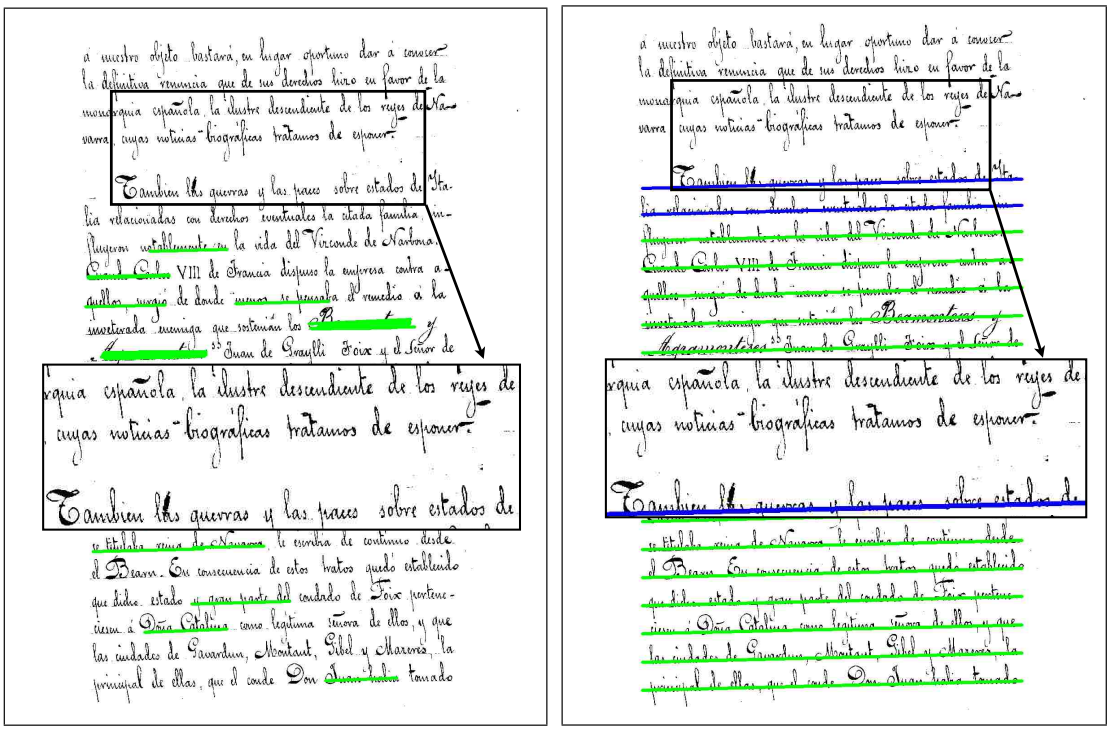
The error statistics are organized in Table 3.3. It is expected that the synthesized datasets were the *easiest* to test: The mean and the standard deviation of errors are the lowest compared to those on the other three real datasets. Madcat has a high scanning resolution (600 DPI) with a large amount of scanning noise within the page, especially in the border area. This is one of the reasons why we observed more errors in terms of the length $\mathcal{L}$ and the x-position of the starting point $\mathcal{P}.x$. There are more errors on the x-positions than the y-positions. This is reasonable since for a given horizontal ruling line, noise and/or broken line segments around the starting position can easily cause confusion for both human subjects and algorithms, whereas

Table 3.3: Performance comparison with one existing algorithm.

Error Stats.	X- $\mathcal{P}.x$	Y- $\mathcal{P}.y$	Length $\mathcal{L}$	# of lines $\mathcal{K}$	Spacing β_1	Skew β_2	Thickness $\mathcal{H}$
	Synthetic Dataset						
Mean (μ)	-4.16	0.55	6.9	0	0	-0.01	0
Std. Dev. (σ)	5.60	2.18	5.11	0	0.01	0.05	1.41
	Madcat Dataset						
Mean (μ)	-50.55	-4.66	76.87	0.03	0.06	0.05	-4.63
Std. Dev. (σ)	43.44	21.42	43.94	0.22	0.29	0.10	1.62
	Germana Dataset						
Mean (μ)	-5.93	1.50	10.60	0.06	-0.31	0.11	-1.01
Std. Dev. (σ)	25.82	37.76	26.88	1.39	3.02	0.38	0.11
	Field Dataset						
Mean (μ)	-41.26	-11.33	66.04	0.02	0.35	0.04	0.06
Std. Dev. (σ)	61.37	121.06	81.89	2.78	3.02	0.25	0.59
	Madcat Dataset (Zheng et al.[266])						
Mean (μ)	-16.60	-26.15	41.20	0.10	-0.86	0.05	-8.53
Std. Dev. (σ)	21.41	69.25	26.44	0.46	0.74	0.10	3.14
	Germana Dataset (Zheng et al.[266])						
Mean (μ)	-47.64	-4.88	91.64	6.70	-12.88	0.01	-0.10
Std. Dev. (σ)	60.81	20.00	61.79	8.26	16.21	0.29	0.33
	Field Dataset (Zheng et al.[266])						
Mean (μ)	-51.67	-28.02	83.57	0.13	0.03	0.03	-0.01
Std. Dev. (σ)	69.50	76.30	85.28	1.74	0.51	0.06	0.55

the determination of the y-position is less affected because it can be computed reliably by other pixels in the line.

We observed that all errors occurred when the algorithms tried to detect missing lines at the top or bottom. This may be due to the fact that we used a hard threshold to decide whether one missing line exists, and it is difficult to estimate a threshold that works for every input. For example in Figure 3.6(a), the two-paragraph page has one extremely light ruling line in between these two paragraphs. In this case, the Hough transform failed to detect any line segments in the first paragraph. Although several missing lines were detected in the second paragraph, no further recovery was made for ruling lines in the first one. The output image is shown in Figure 3.6(b). As a possible future improvement for detecting missing lines, we could consider more flexible decision making schemes rather than one rigid threshold value.



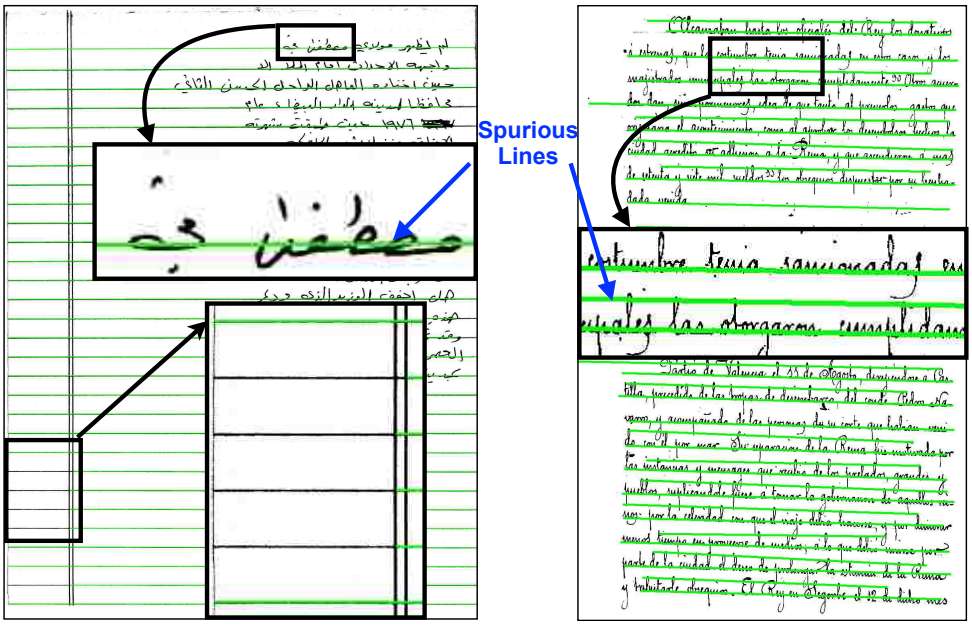
(a) Detected line segments. (b) Multi-line linear regression result.

Figure 3.6: An error case where one ruling line between two paragraphs is missing by our line scanning algorithm.

3.6.3 Comparison on Model Attributes

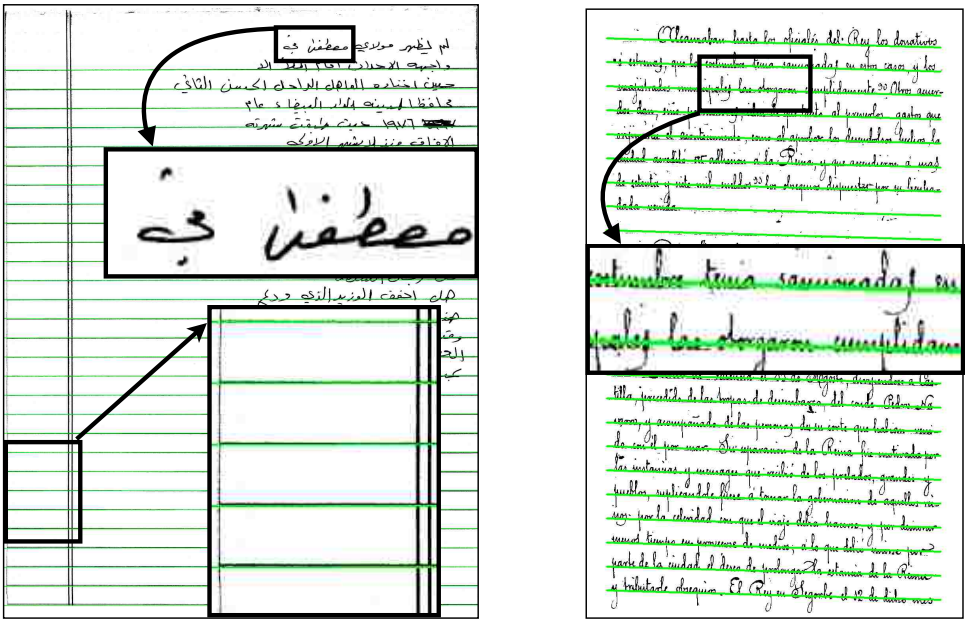
We compare our algorithm with Zheng et al.'s in Figure 3.7. Zheng et al.'s algorithm missed some ruling lines, and also reported some spurious ruling lines. On the other hand, for degraded images on Germana, our algorithm managed to detect light and broken ruling lines derived from other salient ones, which is one of the advantages of our ruling line modeling.

Table 3.3 shows the error statistics of the two algorithms. The number of detected lines is a straightforward metric. Our algorithm reduced the mean error of



(a) Zheng et al.'s algorithm on Madcat.

(b) Zheng et al.'s algorithm on Germana.



(c) Proposed algorithm on Madcat.

(d) Proposed algorithm on Germana.

Figure 3.7: Comparison with Zheng et al.'s algorithm.[266] (a), (b) are outputs by Zheng et al.'s algorithm, while (c), (d) are by our algorithm.

finding correct number of ruling lines from 0.1 to 0.03 on Madcat, 6.70 to 0.06 on Germana, and 0.13 to 0.02 on Field, compared to the HMM-based approach.

The HMM-based algorithm determined the starting point of rulings and length more accurately on Madcat. This is probably because our algorithm was too aggressive in collecting end points but around the border of Madcat images, scanning noise confused the algorithm. However, our algorithm made fewer errors on the number of rulings, the skew angle, and the spacing.

On Germana, our algorithm made fewer errors on all attributes except for skew and thickness. This is probably due to the fact that there is local page warping at the corner of several pages such that the full-page adjustment mentioned in Section 3.4.5 estimates the positions of some of the ruling lines less accurately than line-by-line detection. As a result, for these warped pages, there are parts of detected ruling lines that slightly deviate from the actual ruling lines. This might also explain why their algorithm's thickness estimation is also better.

Field, which contains various page types and layouts, was found to be the most challenging dataset in our evaluation. Although our algorithm detected ruling lines correctly on 62% of the input images, the mean error ($\mu = 0.02$) showed that our algorithm managed to obtain almost correct results. Error analysis showed that most mistakes were made when scanning for the topmost and bottommost light rulings, where severe scanning noise confused the algorithm.

3.6.4 Comparison on Human Efforts

In addition to traditional attribute based evaluation, another paradigm of performance evaluation is to employ human users for correcting algorithmic errors. First,

we used a GUI to display detected ruling lines on the original image. Next, a human user changed the handles in the GUI to adjust the *position*, *skew*, *length*, *spacing*, and *number of lines* if he/she considered the algorithmic output to be imprecise. During this process, the duration of editing, the number of clicks, and all operations of adjustment were recorded. The author of this dissertation conducted human correction for both algorithms. We show the editing time and the number of clicks needed to correct errors on different datasets in Figure 3.8, Figure 3.9, and Figure 3.10. Our method reduced the mean time for a human user to correct algorithmic errors by 50%, 83%, and 72% on Madcat, Germana, and Field, respectively.

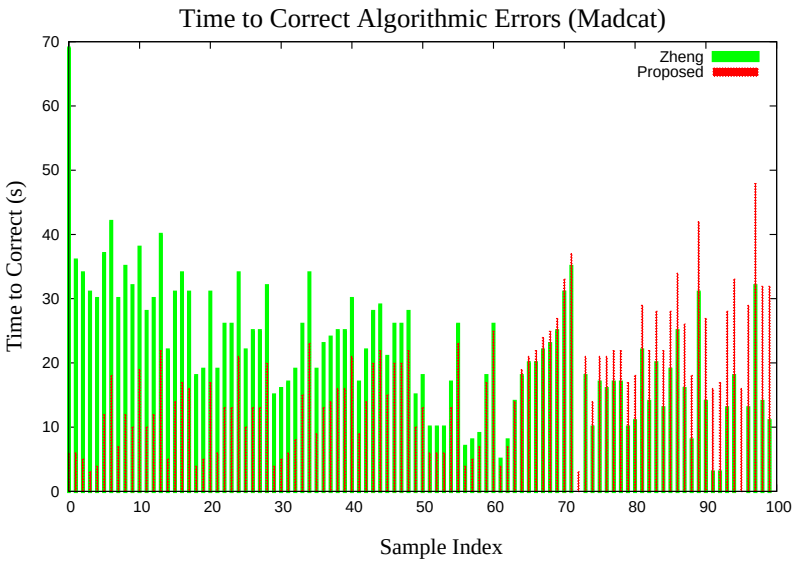
Table 3.3 shows error statistics of ruling attributes, and we observed some performance gains of our method. The gains are well correlated with the decrease of editing time. On Madcat, we know from Table 3.3 that both algorithms performed reasonably well, thus the editing time and the number of operations are close. However, on Germana and Field, the performance differences observed in Table 3.3 is reflected in Figure 3.9 and Figure 3.10. In general, we see that the Zheng et al.'s curves are above ours, which means their algorithm generated more errors than ours.

At a finer granularity, we can split error corrections into five categories: adjusting ruling line positions, skew angles, lengths, spacings, and the number of rulings. We plot the distributions of types of human editing in Figure 3.11, Figure 3.12, and Figure 3.13. On Madcat, Table 3.3 shows more errors by our algorithm for the starting position and the length, so there are more editing operations on Position and Length in Figure 3.11. For Zheng et al.'s algorithm, however, most editing operations are on Skew and Spacing. On Germana, their algorithm generated many spurious lines and missed several ones, so the error statistics in Table 3.3 are large,

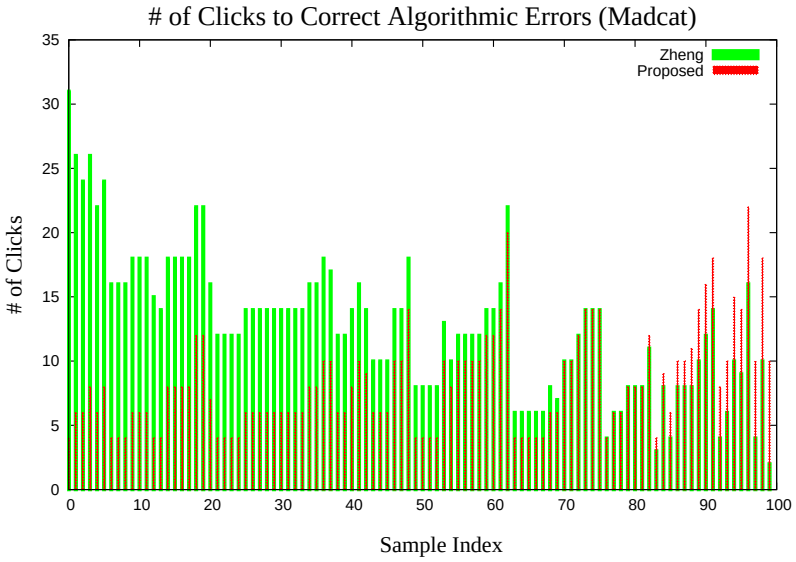
requiring more editing on Spacing and Count. On Field, neither algorithm was perfect in detecting the number of rulings. However, it is clear to see that our algorithm generated more accurate results than Zheng et al.'s algorithm because ours required much fewer human operations. As a quantitative metric, our algorithm required 3 operations on Madcat, 41 on Germana, and 103 on Field on Count, while Zheng et al.'s algorithm took 31 on Madcat, 707 on Germana, and 140 on Field.

3.7 Conclusion

In this chapter, we introduced a model-based ruling line detection algorithm that requires no supervised learning, but only estimation of some parameters on a few sample pages. We demonstrated the effectiveness of our algorithm by comparing it with former work in the literature on three datasets: Madcat, Germana and Field, and also used statistical metrics to show accuracy improvements. Our algorithm reduced the mean error of finding ruling lines from 0.1 to 0.03, 6.70 to 0.06, and 0.13 to 0.02 on these three datasets, compared to an HMM-based approach. For evaluation, we designed a measure that is closely related to the ruling line model so that it is simple to compute and intuitive to understand algorithms' performance differences. Using our proposed method, the average amount of time for a human user to correct algorithmic errors was reduced by 50%, 83%, and 72% on the three standard test datasets Madcat, Germana, and Field, respectively.

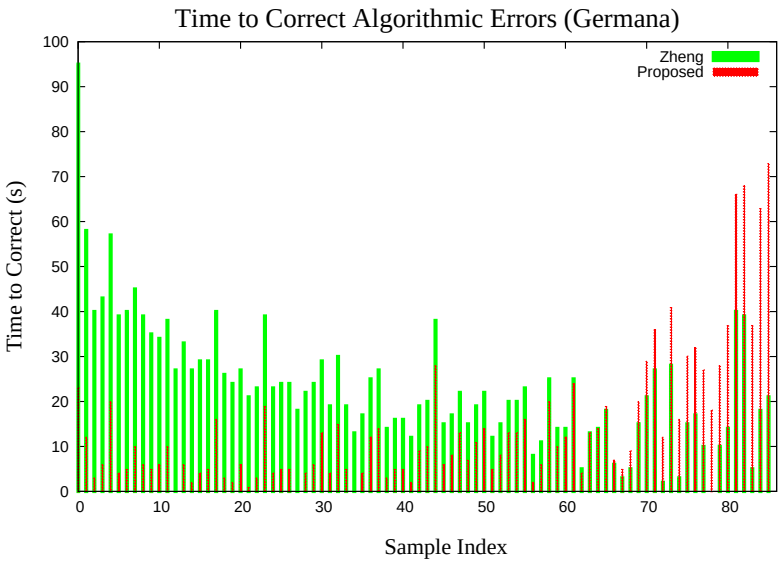


(a) Time to correct errors on Madcat.

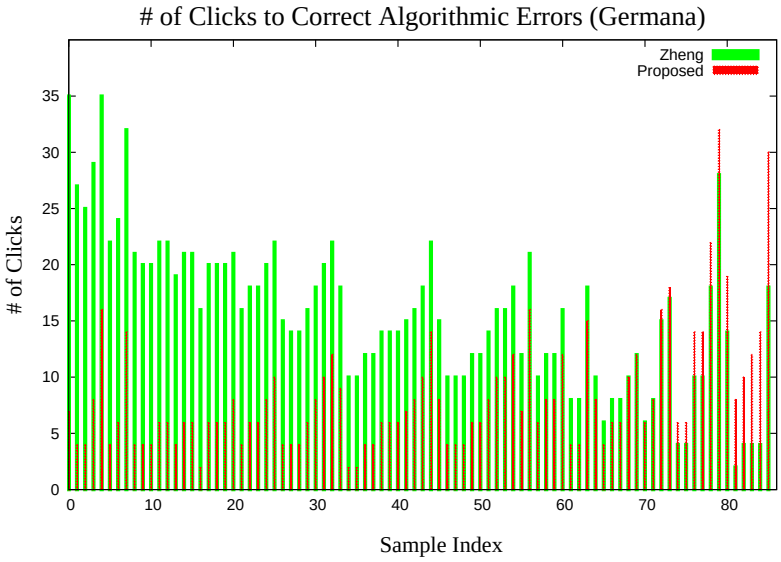


(b) Number of clicks needed on Madcat.

Figure 3.8: Measures of human effort on correction time.

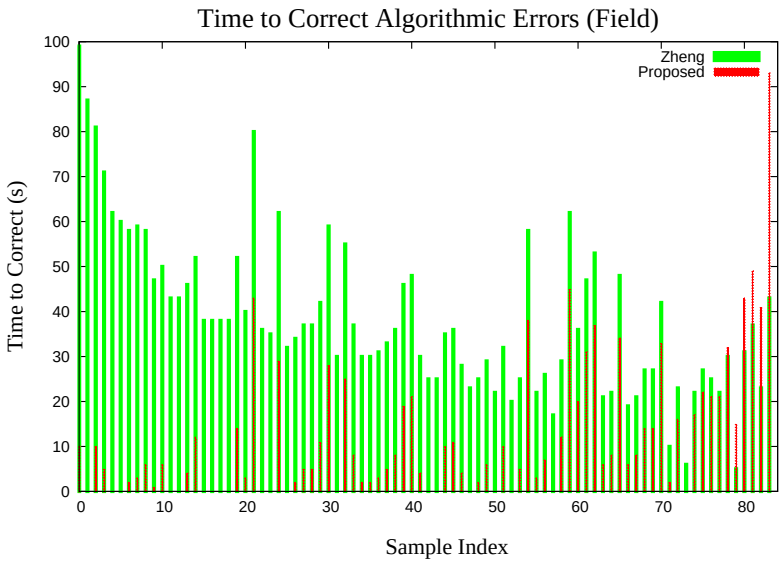


(a) Time to correct errors on Germana.

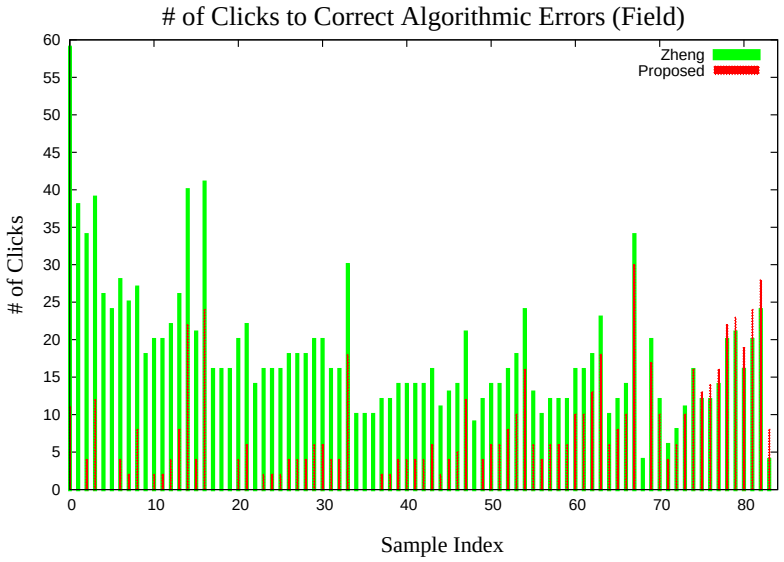


(b) Number of clicks needed on Germana.

Figure 3.9: Measures of human effort on correction clicks.



(a) Time to correct errors on Field.



(b) Number of clicks needed on Field.

Figure 3.10: Measures of human effort on correcting algorithmic errors.

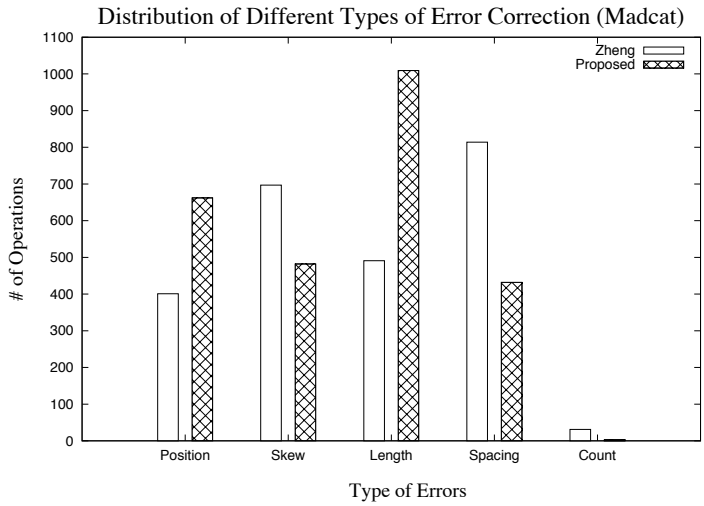


Figure 3.11: Distributions of human editing on Madcat.

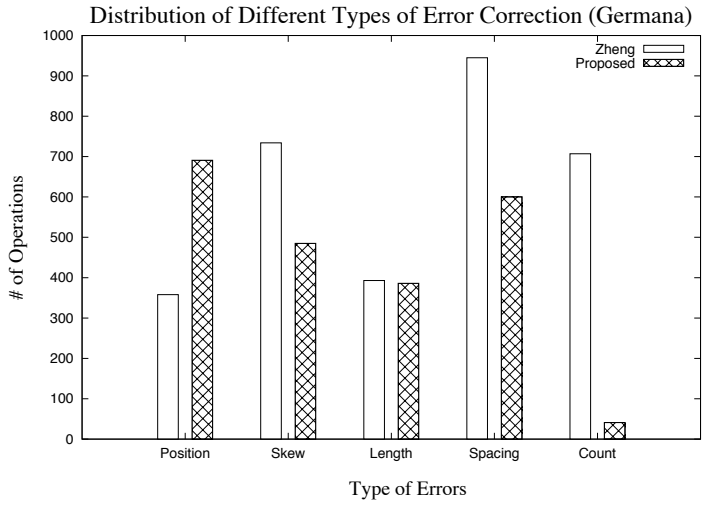


Figure 3.12: Distributions of human editing on Germana.

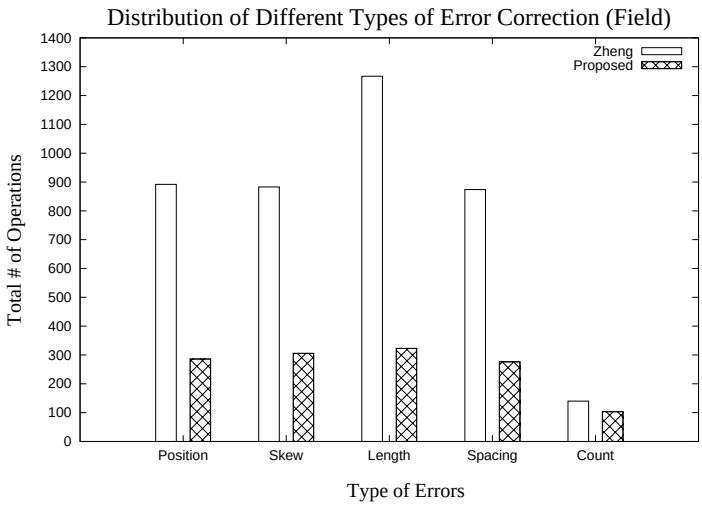


Figure 3.13: Distributions of human editing on Field.

Chapter 4

Ruling-based Tabular Structure Analysis

Pre-printed information is not restricted to ruling lines with regular spacing. For semi-structured documents such as tables and forms, it also includes pre-printed table stubs[247] and tabular structures where rulings are used to separate table/form content of various types from each other and from other document components.

4.1 Introduction

Although it seems straightforward, it turns out to be quite difficult to define a table.[84,154] This is especially true when trying to distinguish between tables and forms: many forms and tables share similar structure. Specifically, both tables and forms may contain tabular structures which consist of orthogonal ruling lines.[85,215]

We adopt one working definition that aligns with our composite-model framework because it reveals different timings when information is added into the final

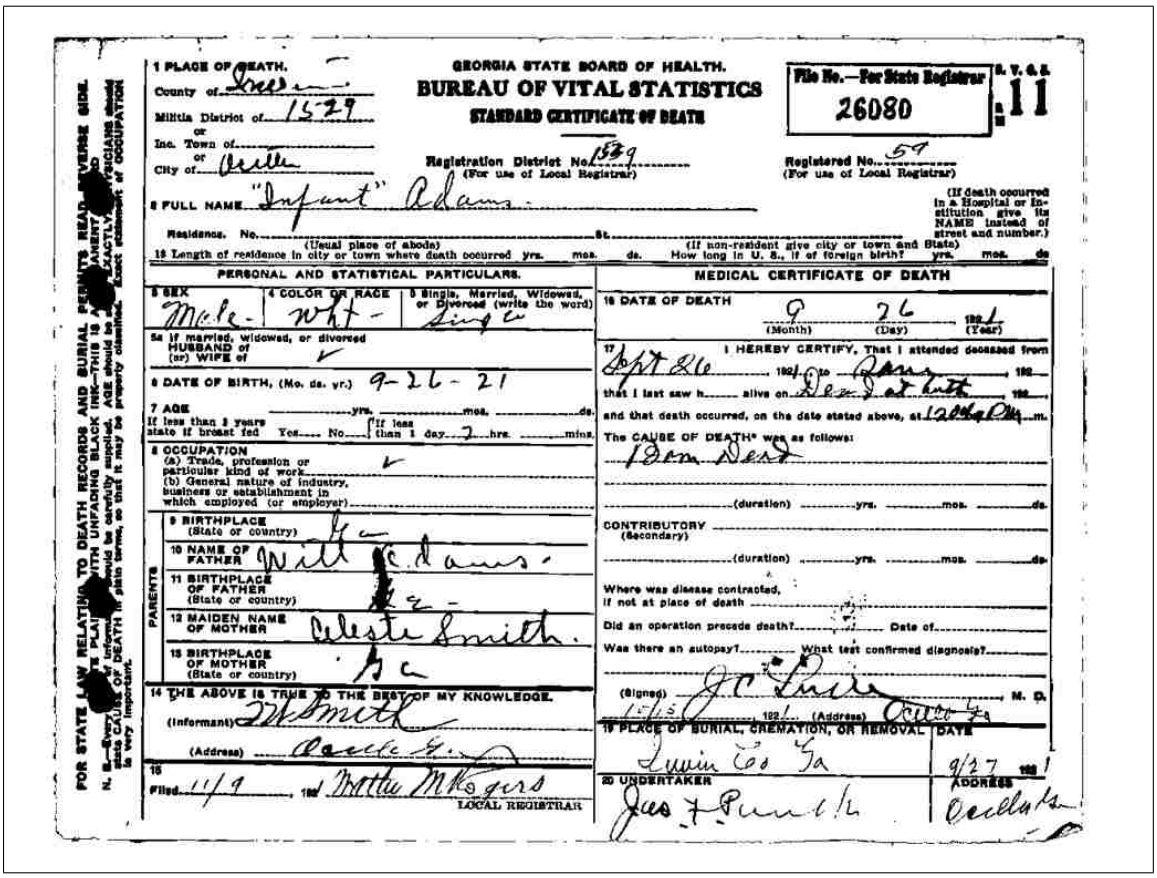


Figure 4.1: A sample document illustrates multiple challenges similar to those present in the evaluation dataset.

document image: tables are designed to *display* information in that the table header and the content are added at the same time, e.g., via pre-printed text or handwriting; forms are designed to *collect* information in that the form header and other text are pre-printed on the paper and later people fill in the form with their handwriting. Table examples include the periodic table, multiplication tables, etc. Form examples include census forms, tax forms, death certificates, etc. In this Chapter, we focus on handwritten forms that exhibit tabular structures.

As one way of organizing relational data, tabular structures have *physical* and

logical structure.[95,96,247] Physical structure describes the locations of tabular components, e.g., headers, rows, columns, cells, rulings. Logical structure refers to the way of connecting tabular components to each other to form a set of relational n-tuples.[263] Note that tabular components may belong to both physical and logical structure from different perspectives. For instance, a table cell can be defined in logical structure by $(Row[i], Column[j])$, and it can also be defined in the physical structure as a rectangular region of pixels in the document image.

Our work is initiated by the Multilingual Automatic Document Classification and Translation (MADCAT) project that aims to categorize a collection of Kurdish handwritten documents that were preserved during the Anfal uprising.[169,170] Thus, the target handwritten document corpus presents characteristics that are not available from controlled environments like labs. Our dataset contains multiple components: machine-printed text, handwriting, pre-printed ruling lines, signatures, logos, etc. In addition, paper conditions and digitization characteristics are different from those in a controlled environment. For example, paper may be ripped, folded, and punched, and for scanned documents, we usually observe skewed pages and low image quality with plenty of "salt-and-pepper" noise, etc. Our evaluation dataset is a collection of Arabic handwritten documents containing forms. Since these forms contain sensitive content such as biographic information, we demonstrate a handwritten death certificate instead[177] in Figure 4.1 to exemplify several kinds of challenges present in the evaluation dataset. In addition, our evaluation images also contain clutter noise around the border of the page, pre-printed tabular structure, pre-printed text, free-form handwriting that may flow beyond the expected white space, and heavy scanning noise across the page. Also, due to the

age of paper and the low quality of scanning, text strokes and pre-printed ruling lines are often broken.

4.2 Related Work

Because of the resemblance of tables and forms, techniques of recognizing their tabular structures are shared in the literature. Hu et al. summarized the problem of tabular structure analysis as two sub-problems: detection and recognition.[109] Laurentini and Viada used horizontal and vertical rulings as initial evidence for tabular structures in machine-printed documents, and then used several tests to exclude non-tabular areas.[136] Hu et al. introduced a tabular structure detection method that does not require ruling lines and can work on machine-printed document images and ASCII files.[109] After segmenting text into lines, they detected the *inside-space* between adjacent text blobs and then used a dynamic programming (DP) for optimally decomposing the entire page into text lines and tabular rows. Shamalian et al. proposed a method that uses pre-defined tabular structure as a complementary input.[215] Using an *ink template*, they searched for a best match of ink templates and the segmented text lines. Shafait and Smith extended tabular structure detection to multi-column documents.[214] However, existing techniques[109,136] assume clean/simple input and/or well-segmented text lines. Our previous attempt[55] showed that substantial pre-processing such as noise elimination and segmentation of text lines and words was required before applying these techniques. For complicated handwritten documents, however, these are not trivial tasks.

Tabular structure recognition usually assumes identified tabular regions and

the goal is to find the physical structure and the logical structure of the tabular model.[95,96,247] There has been plenty of work on machine-printed tabular structure recognition.[50,102,136] Gatos et al. [88] made use of the complete tabular rulings to recognize the tabular structure while Hirayama[102] relied on dynamic programming to align tabular columns. Richarz et al. proposed a method of tabular structure recognition for their semi-supervised transcription system for handwritten historical weather reports.[202] Making use of the pre-printed tabular structure, they used the Hough transform to detect the horizontal and vertical rulings that constitute the tabular structure. Clawson et al. presented a projection-profile based method to detect and extracted handwritten fields from historical census forms.[66] We notice that these existing techniques are evaluated on datasets where rulings are usually salient and sole, meaning no other lines will distract the algorithms for table analysis. This is, however, not the case for our dataset where we need to handle severely broken lines and/or misleadingly alignments of foreground pixels.

Nagy conducted a parallel study on the same dataset as ours where he focused on finding the invariant representation for orthogonal ruling lines that constitute a tabular structure.[177] The idea was to find orthogonal ruling lines from the Hough transform that might represent the tabular structure, by examining the ruling gap ratio that was considered to be invariant to translation, rotation, and scaling. Similar to our proposed approach, this procedure requires the tabular structure template as an input.

In this work, we try to address the problem of form detection and decomposition on noisy handwritten documents. These documents differ from the ones in the literature that they are not collected in a controlled environment but from the field.

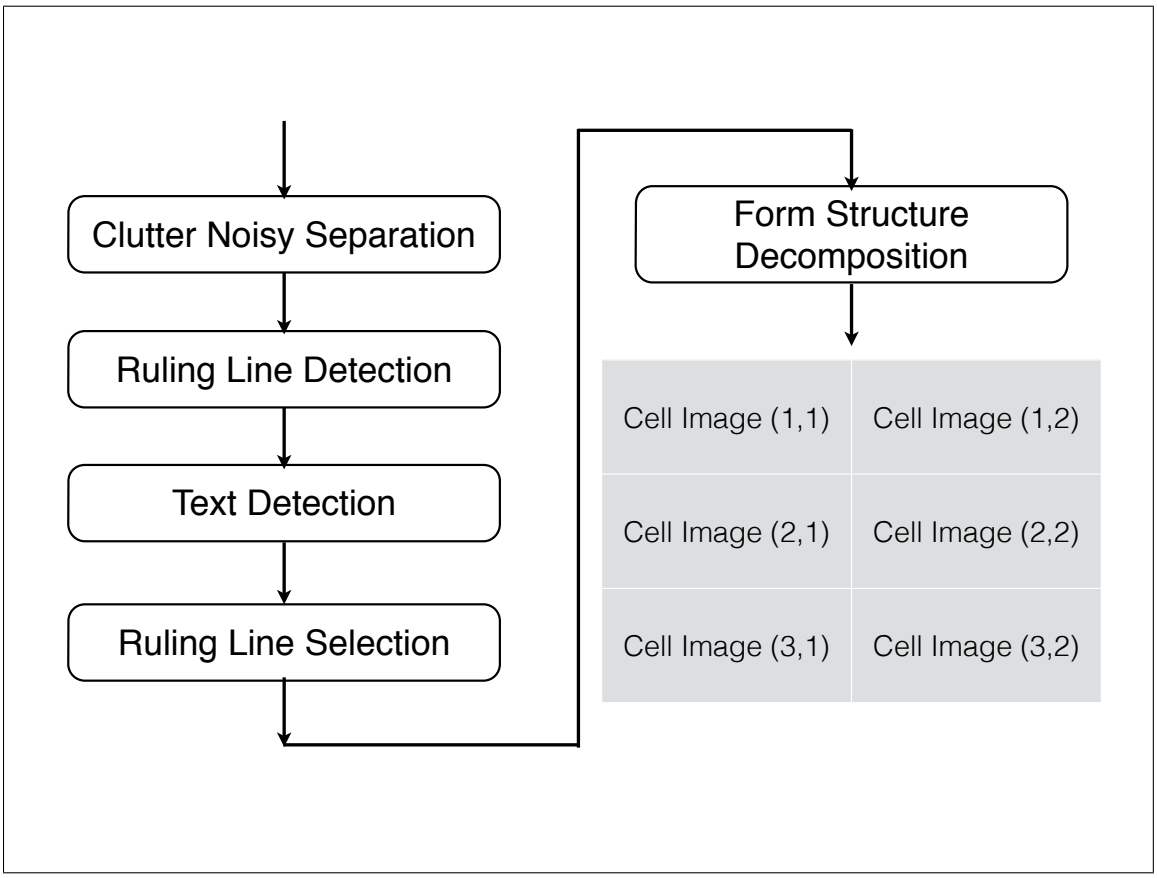


Figure 4.2: The workflow of our form analysis system.

As a result, various types of noise and artifacts make ruling lines hard to detect. After separating clutter noise from the image, our idea is to ensure high recall of tabular rulings and then compute the "key points" that intersect horizontal and vertical rulings. Then we use an optimization procedure to select the most probable subset of rulings that constitute the form. Finally, given the selected key points, we decompose a tabular structure into a 2-D arrangement of cell images.

4.3 System Overview

The high-level workflow is shown in Figure 4.2. In our implementation, an image is modeled as a multi-layer structure where each layer consists of one document component, e.g., scanning noise, ruling lines, machine-printed text, handwriting, etc. In addition, we observe that the same type of form is used multiple times in the dataset. Given this information, we build a form template that summarizes the logical and physical structure and use it to register an unknown input form in a corpus.

The form template specifies the number of rows and columns, row/column spanning, and approximate cell dimensions. Figure 4.3 shows a snapshot of such a form template file. Note that since the same form template is used many times in the dataset, the cell dimension information (heights, widths) in the form template file records only the average values. Thus, the form template serves as another input to help detect and recognize forms.

4.3.1 Clutter Detection

Clutter noise, which refers to the black margin near the image border, is usually introduced by scanning against a black or nonreflective background. Agrawal and Doermann presented a distance transform based approach to detect and remove clutter noise.[6] They detected it using a 2-class SVM classifier with a number of connected-component based features.

Our clutter detection, however, is based on the fact that clutter noise is usually much larger than the other components. Thus, at lower resolutions, we may see only

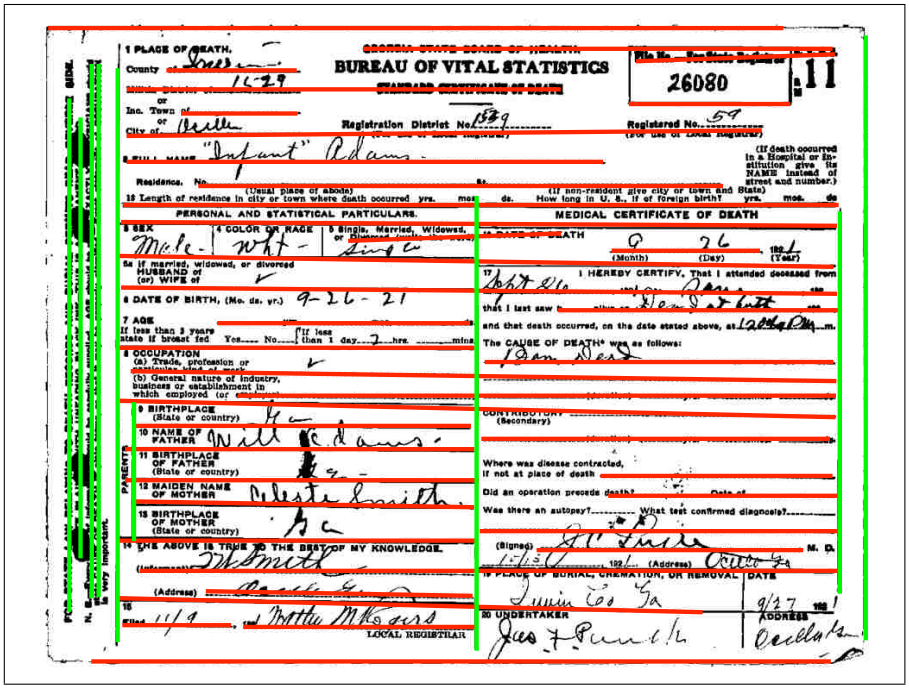
```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
</meta>
</head>
<body>

<table border="1" height="745" width="990" pagesize="fixed" numrows="fixed" modelid="003-1"
docid="40137-0116-00620216">
<tr>
<td row="1" col="1" tokens="4" isheader="true" height="105" width="620" rowspan="2"    ></td>
<td row="1" col="2" tokens="1" isheader="true" height="40" width="370" colspan="2"   ></td>
</tr>
<tr>
<td row="2" col="2" tokens="3" isheader="true" height="65" width="235"   ></td>
<td row="2" col="3" tokens="2" isheader="true" height="65" width="135"   ></td>
</tr>
<tr>
<td row="3" col="1" tokens="9, 10, 11, 12, 13, 14"  height="640" width="620" rowspan="2"    ></td>
<td row="3" col="2" tokens="8"  height="550" width="235"   degraded="true"></td>
<td row="3" col="3" tokens="5, 6, 7"  height="550" width="135"   degraded="true"></td>
</tr>
<tr>
<td row="4" col="2" tokens="16"  height="90" width="235"   degraded="true"></td>
<td row="4" col="3" tokens="15"  height="90" width="135"   degraded="true"></td>
</tr>
</table>
</body>
</html>
```

Figure 4.3: An example of the form template specification.

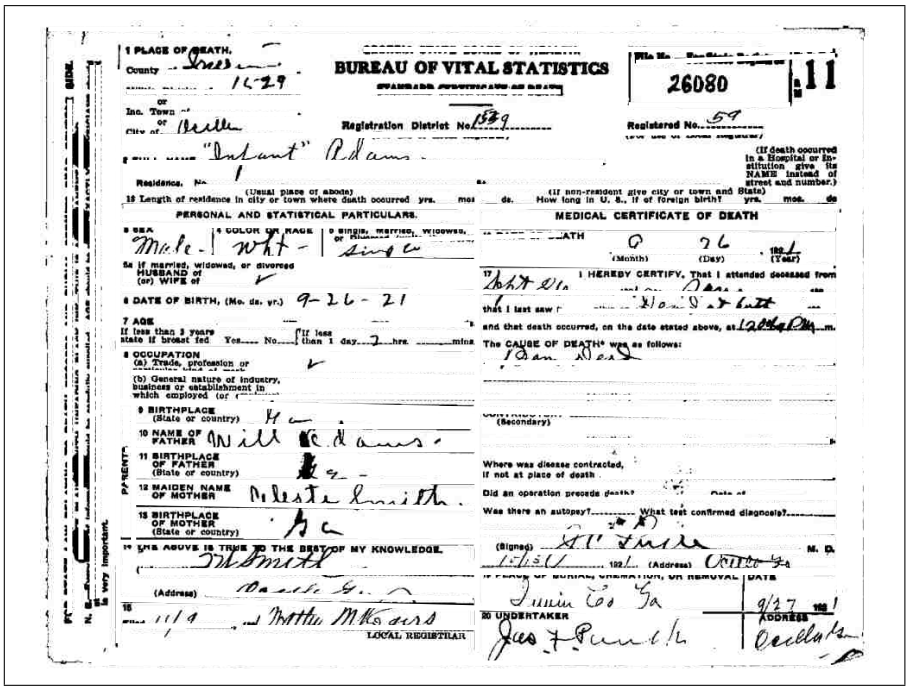
clutter noise. We scale the image down to 1/4 and extract connected components from the scaled image. As expected, large components are mostly clutter noise in the original image, then we mark these clutter pixels in another layer so that they will not be considered in the following processing operations.

4.3.2 Ruling Detection

Although many of the ruling lines in our evaluation dataset are pre-printed, they differ from the ones addressed in Chapter 3 in that they may not have consistent spacing. To detect salient ruling lines, we still use a probabilistic variant of the Hough transform.[165] Since many rulings are broken, small gaps (20 pixels, learned from a development dataset) are allowed during ruling detection. Next, we make use



(a) Detected ruling lines.



(b) An image layer containing primarily text blobs.

Figure 4.4: Detection and separation of various document components.

of the fact that most correct rulings are parallel or orthogonal in order to exclude part of spurious ruling lines detected in the text area. Then, we use the Adaptive Basic Sequential Algorithmic Scheme (Adaptive BSAS,[233] details in Chapter 3) to group clustered line segments based on their ρ values, and compute their parameters (slope, intercept, etc.) using the standard line fitting.

Due to the low image quality, we adjust the parameters in Hough transform to ensure high recall of line segments, which, of course, may give rise to spurious ruling lines in tight text regions. For example, the tolerated gap size between line segments is set to be 1/2 of the average horizontal inter-word spacing estimated by connected-component analysis. Most spurious ruling lines will be addressed during the follow-up processing. Figure 4.4(a) shows the results with horizontal and vertical rulings, marked in red and green, respectively.

4.3.3 Text Detection

Spatial displacement of text can be valuable information to exploit for tabular structure detection and recognition.[54,109] In our current work, text blobs are detected by separating the layers of rulings and clutter noise and then detecting connected components. Next, we transform these text blobs based on the skew angle and then exclude those having unexpected aspect ratios ($\alpha < 0.1$ or $\alpha > 10.0$) of their bounding boxes. This effectively excludes most of the line segments left in the current image layer, as shown in Figure 4.4(b).

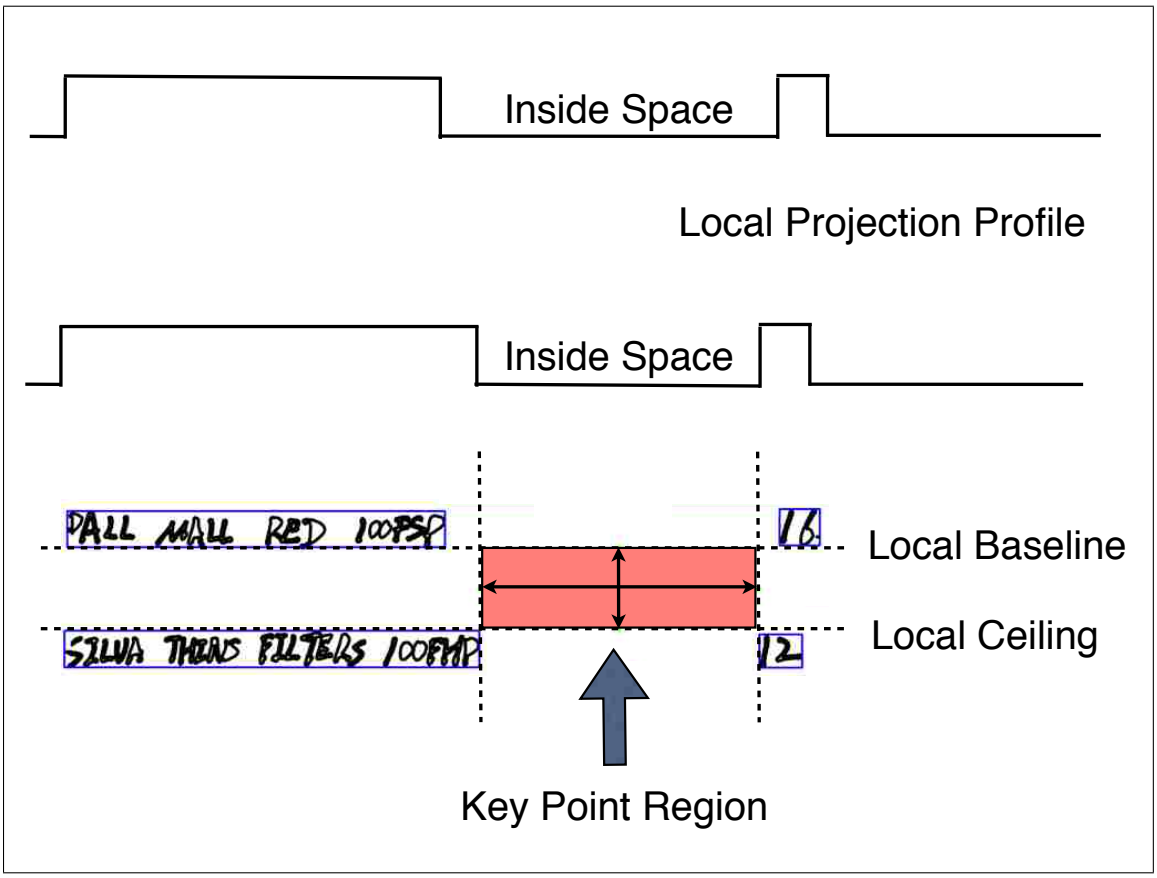


Figure 4.5: An illustration of key points for tabular structures.

4.4 Ruling Selection

At this stage, we assume all the lines are parallel or orthogonal since outliers are filtered out during ruling detection (Section 4.3.2). We echo the idea of *key points*[56] as follows.

Key points are defined to be a white space region within a local 2×2 array of text blobs in which any horizontal or vertical cuts will not affect form cells. An example of a key point is shown in Figure 4.5. For unruled forms, key points usually refer to the white space between text blobs. For ruled forms that have row/column

spans, however, we need to adapt key points as the intersections of form rulings. Therefore, the problem of ruling selection is converted into key point selection. The advantage is that this conversion also reduces the searching space because not all detected ruling lines intersect another.

As a result, several rulings in the text area isolated from the form header are excluded at this stage, as shown in Figure 4.6(a). Note that we have allowed relatively large gaps (100 pixels, learned from a development dataset) between rulings due to degraded images in order to obtain high recall of key points in tabular structures.

Then, we use the Adaptive BSAS clustering to group key points into horizontal and vertical clusters. Now, each key point is indexed by the horizontal and vertical clusters, e.g., $(H_{1,\ldots p}, V_{1,\ldots q})$, where p, q are the numbers of horizontal and vertical key point clusters, respectively.

Next, we formulate key point selection as an optimization problem:

$$argmin_{W \in \Omega} f(W, M) \tag{4.1}$$

where W is a configuration of selected key points for scoring, Ω is the set of all configurations, and M is the form template containing rows r, columns c and cell dimension information. We optimize the key points of the form structure by selecting horizontal and vertical key point clusters separately. Using the horizontal case as an example, now the formulation is specified as follows:

$$argmin_{W_h \in \Omega_h} f(W_h, M_h) \tag{4.2}$$

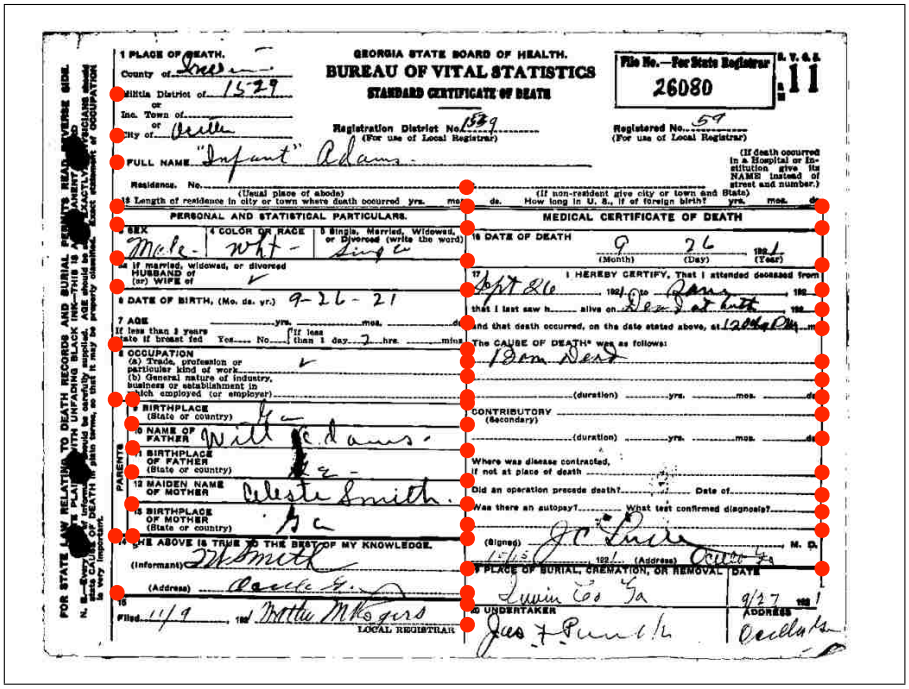
with the constraints that $\|W_h\| = r + 1$ and $\|\Omega_h\| = \binom{p}{r+1}$. The vertical key point

selection is treated similarly. The cost function $f(\cdot)$ computes a real value indicating how close the current key point cluster configuration matches with the form template. In our implementation, we compute the difference between the distance between adjacent clusters and the spacings between form rows or columns:

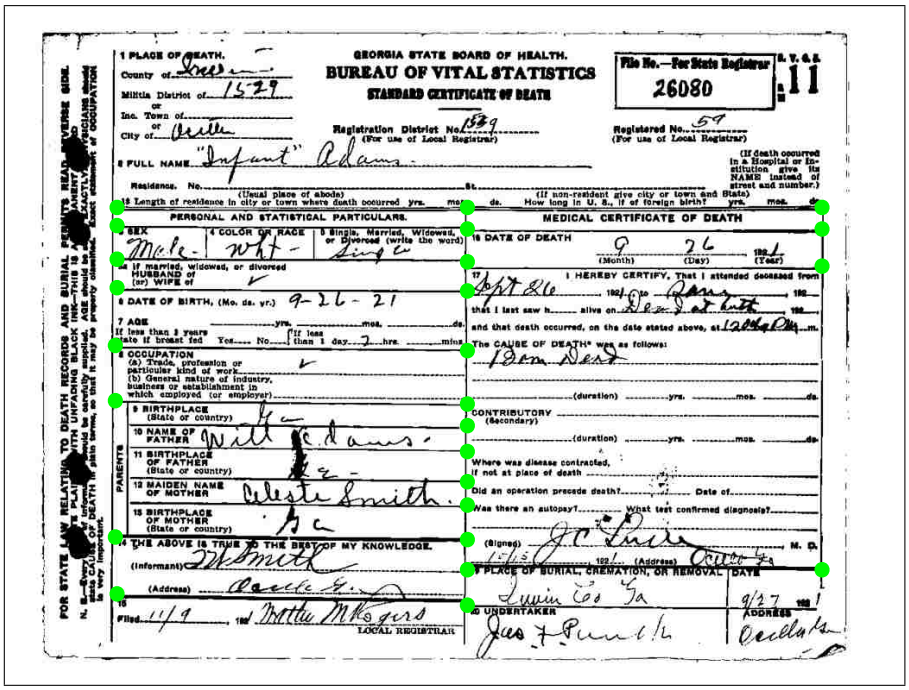
$$\begin{aligned} f(\cdot) &= \|W_h - M_h\| + \mathcal{C}(W_h) \\ &= \sum_{i=1}^{p} \|W_{hi} - M_{hi}\| + \mathcal{C}(W_h) \end{aligned} \tag{4.3}$$

where W_{hi} and M_{hi} are heights for form row i. $\mathcal{C}(W_h)$ is the cost for text displacement against the horizontal rulings. Currently, we only consider the text displacement in the form header, i.e., the accumulated cost of bounding boxes of text blobs adjacent to the second horizontal ruling.

Since p, q are small numbers, it is feasible to enumerate the configuration space for the global optimal configuration that constitutes the form structure. Figure 4.6(b) shows the result of selecting horizontal key point clusters, marked in green dots. Similarly, we run the selection algorithm again using the corresponding vertical key point clusters. Note that for forms with opened sides (left and right in Figure 4.6(a)) we need to add imaginary key points to comply with the constraints in Equation 4.2. Finally, we scan through the obtained key point grid to decompose the tabular structure into a 2-D arrangement of form cells. To handle row/column spanning, we simply skip corresponding adjacent key points vertically/horizontally.



(a) Detected key points.



(b) Key points for the form.

Figure 4.6: Intermediate results from the ruling selection algorithm.

4.5 Experimental Setup

4.5.1 Data Preparation

Our evaluation involved a noisy Arabic handwritten document dataset that contains secret police files of March 1991 uprisings in northern Iraq.[169,170] These files, which contain evidence of the Anfal genocide in Iraqi Kurdistan during the 1980s, have been available to the academic community with restrictions since 1998.[170] Recently, the Linguistic Data Consortium (LDC)[1] prepared part of this corpus as one evaluation dataset for the Multilingual Automatic Document Classification and Translation (MADCAT) project.[3] Note that although this dataset belongs to the same research project MADCAT, it is not the one used for pre-printed ruling lines in the other chapters of this dissertation.

Since this corpus is not prepared in a controlled environment, it includes noise, multiple artifacts, and complicated document layouts. A sample document presenting some of the challenges is shown in Figure 4.1. In total, we have annotated and evaluated 61 Arabic documents from 16 form templates. Table 4.1 shows some statistics in this dataset based on manual investigation. These statistics reflect the variations in and complexity of the tabular structures.

4.5.2 Evaluation

We evaluated the system by computing *precision* and *recall* on form cell images:

$$\begin{aligned} \text{precision} &= \frac{\text{number of correctly detected form cells}}{\text{total number of detected form cells images}} \\ \text{recall} &= \frac{\text{number of correctly detected form cells}}{\text{total number of ground-truthed form cells}} \end{aligned} \tag{4.4}$$

Table 4.1: Statistics of human perception on the Arabic handwritten form documents.

Page-wise Document Characteristics	Max	Median	Min
# of image/model	29	6	1
# of vertical rulings	27	5	3
# of open-ended solid horizontal rulings	27	4	0
# of open-ended dashed horizontal rulings	24	0	0
# of close-ended horizontal rulings	12	0	0
# of headers containing printed Arabic phrases	27	10	4
# of other printed Arabic phrases (e.g., from title)	89	14	0

Currently, we considered a form cell detection to be correct as long as the detected 2×2 tuple of key points corresponds to the four vertices of *any* ground-truthed form cells. In other words, the logical mapping of form cells was not yet evaluated directly, e.g., a missing form row will not render all the form cells below that row wrong.

4.5.3 Comparison

We compared our algorithm with a cross matrix based method proposed by Shi, et al.[216] Their idea was to first compute a 2-D matrix of rulings which intersect orthogonal rulings, for both the form template and input image (Model[·][·] and Scene[·][·]). Next, considering a relatively small number of vertical rulings, they enumerated all possible combinations of vertical rulings. In each enumeration, they applied a dynamic programming (DP) algorithm to select the optimal horizontal rulings. In this way, they converted the problem of selecting an optimal subset of rulings to optimally aligning two sets of rulings, which is similar to the edit distance computation between two strings. Finally, the horizontal rulings were computed by

back tracking the score matrix in the DP framework. In essence, their approach and ours used similar logic to select horizontal/vertical rulings as the tabular structure, but our method made use of cell information in the form template, thus is expected to be more accurate than the cross matrix based approach.

We restate the DP framework here for reference. Let $M_i, i \in [1, n]$ denote horizontal model rulings, where n is the number of horizontal rulings in the form template. Likewise, scene rulings are represented by $S_j, j \in [1, N]$ where N is the number detected in the image. The matching score of a model ruling and a scene ruling is defined as the Hamming distance between two rows in the cross matrices.

$$\mathcal{C}(M_i, S_j) = \sum_{k=1}^{K} \|\text{Model}[i][k] - \text{Scene}[j][k]\| \tag{4.5}$$

where $k \in [1, \ldots, K]$ denote the index to the current vertical ruling configuration.

The definition for the score matrix H resembles the computation of edit distance, as follows:

$$H[i][j] = min \begin{cases} H[i-1][j-1] + \mathcal{C}_{sub}(M_i, S_j) \\ H[i-1][j] + \mathcal{C}_{del}(M_i) \\ H[i][j-1] + \mathcal{C}_{ins}(S_j) \end{cases} \tag{4.6}$$

where $i \in [1, n]$, $j \in [1, N]$, and $\mathcal{C}_{del}(\cdot)$, $\mathcal{C}_{ins}(\cdot)$ are the costs for deletion and insertion in computing the edit distance between two strings. In the tasks of ruling selection, we need to virtually forbid deletion by assigning it a high cost, otherwise deletion may cause a smaller number of rulings to be selected after the alignment. Thus, we set $\mathcal{C}_{del}(\cdot) = 100$ and $\mathcal{C}_{ins}(j) = \mathcal{C}(0, S_j)$. After computing the score matrix, we tracked back from $H[n][N]$ to obtain the alignment of model and scene rulings, in

which the positions of substitution were recorded as the indices of selected rulings.

4.6 Experimental Results

We evaluated our algorithm using 61 documents from 16 form templates containing 3,627 form cells and found out that the precision is 88.60% and the recall is 87.90%. On the other hand, the cross matrix based method obtained precision and recall of 85.90% and 84.20%, respectively. The performance gains were statistically significant according to the McNemar test[72] (Section 1.7) with a confidence level of 95%. A complete list of form templates present in the Arabic handwritten dataset are rendered in Figure 4.7. Note that the detected key points do not have to form a complete 2-D grid since we can compute the rest based on the form template.

Our method did outperform the cross matrix based method; however, we made several observations that may be useful for further improvement. First, spacing between adjacent rulings has proven to be characteristic to exploit for selecting the correct rulings. Second, the cross matrix method assumes less input information but relies on high recall of line segments, which is challenging in such a degraded dataset. If the part of the ruling where an intersection is expected is missing, there will be a cost in the cross matrix. Lastly, the spatial displacement of text blobs can be useful, especially pre-printed text in form headers, because it is usually well spaced within the adjacent ruling lines. On the other hand, user-added handwriting becomes a hindrance because it may lie on ruling lines and/or in arbitrary writing directions.

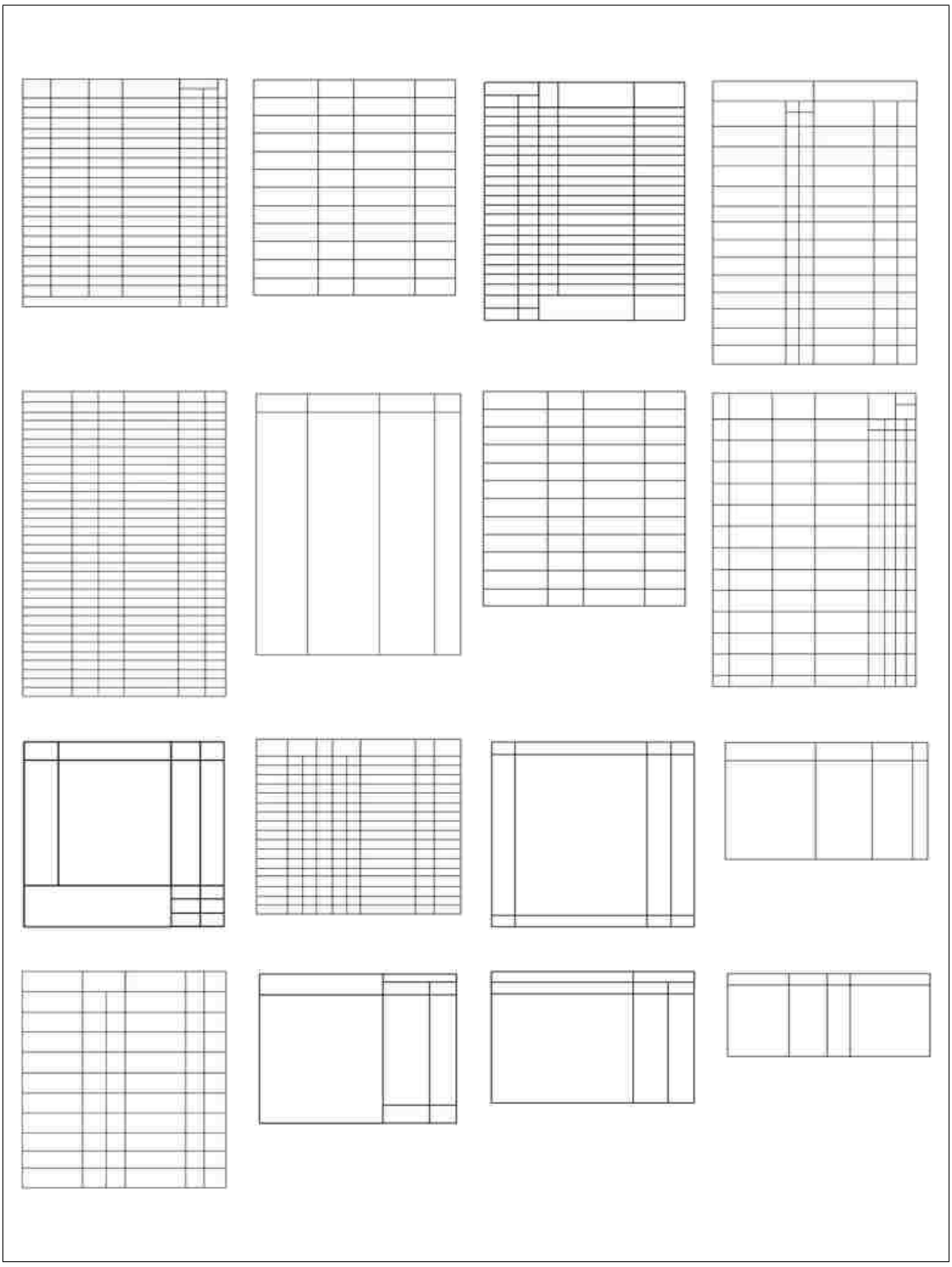


Figure 4.7: All form templates present in our evaluation dataset.

4.7 Conclusions

Ruled forms are another example of mixture of pre-printed information, i.e., form ruling lines, and user-added data, i.e., handwritten text. In this chapter, we have shown the complexity of processing handwritten documents that were not prepared specifically for DIA research. These documents usually contain multiple kinds of noise and artifacts, and also show complex layouts which include pre-printed text, user-added handwriting, forms, ruling lines, etc. Therefore, it is more challenging than analyzing handwritten form datasets prepared in research labs. Experimental results on 61 pages from 16 form templates showed a cell precision of 88.60% and a recall of 87.90%, which was 2.7% and 3.7% of error deduction in precision and recall, compared to an existing approach.

Chapter 5

Writer Identification in Composite-model Analysis

In Section 1.4, we discussed possible negative effects of removing ruling lines in writer identification, and also introduced a model-base approach to detect ruling lines within a page that minimizes the Least Square Error in Chapter 3. In this chapter, we revisit the problem of writer identification using the proposed composite-model framework and demonstrate the potential benefits of ensuring image integrity. Specifically, this means separating three components of the composite-model framework: pre-printed information, user-added data, and digitization characteristics, and drawing connections between these components.

5.1 Overview

Traditional document analysis consists of a pipeline of processing stages where each stage makes assumptions about the nature of the input image. These assumptions

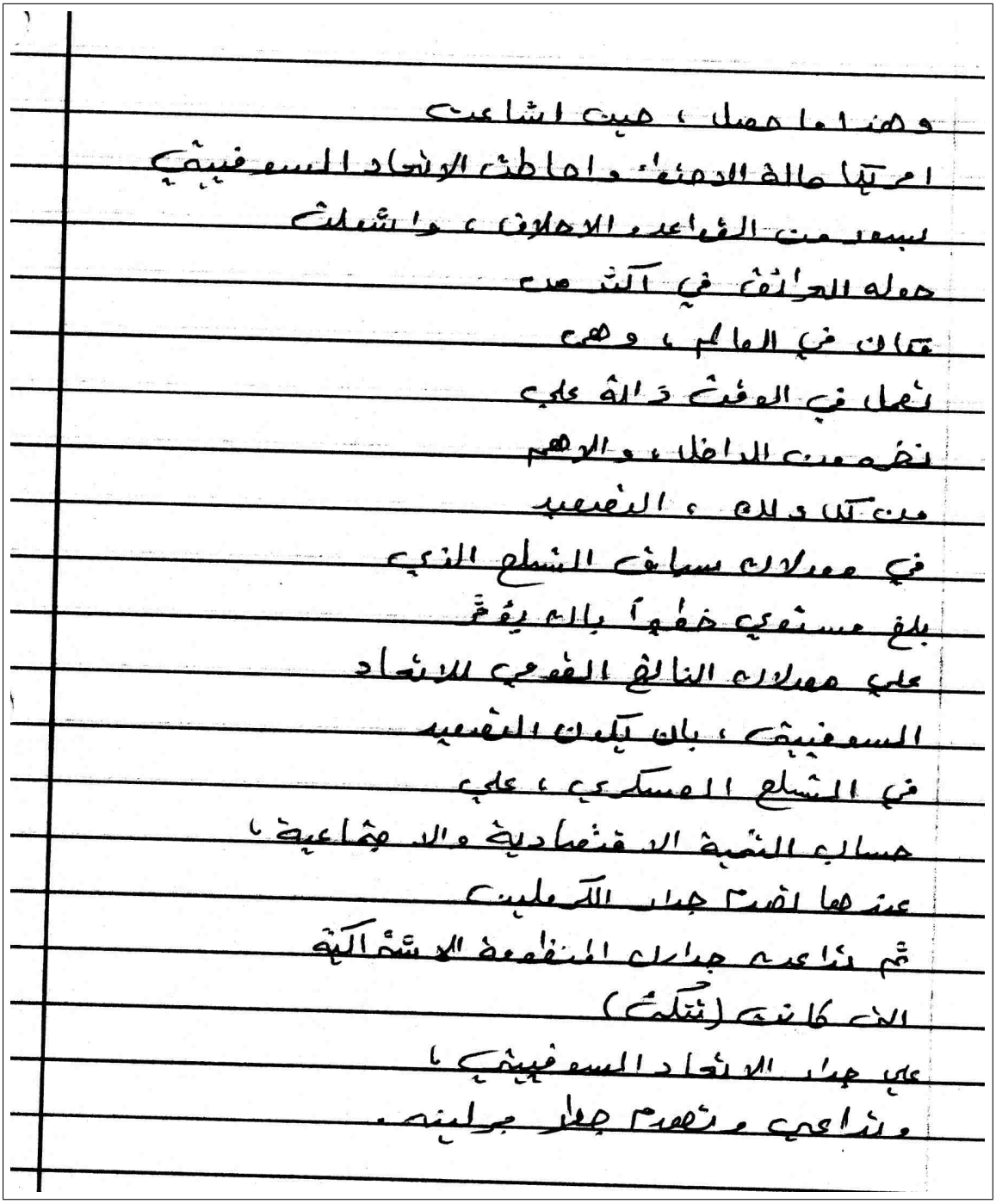


Figure 5.1: An Arabic document with pre-printed ruling lines.

amount to pre-conditions that must be satisfied before the procedure in question can function properly. For example, a layout analysis module may be designed to operate under the assumption that the input page image contains no skew. Or an OCR module only operates on segmented text lines, words, or characters which are extracted by text detection and segmentation. Usually these pre-conditions are satisfied by another module in the pipeline that detects violations of the assumption (e.g., the presence of skew due to the page being scanned at an angle) and corrects for it (e.g., by rotating the bitmap image to counteract the input skew). Traditionally, such detection and correction procedures are included in pre-processing. As a result, the input bitmap is modified irreversibly as it is passed along the document processing pipeline (see Section 1.3 for more details). While this attempt of normalizing the input may make it easier to design a particular step, important information that could be useful to later stages may be lost along the way.

We examine the DIA processing methodology in a different way, as shown in Figure 1.5. Specifically, we attempt to identify pre-printed information, artifacts, and digitization characteristics and pass their attributes along the pipeline, so that later stages such as feature extraction can counteract them or even make use of them. In this Chapter, we deal with one type of pre-printed information, ruling lines, and their correlated scanning effects, page skew, in the task of writer identification. Our philosophy is to virtually counteract page skew while reserving image integrity. Also, we investigate whether people's handwriting is changed because of pre-printed ruling lines and how to exploit such changes.

Pre-printed ruling lines have recently stimulated interest in handwritten document image analysis.[5, 21, 44, 46] Figure 5.1 shows a noisy handwritten document with

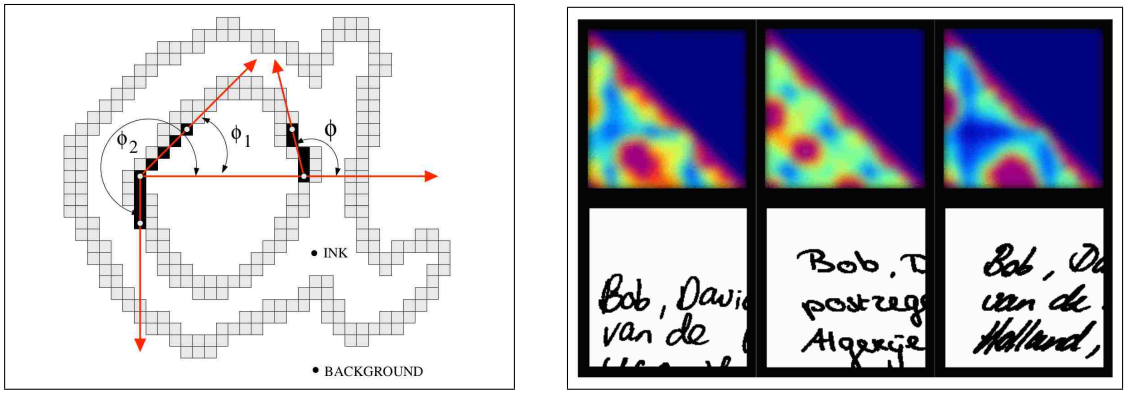
pre-printed ruling lines. Our discussion in Section 1.4 shows that it is not always necessary to remove ruling lines in handwritten documents, because it either requires complicated heuristics rules[21,46] or carefully designed classification[45,135] to recover broken handwritten strokes. These observations motivate us to study the effects of pre-printed ruling lines on writer identification. This is also an example of investigating interaction between pre-printed information and user-added data.

5.2 Writer Identification Modules

Our composite-model framework suggests that we should not alter the original input image, but rather extract useful information at each stage in the processing pipeline and pass on the information for use in later stages. In this section, we describe the building blocks in our analysis of writer identification.

5.2.1 Ruling Line Detection

Our model-based ruling line detection algorithm takes advantage of the facts that rulings have consistent spacing $\boldsymbol{\beta_1}$ and approximately the same length $\mathcal{L}$, skew angle $\boldsymbol{\beta_2}$, and thickness $\mathcal{H}$.[54] One advantage is that these constraints guarantee a globally optimal solution under the Least Squares Error (LSE). Technical details are described in Chapter 3. Next, for feature extraction, these pre-printed rulings are represented as lists of pixel sequences.



(a) Contour-hinge feature extraction.

(b) Various handwriting samples and their corresponding contour-hinge features.

Figure 5.2: An illustration of computing contour-hinge features.

5.2.2 Feature Extraction

Contour-hinge features are shown to be useful statistical features for writer identification.[40] An illustration of this feature extraction procedure is shown in Figure 5.2[1].

The feature computation is based on contours extracted from connected-component analysis. Specifically, for each pair of adjacent segments (each 5 pixels long) along the contours, we compute their angles from the horizontal axis and treat them as two random variables ϕ_1 and ϕ_2 in Figure 5.2(a). Quantizing the angle plane ($[0, 2\pi)$) into 24 bins ($2n$, $n = 12$), we accumulate the votes in each bin as we traverse all contours. Because of the assumed symmetry in the histogram, only half of the bins ($\phi_2 \geq \phi_1$) are used to compute a probabilistic distribution function (PDF). Thus, the finalized feature vector is 300-dimensional ($C_{2n}^2 + 2n$=300).

[1]Diagrams of contour-hinge features are courtesy of Dr. Lambert Schomaker and Dr. Marius Bulacu.

5.2.3 Writer Identification

We use Support Vector Machines (SVMs) for writer identification. SVMs construct a hyperplane with maximum margin in higher dimensional vector space, where a non-linearly separable classification problem in the original vector space may become linearly separable after projecting these feature vectors into higher dimensional space by different mapping functions. The mapping functions are called *kernels* in the literature.

In our experiments, we use the libSVM tool.[51] We use the Radial Basis Function (RBF) kernel because it offers better discrimination than the linear kernel, while using fewer parameters than the polynomial kernel. In our experiments, we set the cost $c = 10000$ based on an independent development set that contain three documents per writer, and then we normalize feature vectors into the unit hypercube. In addition, we use the probabilistic option in libSVM in order to compute the "Top-N" lists for writer identification.

5.3 A Composite-model Approach

5.3.1 Handling Ruling Lines in Feature Extraction

To eliminate rulings effects in feature extraction, we skip the two contour segments if they lie on a ruling. Otherwise, we compensate for the two local angles by rotating them $-\boldsymbol{\beta_2}$. We detect the "salt-and-pepper" noise and ignore it since its contour length is usually small. In this way, we can effectively handle scanning noise, page

skew, and the presence of rulings during feature extraction, rather than in the conventional pre-processing stage. Figure 5.3 shows invalid contour points overlapping rulings in red and valid ones in blue. Some contour segments are not colored because the authors suggest using only half of the PDF matrix.[40]

On the other hand, we believe that the presence of ruling lines changes the way an individual writes which may prove useful for writer identification. Therefore, the displacement features are computed on a per-page basis for sufficient votes in the histogram. First, we compute the upper and lower profiles in each word polygon. Next, we find two adjacent rulings which bound the upper and lower profiles. Finally, we quantize the distances from the profiles to the rulings and accumulate the votes in a histogram.

We illustrate this feature extraction process in Figure 5.4. The bin size is 20 pixels and the histogram has 41 bins to account for both positive and negative displacement. These numbers are empirically determined such that the displacement covers full spacing above and below from the handwriting location. Normalizing the histogram, we then obtain 41-D displacement feature vectors. We concatenate them with the original contour-hinge features for classifier training and testing.

We evaluate three systems in our experiments. **Remove-Ruling** is the baseline system that reflects the conventional paradigm where ruling removal and broken stroke recovery is adopted. The other two represent how our attempt of handling and exploiting them.

Remove-Ruling : remove ruling lines and try to recover broken strokes by local shape analysis.[46]

Offset-Ruling : detect ruling lines using a model-based method and account for

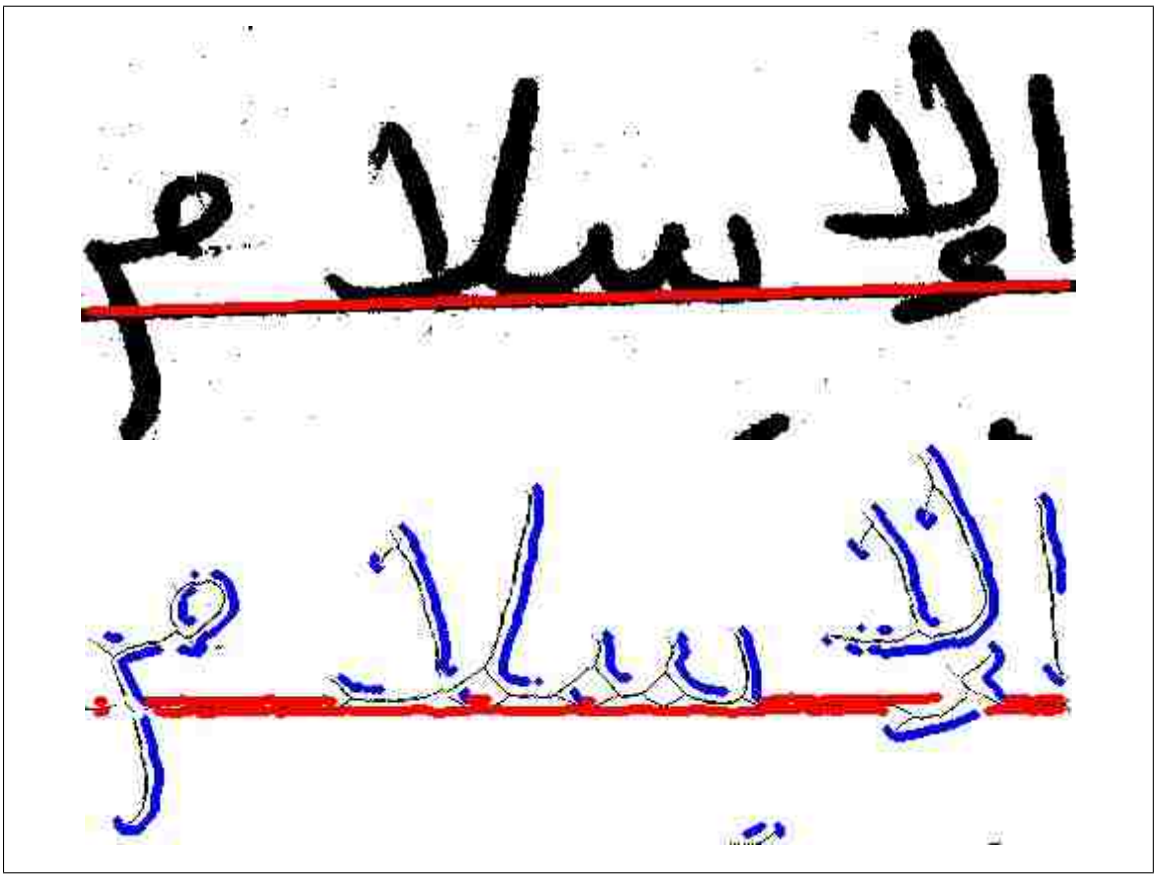


Figure 5.3: Accounting for rulings during feature extraction. In the lower half, blue pixels are valid contour pixels that contribute to the contour-hinge features, while red pixels are contour pixels that overlap the ruling lines.

them during feature extraction.

Exploit-Ruling : add displacement features to **Offset-Ruling**.

5.3.2 Handle Page Skew in Feature Extraction

We test two ways to handle page skew when computing contour features. First, when traversing contours, we explicitly subtract the page skew from the hinge segment

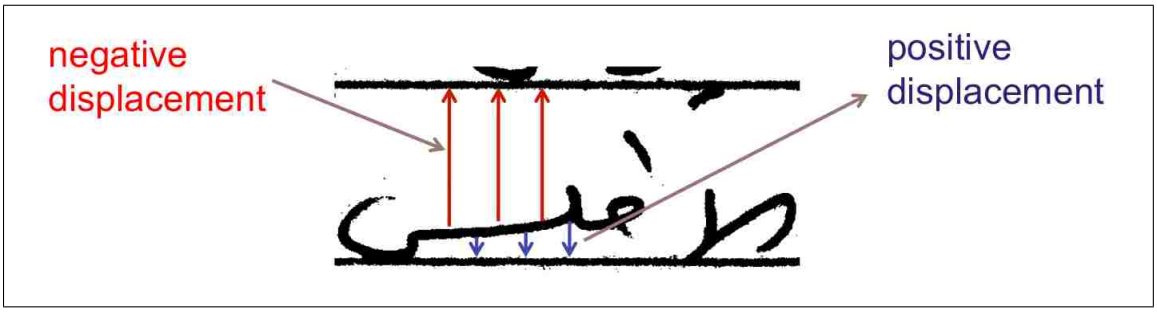


Figure 5.4: An illustration of computing displacement features exploiting pre-printed ruling lines.

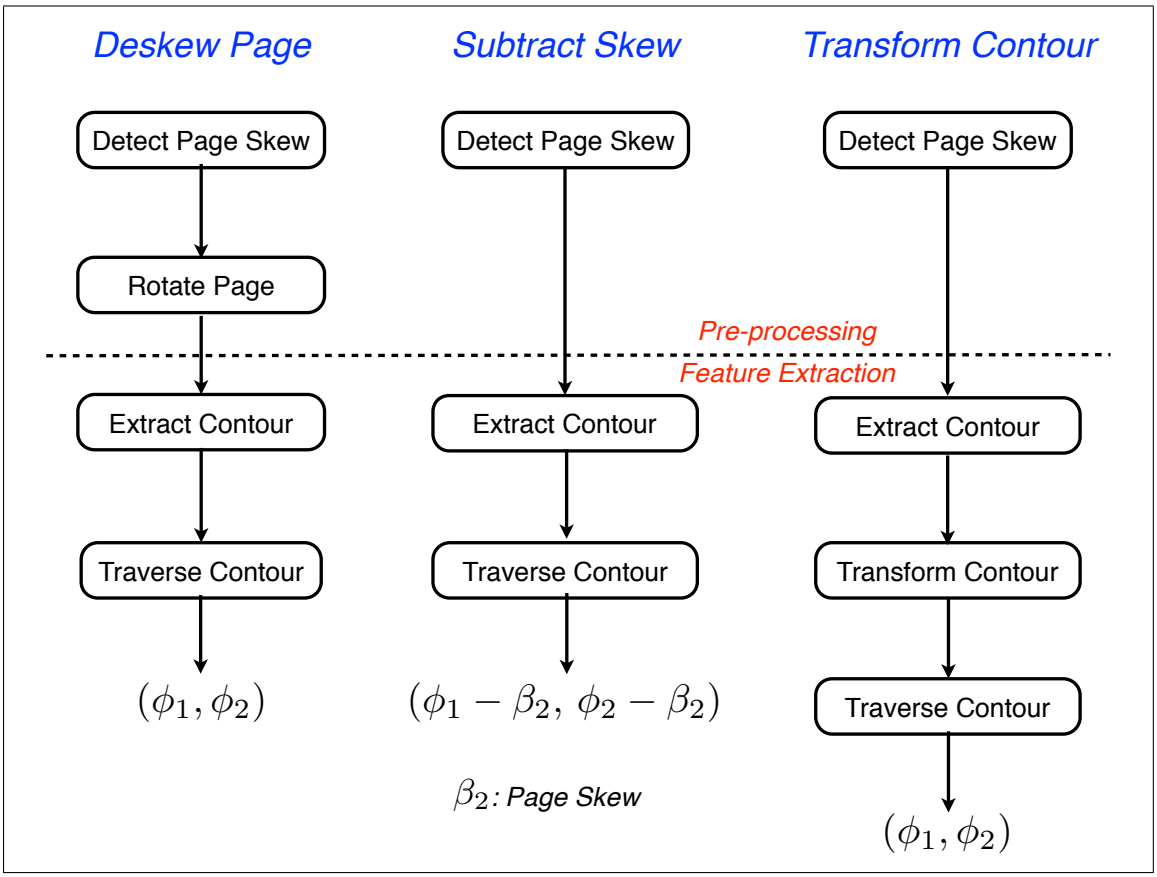


Figure 5.5: A workflow diagram showing processing modules in different feature extraction methods in evaluation. $(\cdot, \cdot)$ means the actual angles used to index in the PDF matrix.

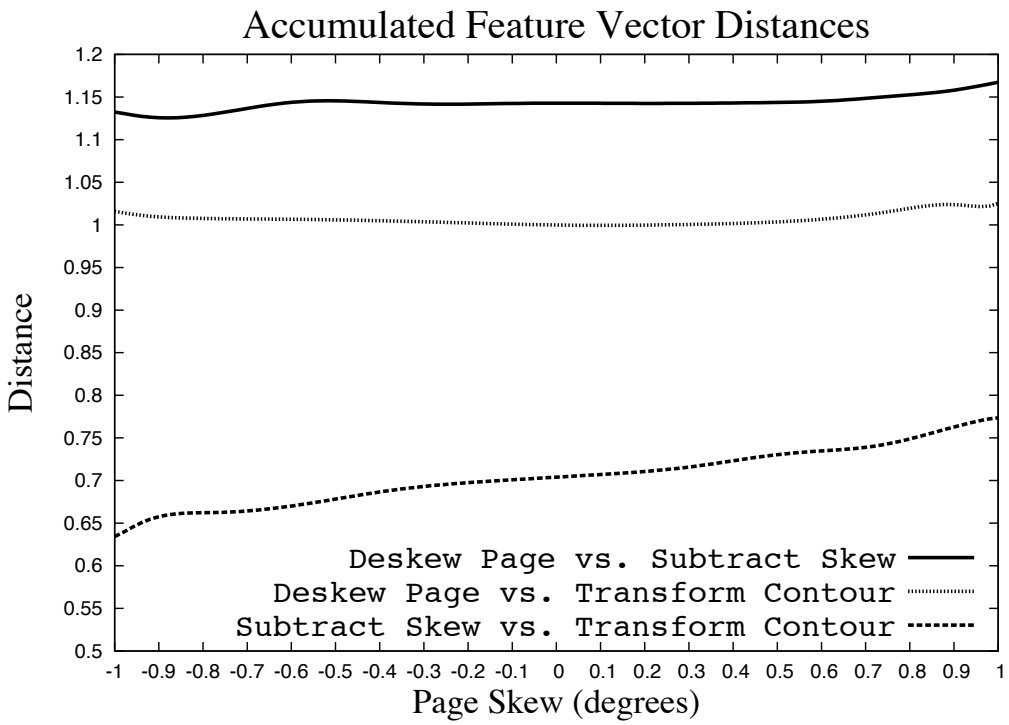


Figure 5.6: Differences of accumulated feature vectors between the three methods.

angles. Second, we transform the coordinates of extracted contour pixels against the skew direction before extracting the hinge features. Therefore, including the traditional deskewing method we have three approaches to evaluate, as shown in Figure 5.5.

These methods differ in the order that page skew is handled in the processing pipeline. **Deskew Page** normalizes page skew during pre-processing, by rotating the bitmap directly. **Subtract Skew** pretends there is no page skew until computing

the indices of bins in the PDF matrix. **Transform Contour** first extracts the contours and then rotates them in a continuous coordinate system before computing the contour hinge angles. Note that this resolves the problematic coordinate transform in **Deskew Page** because here we use real values to represent point coordinates.

Deskew Page: this serves as the baseline system which rotates the image against to page skew direction, as in traditional pre-processing.

Subtract Skew: when traversing contours, subtract the page skew from the angles of hinge segments, and then compute the indices in the PDF matrix.

Transform Contour: counteract the page skew by transforming the coordinates of extracted contours in a continuous coordinate system.

In a continuous world without quantization effects, these methods would generate the same feature vector. Due to the discrete 2-D digital grid, however, they may generate distinct feature vectors, as in Figure 5.6. We generate this figure by extracting features on a standard ellipse under different page skew in $[-1.0, 1.0]$. After extracting feature vectors from the three methods, we compute the accumulated distances between pairs of methods and then plot their distributions. It is clear that **Deskew Page** tends to generate feature vectors that are less likely to be similar to those from the other two approaches. This is understandable because rotating pixels by resampling on a 2-D grid is irreversible and may introduce artifacts.[176]

In the original proposal of contour-hinge features,[40] the authors only use half of the matrix as a feature vector ($\phi_2 >= \phi_1$), considering the other half redundant. We examine the benefits of using the full PDF matrix as feature vectors.

Half_PDF: the baseline method from the original literature which uses half of the PDF matrix as feature vectors, as in the original literature.[40]

Full_PDF: use the full matrix as feature vectors, so the feature vectors are $(2n)^2 =$ 576-D, $n = 12$.

5.4 Experimental Setup

The Arabic dataset for evaluation was provided by the Linguistic Data Consortium (LDC).[1] 61 writers contributed their handwriting into our evaluation dataset, each of whom contributed 10 handwritten pages. Each page was scanned at 600 DPI with a bitonal setting. A typical size for a page image is $5100w \times 6600h$. A sample document of this dataset is shown in Figure 5.1.

We divided the dataset into five folds, each containing two pages by each writer. Each page was then annotated with polygon bounding boxes for handwritten words and text lines, along with the corresponding text transcription. Using a 5-fold cross-validation, each text line was tested once. In total, we evaluated 4,890 text lines in our experiments. Conventional pre-processing such as median filtering or deskewing was not used.

Classification using SVMs with a radial basis function (RBF) kernel was conducted on a text line basis. Given the fact that each page was scribed by one single person, all the writer identification accuracy reported in this Chapter was conducted on the page level which represented the most voted writer ID for all text lines within the same page.

Table 5.1: Writer identification accuracy on different approaches of handling pre-printed ruling lines. All the numbers are Top-1 accuracy in the output n-best lists.

	Remove-Ruling	**Offset-Ruling**	**Exploit-Ruling**
Fold 0	60.88%	66.35%	69.09%
Fold 1	60.39%	65.74%	67.02%
Fold 2	60.06%	63.91%	64.30%
Fold 3	62.39%	67.14%	68.09%
Fold 4	62.37%	65.99%	69.08%
Average	61.22%	**65.83%**	**67.67%**

5.5 Experimental Results

5.5.1 Handle Ruling Lines in Feature Extraction

Table 5.1 presents the Top-1 accuracy of different approaches in our evaluation. As we can see, **Offset-Ruling** outperformed the baseline **Remove-Ruling**. This result shows that it is feasible to handle the effects of rulings during feature extraction, which avoids the difficult problem of recovering broken strokes. By examining the sample images in which **Offset-Ruling** performed better, we found out that the ruling line removal method was inferior in the sense that we observed remaining ruling line segments and mistakenly recovered strokes in the results, which reflected a negative accuracy impact in the baseline.

We also found that the presence of pre-printed rulings did help identify writers. By adding the displacement features in **Offset-Ruling**, we boosted the performance to 67.67%, an absolute gain of 1.84% over **Offset-Ruling**, and 6.45% over

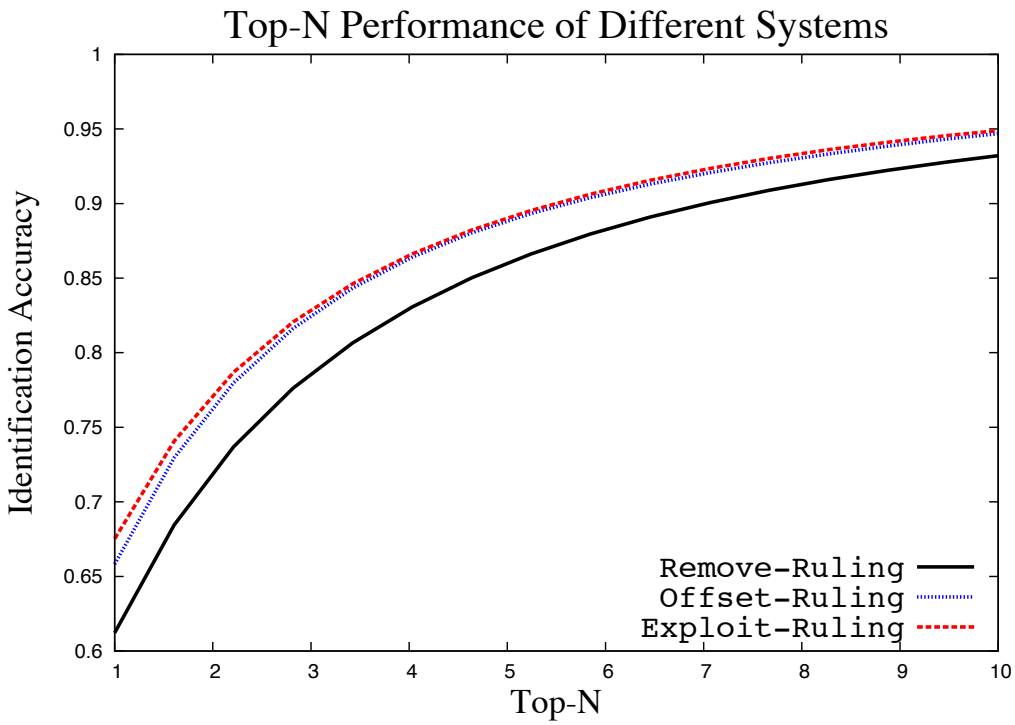


Figure 5.7: The Top-N performance of evaluated systems.

Remove-Ruling. Figure 5.7 shows the consistency of performance gains obtained in **Offset-Ruling** and **Exploit-Ruling** over Top-N choices. All the performance gains are statistically significant with a confidence level of 95%, validated by the McNemar's Test presented in Section 1.7.

Viewing this experiment in our composite-model framework, we found it feasible to pass pre-printed information along the processing pipeline and to make use of them for the follow-up modules like feature extraction. This is an advantage of our composite-model framework in document image analysis.

Table 5.2: Writer identification accuracy on different methods.

	Deskew	Subtract Skew	Transform Contour
Fold 0	75.83%	77.76%	75.73%
Fold 1	74.35%	71.58%	78.48%
Fold 2	80.61%	79.45%	80.19%
Fold 3	78.31%	77.29%	81.81%
Fold 4	81.59%	82.48%	81.40%
Average	78.14%	77.71%	**79.52%**

Table 5.3: Asymmetric PDF matrix in feature extraction.

	Half PDF	Full PDF
Deskew	73.47%	**78.14%**
Transform Countour	74.89%	**79.52%**

5.5.2 Handling Page Skew in Feature Extraction

We showed performance gains from our proposed systems that counteracted page skew during feature extraction over rotating bitmaps during pre-processing. The experimental results are summarized in Table 5.2. The baseline seemed to outperform the subtract-skew based system, but McNemar's Test (see Section 1.7) showed that this performance difference of 0.43% is not statistically significant. In other words, these two systems performed similarly. If we chose to rotate the contours during feature extraction, however, we obtained a performance gain (1.4%) with statistical significance (Section 1.7). This result validated our hypothesis that it is possible to avoid the damage caused by rotating bitmaps during traditional pre-processing but to exploit it during feature extraction.

Next, we explain why we chose to use the full PDF matrix as feature vector

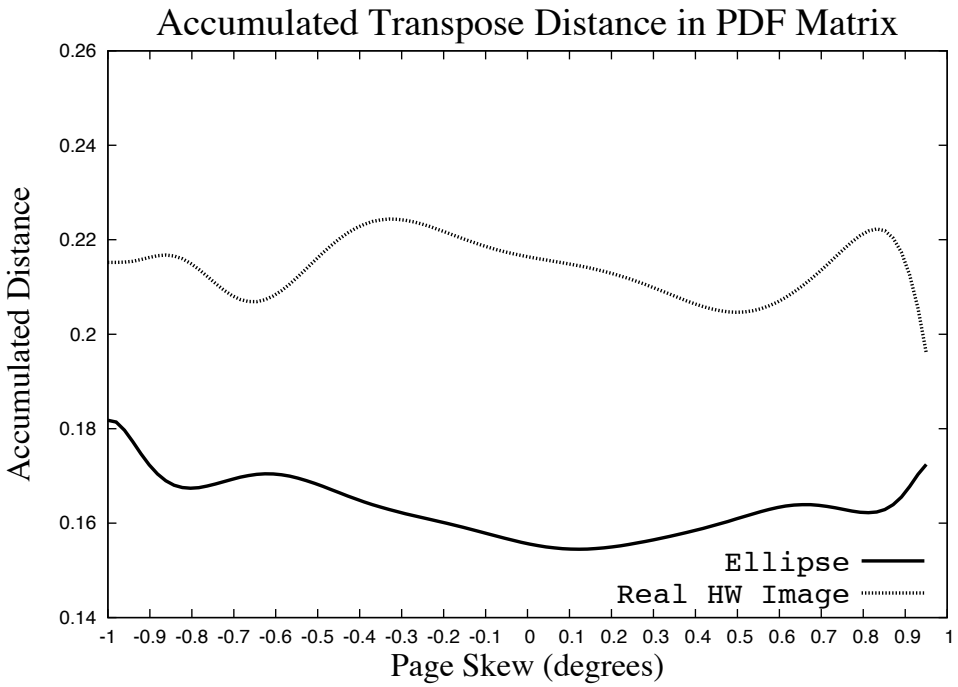


Figure 5.8: Transposed PDF matrix distance of different objects. The ellipse is rotated at different angles in $[-1.0°, 1.0°]$ and the other curve is summarized with the evaluation dataset.

for writer identification. In Bulacu and Schomaker's work,[40] they used half of the matrix as a feature vector, considering the other half redundant. This idea assumes the contours are symmetric with respect to the horizontal axis, so the PDF matrix is symmetric. We investigated this by rotating a standard ellipse by different angles for feature extraction. For each skew angle in $[-1.0, 1.0]$, we computed the accumulated

transpose matrix distance D in the PDF matrix M:

$$D = \sum_{i=1}^{2n} \sum_{j=i+1}^{2n} \|M[i][j] - M[j][i]\| \tag{5.1}$$

where $n = 12$. Then, we computed the average distance for each bin in the skew range. Likewise, we computed this metric using all the text line images in our evaluation dataset. The difference of the two is shown in Figure 5.8. Although the PDF matrix of the text lines seemed symmetric, the distance between their transposed elements was significantly larger than that from a real symmetric object. Hence, we considered this difference may be useful information to exploit. The results in Table 5.3 validated our hypothesis, showing that the subtle differences in the transposed entries (shown in Figure 5.8) played an important role in identifying writers. All methods obtained a large performance gain over the baseline which used only half of the PDF matrix.

5.6 Conclusions

We presented a new methodology for handling pre-printed ruling lines and page skew in the tasks of writer identification. As pre-printed information, ruling lines are considered challenging artifacts to remove during pre-processing. We investigated them within the framework of composite-model where we extracted them from the processing pipeline and attempted to counteract them during feature extraction. Ensuring this image integrity enabled us to further exploit the impact of pre-printed ruling lines on people's handwriting behaviors, which was reflected in performance gains when adding the displacement features.

We studied several possibilities of handling page skew during feature extraction rather than the traditional pre-processing. We showed by experiments that it is entirely feasible to counteract page skew during feature extraction without modifying the original image permanently. We also examined several ways of implementing the contour-hinge features and their relationships to pre-printed ruling lines and the page skew. The experimental evaluation showed significant performance gains when all these subtleties were properly addressed.

Chapter 6

Summary and Future Directions

6.1 Dissertation Summary

Traditional DIA methodology consists of a pipeline of processing stages where each stage makes assumptions about the nature of the input image. These assumptions add up to pre-conditions that must be satisfied for the procedure in question to work properly. In general, these pre-conditions are satisfied by providing a preceding module that detects violations of the assumption and corrects for it. As a result, the original image is modified in an irreversible way as it is passed along the document processing pipeline. Although it becomes easier to design a particular module after this type of normalization, important image information that could be useful to later stages may be discarded along the pipeline.

We introduced a composite-model framework for handwritten document image understanding which consists of three major components: pre-printed information,

user-added data, and digitization characteristics. In general, a handwritten document can be decomposed into these three components ordered by the time each was generated. Other variants may include iterations of these three operations: printing, writing and scanning (faxing). Viewing DIA this way, we were able to separate individual component and maintain information integrity by passing them along the processing pipeline. The advantage of ensuring information integrity is that it provides us with opportunities to exploit information that is usually eliminated otherwise, in the traditional methodology.

Specifically, we first investigated one type of pre-printed information, pre-printed ruling lines, that are prevalent in handwritten document image analysis. Ruling lines are designed to help people write neatly, but they raise several challenges when conducting document image analysis. One common situation is that handwriting constantly overlaps pre-printed ruling lines due to scanning. Traditionally, these ruling lines are treated as artifacts and thus are excluded during pre-processing. This approach, however, can not avoid modifying handwritten strokes, which can negatively affect follow-up handwriting analysis such as writer identification. In our proposed ruling line detection algorithm, we found that our approach managed to make errors of one order of magnitude less than the existing one in the literature (see Table 3.3), on three standard test datasets of different scripts and conditions.

Second, we preserved image integrity by detecting and passing the attributes of rulings lines on the processing pipeline so that it is possible to compensate for them at later stages, such as feature extraction. It turned out to be beneficial to exploit the impact of ruling lines on people's handwriting. The significant performance gain of 6.45% in writer identification is the direct result of ensuring image integrity.

Third, we investigated the possibility of handling page skew during feature extraction, rather than in pre-processing. Page skew, one type of scanning variation, can be determined by analyzing pre-printed ruling lines. In the literature, page skew is also treated as a type of artifact and is normalized during pre-processing by rotating the bitmap. This processing, however, is undesirable on a discrete 2-D grid and modifies the bitmap permanently. Therefore, we introduced a new method of correcting page skew by detecting and passing over the skew angle to feature extraction, which compensated for the skew angle when computing the local angular features. Experimental evaluation showed that our method provided equally discriminative features for writer identification while preserving the image integrity. We also investigated the impact of ruling lines on the implementation of feature extraction and found out that the full PDF matrix is more useful for feature extraction than the half matrix suggested originally.

To sum up, we proposed an information preserving framework for analyzing noisy handwritten document images. Within this framework, we extracted and separated information into three components: pre-printed information, user-added data, and digitization characteristics. By examining the impact of pre-printed information and digitization characteristics, we were able to obtain significant performance gains in writer identification without discarding information before feature extraction.

6.2 Future Research Directions

This dissertation should be viewed as a starting point for a preferable way of document image analysis. With this unified composite-model framework, it is possible

to avoid modifying bitmaps permanently while accounting for the impact of various pre-printed information and digitization characteristics.

6.2.1 Ruling Lines: Pre-printed vs. Hand-drawn

It is possible for users to draw ruling lines by hand, with or without a ruler. This is common in notebooks that contain several types of handwritten notes, e.g., for course work.[58] Hand-drawn ruling lines indicate clearly a user's intent to separate the region of interest from the rest of the page, regardless of whether pre-printed ruling lines are present. This type of information would be useful in aiding any composite-model based handwritten document image analysis.

As for the differentiation of these types of ruling lines, hand-drawn ones without using rulers, which are likely to be high-order polynomial curves, can be approximated by a sequence of line segments that are chained together. So this type of ruling is easier to detect. On the other hand, hand-drawn lines using rulers will resemble pre-printed ruling lines, so they are more difficult to detect. We might want to start with the following 2-class classification scheme:

(i) Use Hough transform to obtain short line segments and then group them into clusters based on their ρ values.

(ii) For each cluster, fit a line with the least square errors.

(iii) Extract features include statistics of ρ and θ of all lines segments, error statistics of them against the fitted lines, and statistics of the line thickness.

In the end, we can separate pre-printed ruling lines and hand-drawn ones, and

hand-drawn ruling lines can be further exploited for analysis such as sketch recognition, writer identification, etc.

6.2.2 Handwritten Tabular Structure Analysis

The requirement of a form template input in Chapter 4 seems artificial and the manual process of generating such a model file is time-consuming and error-prone. It would be useful to help automate this process when examining a large-scale document image corpus from scratch.

First, we may want to address the model generation problem in a semi-automated way: group document images by similarity analysis on their document layout structures and then ask human examiners to review the results and correct algorithmic errors. This similarity analysis would require extraction of other document components such as text, logos, signatures, machine-printed business header, footnote page number, punch holes, stapling marks, etc. Also, it would be interesting to adapt by human corrections such that the clustering algorithm may make less mistakes as the clustering proceeds forward.

Second, for noisy handwritten documents as in this thesis, the task is essentially an optimization problem of selecting the most plausible subset of line segments that constitute the tabular structure. The current cost function we used in Chapter 4 is still simple. For example, the shared characteristics within the same cluster should be exploited for tabular structure analysis, including the displacement of machine-printed text and handwriting, and their relative displacement against detected ruling line segments. Therefore, the optimization of ruling line selection should be conducted on clusters rather than individual documents.

6.2.3 Image Integrity for Handwriting Recognition

We used writer identification as an example of DIA tasks and examined it within the composite-model framework. Similar idea can be extended to handwriting recognition too. In addition to all the benefits of preserving image integrity for writer identification which can be used to adapt writer-dependent handwriting recognition engine, we may also compensate for slant that is usually corrected on the word/character level, where the slant angle of such components is often normalized to facilitate feature extraction.

Note that this is also a process of image modification. We may want to examine when the slant angle is estimated, how to compensate for that during feature extraction for handwriting recognition. For example, Gabor filters can be used for directional feature extraction.

6.2.4 Future Work at a Higher Level

We have proven it is beneficial to use the composite-model framework for single-page DIA. It would be interesting to investigate the framework in cross-page DIA because that is closer to an industrial setting. Cross-page DIA goes beyond the scope of clustering documents based on similarity analysis. It covers topics of iterative learning between pages, and adaptation to a specific script, a handwriting style, document structures, digitization characteristics, etc. Within this framework, pre-printed information, user-added data, and digitization characteristics can serve as useful meta-data information for document analysis at the corpus level.

Pre-printed information may play a central role in helping organize documents in an unordered corpus. For example, pre-printed ruling line model can be used to

distinguish documents scanned from different notebooks, assuming these notebooks have different ruling line attributes, spacing, number of lines, etc. In addition, pre-printed business headers, logos, and index numbers in the header/footnote are tremendously useful in grouping documents and even sorting them. Reading order analysis within a document corpus is considered one of the crucial tasks in cross-page document analysis.

Challenges remain no matter what type of DIA model or framework is proposed. For example, we are aware of radical differences between document collections from research labs versus from the field. The former represents a controlled environment that affects every aspect of the final document corpus: writing materials, writing instruments, scanner settings, and more importantly elicited handwriting. For example, the guideline on the instruction page of data collection, the time constraints to collect handwriting, and how human subjects are motivated/trained, etc., all affect handwriting. Although convenient for researchers to conduct systematic evaluation, such datasets would not exhibit as much complexity as in those collected in the field, e.g., historical document corpora scanned without eliminating any document samples. Thus, we would like to call for more handwritten datasets that contain minimum elicitation and curation, while preserving inter-person variations of handwriting. The transition to focus on this type of datasets is what we believe to be one of the trends in the DIA community.

Bibliography

[1] The Linguistic Data Consortium. http://www.ldc.upenn.edu/.

[2] Merriam-Webster Dictionary. http://www.merriam-webster.com/.

[3] Multilingual Automatic Document Classification and Translation Evaluation (MADCAT). http://www.nist.gov/itl/iad/mig/madcat.cfm.

[4] NIST Handprinted Forms and Characters Database. http://www.nist.gov/srd/nistsd19.cfm.

[5] W. Abd-Almageed, J. Kumar, and D. Doermann. Page rule-line removal using linear subspaces in monochromatic handwritten Arabic documents. In *Proc. of the 12th International Conference on Document Analysis and Recognition*, pages 768–772, 2009.

[6] M. Agrawal and D. Doermann. Clutter noise removal in binary document images. In *International Conference on Document Analysis and Recognition*, pages 556–560, 2009.

[7] M. Agrawal and D. Doermann. Voronoi++: A dynamic page segmentation approach based on Voronoi and docstrum features. In *Proceedings of*

the International Conference on Document Analysis and Recognition, pages 1011–1015, 2009.

[8] M. Agrawal and D. Doermann. Context-aware and content-based dynamic Voronoi page segmentation. In *Proceedings of the International Workshop on Document Analysis Systems*, pages 73–80, 2010.

[9] T. Akiyama and N. Hagita. Automated entry system for printed documents. *Pattern Recognition*, **23**[11]:1141–1154, 1990.

[10] H. Al-Yousefi and S. Udpa. Recognition of Arabic characters. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **14**[8]:853–857, 1992.

[11] A. Amin. Off-line Arabic character recognition: a survey. In *Proceedings of the 4th International Conference on Document Analysis and Recognition*, pages 596–599, 1997.

[12] A. Amin. Off-line Arabic character recognition: the state of the art. *Pattern Recognition*, **31**[5]:517–530, 1998.

[13] A. Amin and J. Mari. Machine recognition and correction of printed Arabic text. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Special Issue)*, **19**[5]:1300 – 1306, 1989.

[14] C. An and H. Baird. The convergence of iterated classification. In *Proceedings of the International Workshop on Document Analysis Systems*, pages 663–670, 2008.

[15] A. Antonacopoulos, C. Clausner, C. Papadopoulos, and S. Pletschacher. Historical document layout analysis competition. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1516–1520, 2011.

[16] A. Antonacopoulos, C. Clausner, C. Papadopoulos, and S. Pletschacher. ICDAR 2013 competition on historical newspaper layout analysis. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1454–1458, 2013.

[17] A. Antonacopoulos, B. Gatos, and D. Bridson. Icdar 2005 page segmentation competition. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 75–79, 2005.

[18] A. Antonacopoulos, B. Gatos, and D. Bridson. Page segmentation competition. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1279–1283, 2007.

[19] A. Antonacopoulos, B. Gatos, and D. Karatzas. Icdar 2003 page segmentation competition. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 688–692, 2003.

[20] A. Antonacopoulos, S. Pletschacher, D. Bridson, and C. Papadopoulos. ICDAR 2009 page segmentation competition. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1370–1374, 2009.

[21] K. Arvind, J. Kumar, and A. Ramakrishnan. Line removal and restoration of handwritten strokes. In *Proc. of the 7th international Conference on Computational Intelligence and Multimedia Application*, pages 208–214, 2007.

[22] H. Avi-Itzhak, J. Van Mieghem, and L. Rub. Multiple subclass pattern recognition: a maximin correlation approach. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **17**[4]:418–431, 1995.

[23] H. Avi-Lizhak, T. Diep, and H. Garland. High accuracy optical character recognition using Neural Networks with centroid dithering. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **17**[2]:218–223, 1995.

[24] A. Bagdanov and J. Kanai. Projection profile based skew estimatino algorithm for JBIG compressed images. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 401–405, 1997.

[25] H. Baird. The skew angle of pirnted documents. In *Proceedings of the Conference Society of Phtographic Scientists and Engineers*, **40**, pages 21–24, 1987.

[26] H. Baird. Document image defect models. In H. Baird, H. Bunke, and K. Yamamoto, editors, *Proceedings of the International Workshop on Syntactic and Structural Pattern Recognition*, pages 546–556, 1990.

[27] H. Baird. Calibration of document image defect models. In *Proceedings of the 2nd Annual Symposium on Document Analysis and Information Retrieval*, pages 1–16, 1993.

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

[29] H. Baird. Document image quality: making fine discrinimations. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 459–462, 1999.

[30] H. Baird. The state of the art of document image degradation modeling. In *Proceedings of the International Workshop on Document Analysis Systems*, pages 1–16, 2000.

[31] H. Baird, H. Bunke, and K. Yamamoto, editors. *Structured Document Image Analysis*, chapter Document image defect models. Springer-Verlag 1992, 1995.

[32] H. Baird, S. Jones, and S. Fortune. Image segmentation by shape-directed covers. In *Proceedings of the International Conference on Pattern Recognition*, pages 820–825, 1990.

[33] H. Baird and G. Nagy. A self-correcting 100-font classifier. In *Proceedings of the 1994 Electronic Imaging: Science and Technology, Proceedings of SPIE*, 1994.

[34] J. Banerjee, A. Namboodiri, and C. Jawahar. Contextual restoration of severely degraded document images. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, pages 517–524, 2009.

[35] A. Bensefia, A. Nosary, T. Paquet, and L Heutte. Writer identification by writer's invariants. In *Proc. the international workshop on frontiers in handwriting recognition*, pages 274–279, 2002.

[36] J. Bernsen. Dynamic thresholding of grey-level images. In *Proceedings of the International Conference on Pattern Recognition*, pages 1251–1255, 1986.

[37] R. Bertolami and H. Bunke. Integration of n-gram language models in multiple classifier systems for offline handwritten text line recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, **22**[7]:1301–1321, 2008.

[38] R. Bozinovic and S. Srihari. Off-line cursive script word recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **11**[1]:68–83, 1989.

[39] T. Breuel. Two geometric algorithms for layout analysis two geometric algorithms for layout analysis two geometric algorithms for layout analysis. In *Proceedings of the International Workshop on Document Analysis Systems*, pages 188–199, 2002.

[40] M. Bulacu and L. Schomaker. Text-independent writer identification and verification using textural and allographic features. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **29**:701–717, 2007.

[41] M. Bulacu, L. Schomaker, and L. Vuurpijl. Writer identification using edge-based directional features. In *Proc. the 7-th international conference on document analysis and recognition*, pages 937–941, 2003.

[42] H. Bunke, S. Bengio, and A. Vinciarelli. Offline recognition of unconstrained handwritten texts using hmms and statistical language models offline recognition of unconstrained handwritten texts using hmms and statistical language models offline recognition of unconstrained handwritten texts using HMMs and statistical language models. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **26**[6]:709–720, 2004.

[43] D. Burr. Elastic matching of line drawings. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **3**[6]:708–712, 1981.

[44] H. Cao and V. Govindaraju. Handwritten carbon form preprocessing based on Markov Random Field. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[45] H. Cao and V. Govindaraju. Preprocessing of low-quality handwritten documents using Markov Random Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **31**:84–96, 2009.

[46] H. Cao, R. Prasad, and P. Natarajan. A stroke regeneration method for cleaning rule-lines in handwritten document images. In *Proc. of the MOCR workshop at the 10th international Conference on Document Analysis and Recognition*, 2009.

[47] N. Carter and R. Bacon. *Automatic recognition of printed music*, pages 456–465. Springer-Verlag, 1992.

[48] R. Casey and G. Nagy. Recognition of printed chinese characters. *IEEE Transactions on Electronic Computers*, **15**[1]:91–101, 1966.

[49] R. Cattoni, T. Coianiz, S. Messelodi, and C. Modena. Geometric layout analysis techniques for document image understanding: a review. Technical report, IRST, 1998.

[50] F. Cesarini, S. Marinari, L. Sarti, and G. Soda. Trainable table location in document images. In *International Conference on Pattern Recognition*, pages 236–240, 2002.

[51] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent System and Technology*, **2**[3], 2001.

[52] J. Chen. Handwritten biometric systems and their robustness evaluation: a survey. Technical report, Lehigh University, 2009.

[53] J. Chen and H. Lee. An efficient algorithm for form structure extraction using strip projection. *Pattern Recognition*, **31**[9]:1353–1368, 1998.

[54] J. Chen and D. Lopresti. A model-based ruling line detection algorithm for noisy handwritten documents. In *Proceedings of the 11th International Conference on Document Analysis and Recognition*, pages 404–408, September 2011.

[55] J. Chen and D. Lopresti. Table detection in noisy off-line handwritten documents. In *Proceedings of the 2011 11th International Conference on Document Analysis and Recognition*, pages 399–403, September 2011. 399-403.

[56] J. Chen and D. Lopresti. Model-based tabular structure detection and recognition in noisy handwritten documents. In *Proceedings of the 13th International Conference on Frontiers in Handwriting Recognition*, pages 75–80, 2012.

[57] J. Chen, D. Lopresti, and E. Kavallieratou. The impact of ruling lines on writer identification. In *Proc. of the 12th International Conference on Frontiers in Handwriting Recognition*, pages 439 – 444, 2010.

[58] J. Chen, D. Lopresti, and B. Lamiroy. A real-world noisy unstructured handwritten notebook corpus for document image analysis research. In *Proceedings of the Joint Workshop on Multilingual OCR and Analytics for Noisy Unstructured Text Data*, 2012.

[59] M. Chen, A. Kundu, and J. Zhou. Off-line handwritten word recognition using a Hidden Markov Model type stochastic network. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **16**[5]:481–496, 1994.

[60] Y. Chenevoy and A. Belaid. Hypothesis management for structured document recognition. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 121–129, 1991.

[61] W. Cheng and D. Lopresti. Parameter calibration for synthesizing realistic-looking variability in offline handwriting. In *Proc. Document Recognition and Retrieval XVIII (IS&T/SPIE International Symposium on Electronic Imaging)*, 2011.

[62] C. Chow and T. Kaneko. Automatic detection of the left ventricle from cineangiograms. *Computers and Biomedical Research*, **5**:388–410, 1972.

[63] C. Chun and R. Chamchong. A review of evaluation of optimal binarization technique for character segmentation in historical manuscripts. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 236–240, 2010.

[64] G. Ciardiello, G. Scafuro, M. Degrandi, M. Spada, and M. Roccotelli. An experimental system for office document handling and text recogntion. In *Proceedings of the International Conference on Pattern Recognition*, pages 739–743, 1988.

[65] D. Ciresan, U. Meier, and J. Schmidhuber. Multi-column Deep Neural Networks for image classification. In *Proceedings of the 25th IEEE Conference on Computer Vision and Pattern Recognition*, pages 3642–3649, 2012.

[66] R. Clawson, K. Bauer, G. Chidester, M. Tyler-Pohontsch, D. Kennard, J. Ryu, and W. Barrett. Automated recognition and exrtaction of tabular fields for the indexing of census records. In *Proc. Document Recognition and Retrieval XVIII (IS&T/SPIE International Symposium on Electronic Imaging)*, pages 8658–17, 2013.

[67] B. Couasnon and Camillerapp. A way to separate knowledge from program in structured document analysis: application to optical music recognition. In *Proceedings of the 3rd International Conference on Document Analysis and Recognition*, pages 1092–1097, 1995.

[68] V. d'Andecy, J. Camillerapp, and I. Leplumey. Kalman filtering for segment detection: application to music score analysis. In *Proc. of the International Conference on Pattern Recognition*, pages 301–305, 1994.

[69] A. Dengel. Initial learning of document structure. In *Proceedings of the 2nd International Conference on Document Analaysis and Recognition*, pages 86–90, 1993.

[70] B. Dhandra and M. Hangarge. Morphological reconstruction for word level script identification. *International Journal of Computer Science and Security*, **1**[1]:41–51, 2007.

[71] M. Diem, S. Fiel, A. Garz, M. Keglevic, F. Kleber, and R. Sablatnig. ICDAR 2013 competition on handwritten digit recognition. In *Proceedings of the 2013 12th International Conference on Document Analysis and Recognition*, pages 1454–1459, 2013.

[72] T. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, **10**:1895–1923, 1998.

[73] D. Doermann, E. Rivlin, and I. Weiss. Logo recognition using geometric invariants. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 894–897, 1993.

[74] D. Doermann, E. Rivlin, and I. Weiss. Applying and differential invariants for logo recognition. *Machine Vision and Application*, **9**[2]:73–86, 1996.

[75] V. Dongre and V. Mankar. A review of research on devnagari character a review of research on Devnagari character recognition. *International Journal of Computer Applications*, **12**[2], 2010.

[76] D. Dori. Orthogonal zig-zag: An algorithm for vectorizing engineering drawings compared with hough transform. *Advances in Engineering Software*, **28**[1]:11–24, 1998.

[77] D. Dori, Y. Liang, and J. Dowell. Sparse-pixle recognition of primitives in engineering drawings. *Machine Vision and Applications*, **6**[2-3]:69–82, 1993.

[78] D. Dori and W. Liu. Sparse pixel vectorization: an algorithm and its performance evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **21**[3]:202–215, 1999.

[79] R. Duda and P. Hart. Use of the Hough transformation to detect lines and curves in pictures. *Communications of the ACM*, **15**[1]:11–15, 1972.

[80] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley-Interscience, 2000.

[81] L. Eikvil, T. Taxt, and K. Moen. A fast adaptive method for binarization of document images. In *Proceedings of the 1st International Conference on Document Analaysis and Recognition*, pages 435–443, 1991.

[82] F. El-Khaly and M. Sid-Ahmed. Machine recognition of optically captured machine printed Arabic text. *Pattern Recognition*, **23**[11]:1207–1214, 1990.

[83] A. El-Yacoubi, M. Gilloux, R. Sabourin, and S. Suen. An HMM-based approach for off-line unconstrained handwritten word modeling and recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **21**[8]:752–760, 1999.

[84] D. Embley, M. Hurst, D. Lopresti, and G. Nagy. Table-processing paradigms: a research survey. *International Journal on Document Analysis and Recognition*, **8**[2]:66–86, 2006.

[85] K. Fan, J. Lu, and G. Chen. A feature point clustering approach to the recognition of form documents. *Pattern Recognition*, **31**[9]:1205–1220, 1998.

[86] J. Fisher, S. Hinds, and K. D'Amato. A rule-based system for document image segmentation. In *Proceedings of the International Conference on Pattern Recognition*, pages 567–572, 1990.

[87] H. Fujisawa and Y. Nakano. A top-down approach for the analysis of documents. In *Proceedings of the International Conference on Pattern Recognition*, pages 113–122, 1990.

[88] B. Gatos, D. Danatsas, I. Pratikakis, and S. Perantonis. Automatic table detection in document images. In *Proceedings of the Third International Conference on Advances in Pattern Recognition*, pages 609–618, 2005.

[89] B. Gatos, K. Ntirogiannis, and I. Pratikakis. ICDAR 2009 document image binarization contest. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1375–1382, 2009.

[90] B. Gatos, N. Papamarkos, and C. Chamzas. Skew detection and text line position determination in digitized documents. *Pattern Recognition*, **30**[9]:1505–1519, 1997.

[91] E. Grosicki and H. El Abed. ICDAR 2011 French handwriting recognition competition. In *Proceedings of the 2011 11th International Conference on Document Analysis and Recognition*, pages 1459–1463, 2011.

[92] E. Grosicki, M. Carre, J. Brodin, and E. Geoffrois. Results of the RIMES evaluation campaign for handwritten mail processing. In *Proc. the 11th International Conference on Frontiers in Handwriting Recognition*, pages 941–945, 2008.

[93] I. Guyon, P. Albrecht, Y. Le Cun, J. Denker, and W. Hubbard. Design of a Neural Network character recognizer for a touch terminal. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **24**[2]:105–119, 1991.

[94] J. Ha, R. haralick, and I. Phillips. Recursive X-Y cut using bounding boxes of connected components. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 952–955, 1995.

[95] J.C. Handley. *Electronic imaging technology, Chapter 8 (Document Recognition)*. IS&T/SPIE Optical Engineering Press, 1999.

[96] R.M. Haralick. Document image understanding: geometric and logical layout. In *Proceedings of the 1994 Computer Vision and Pattern Recognition*, pages 385–390, 1994.

[97] A. Hashizume, P. Yeh, and A. Rosenfeld. A method of detecting the orientation of aligned components. *Pattern Recognition Letters*, **4**:125–132, 1986.

[98] D. Hebert, S. Nicolas, and T. Paquet. Discrete CRF based combination framework for document image binarization. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1165–1169, 2013.

[99] Schantz Herbert. *The history of OCR, optical character recognition.* Recognition Technologies Users Association, 1982.

[100] C. Hertel and H. Bunke. *A set of novel features for writer identification*, pages 679–687. Springer, 1998.

[101] S. Hinds, J. Fisher, and D. D'Amato. A document skew detection method using run-length encoding and the Hough transform. In *Proceedings of the International Conference on Pattern Recognition*, pages 464–468, 1990.

[102] Y. Hirayama. A method for table structure analysis using DP matching. In *International Conference on Document Analysis and Recognition*, pages 583–586, 1995.

[103] T. Ho and H. Baird. Large-scale simulation studies in image pattern recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **19**[10]:1067–1079, 1997.

[104] T. Ho, J. Hull, and S. Srihari. Decision combination in multiple classifier systems. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **16**[1]:66–75, 1994.

[105] J. Hochberg, P. Kelly, T. Thomas, and L. Kerns. Automatic script identification from document images using cluster-based templates. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **19**[2]:176–181, 1997.

[106] O. Hori and D. Doermann. Quantitative measurement of the performance of raster-to-vector conversion algorithms. In Tombre Kasturi, editor, *Graphics recognition – methods and applications*, pages 57–68. Springer, 1996.

[107] O. Hori and S. Tanigawa. Raster-to-vector conversion by line fitting based on contours and skeletons. In *Proc. of International Conference on Document Analysis and Recognition*, pages 353–358, 1993.

[108] J. Hu, R. Kashi, D. Lopresti, G. Nagy, and G. Wilfong. Why table ground-truthing is hard. In *Proceedings of the 6th International Conference on Document Analysis and Recognition*, pages 129–133, September 2001.

[109] J. Hu, R. Kashi, D. Lopresti, and G. Wilfong. Medium-independent table detection. In *Document Recognition and Retrieval VIII (IS&T/SPIE Electronic Imaging*, pages 44–55, 2001.

[110] J. Hu, R. Kashi, D. Lopresti, and G. Wilfong. Evaluating the performance of table processing algorithms. *International Journal of Document Analysis and Recognition*, **4**[3]:140–153, 2002.

[111] X. Huang, J. Gu, and Y. Wu. A constrained approach to multifont Chinese character recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **15**[8]:838–843, 1993.

[112] D. Hunter. *Papermaking: the history and technique of an ancient craft*. Courier Dover Publications, 1978.

[113] J. Illingworth and J. Kittler. A survey of the Hough transform. *Graphical Model and Image Processing*, **44**[1]:87–116, 1988.

[114] R. Ingold and D. Armangil. A top-down document analysis method for logical structure recognition. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 41–49, 1991.

[115] Y. Ishitani. Document skew detection based on local region complexity. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 49–52, 1993.

[116] A. Jain, F. Griess, and S. Connell. On-line signature verification. *Pattern Recognition*, **35**:2963–2972, 2002.

[117] A. Jain and B. Yu. Document representation and its appilcation to page decomposition. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **20**[3]:294–308, 1998.

[118] R. Jain and D. Doermann. Offline writer identificaiton using k-adjacent segmetns. In *Proceedings of the 2013 12th International Conference on Document Analysis and Recognition*, pages 769–773, 2013.

[119] E. Justino, F. Bortolozzi, and R. Sabourin. A comparison of SVM and HMM classifiers in the off-line signature verification. *Pattern Recognition Letters*, **26**:1377–1385, 2004.

[120] S. Kahan, T. Pavlidis, and H. Baird. On the recognition of printed characters of any font and size. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **9**[2]:274–288, 1987.

[121] T. Kanungo, R. Haralick, and H. Baird. Power functions and their use in selecting distance functions for document degradation model validation. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 734–739, 1995.

[122] T. Kanungo, R. Haralick, and H. Baird. Validation and estimation of document degradation models. In *Proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval*, pages 217–225, 1995.

[123] T. Kanungo, R. Haralick, H. Baird, and W. Stuetzle. Document degradation models: parameter estimation and model validation. In *Proceedings of the International Workshop on Machine Vision Applications*, 1994.

[124] T. Kanungo, R. Haralick, H. Baird, W. Stuetzle, and S. Madigan. A statistical, nonparametric methodology for document degradation model

validation. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **22**[11]:1209–1223, 2000.

[125] T. Kanungo, R. Haralick, and I. Phillips. Global and local document degreadation models. In *Proceedings of the 2nd International Conference on Document Analaysis and Recognition*, pages 730–734, 1993.

[126] T. Kanungo and S. Mao. Stochastic language models for style-directed stochastic language models for style-directed layout analysis of document images. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **12**[5]:583–596, 2003.

[127] A. Khotanzad and Y. Homg. Invariant image recognition by Zernike moments. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **12**[5]:489–497, 1990.

[128] G. Kim and V. Govindaraju. A lexicon driven approach to handwritten word recognition for real-time applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19**[4]:366–379, 1997.

[129] F. Kimura, M. Shridhar, and Z. Chen. Improvments of a lexicon directed algorithm for recognition of unconstrained handwritten words. In *Proceedings of the 2nd International Conference on Document Analaysis and Recognition*, pages 18–22, 1993.

[130] K. Kise, A. Sato, and M. Iwata. Segmentation of page images using the area Voronoi diagram. *Computer Vision and Image Understanding*, **70**[3]:370–382, 1998.

[131] K. Kise, O. Yanagida, and S. Takamatsu. Page segmentation based on thinning of background. In *Proceedings of the International Conference on Pattern Recognition*, pages 788–792, 1996.

[132] A. Koerich, R. Sabourin, and C. Suen. Lexicon-driven HMM decoding for large vocabulary handwriting recognition with multiple character models. *International Journal of Document Analysis and Recognition*, **6**[2]:126–144, 2003.

[133] B. Kong, I. Phillips, R. Haralick, A. Prasad, and R. Kasturi. A benchmark: performance evaluation of dashed-line detection algorithms. In Tombre Kasturi, editor, *Graphics recognition – methods and applications*, pages 270–285. Springer, 1996.

[134] M. Krishnamoorthy, G. Nagy, S. Seth, and M. Viswanathan. Syntactic segmentation and labeling of digitized pages from technical journals. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **15**[7]:737–747, 1993.

[135] J. Kumar and D. Doermann. Fast rule-line removal using integral images and Support Vector Machines. In *Proceedings of the 11th International Conference on Document Analysis and Recognition*, pages 584–588, September 2011.

[136] A. Laurentini and P. Viada. Identifying and understanding tabular material in compound documents. In *International Conference on Pattern Recognition*, pages 405–409, 1992.

[137] D. Le, G. Thoma, and H. Wechsler. Automated page orientation and skew angle detection for binary document images. *Pattern Recognition*, **27**[10]:1325–1344, 1994.

[138] V. Le, M. Visani, C. Tran, and J. Ogier. Logo spotting for document categorization. In *Proceedings of the International Conference on Pattern Recognition*, pages 3484–3487, 2012.

[139] F. Leclerc and R. Plamondon. Automatic signature verification: the state of the art – 1989-1993. *International Journal of Pattern Recognition and Artificial Intelligence*, **8**:643–660, 1993.

[140] E. Lecolinet and J. Crettez. A grapheme-based segmentation technique for cursive script recognition. In *Proceedings of the 1st International Conference on Document Analaysis and Recognition*, pages 740–748, 1991.

[141] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, **86**, pages 2278–2324, 1998.

[142] K. Lee, Y. Choy, and S. Cho. Geometric structure analysis of document images: a knowledge-based approach. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **22**[11]:1224–1240, 2000.

[143] K. Lee, K. Eom, and R. Kashyap. Character recognition based on programmed attribute-dependent grammar. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **14**[11]:1122–1128, 1992.

[144] S. Lee and D. Ryu. Parameter free geometric document layout analysis. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **23**[11]:1240–1256, 2001.

[145] B. Li, Z. Sun, and T. Tan. Hierarchical shape primitive features for online text-independent writer identification. In *Proc. 10th International Conference on Document Analysis and Recognition*, pages 986–990, Barcelona, Spain, August 2009.

[146] Y. Li, D. Lopresti, G. Nagy, and A. Tomkins. Validation of image defect models for optical character recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **18**[2]:99–107, 1996.

[147] Z. Li, M. Schulte-Austum, and M. Neschen. Fast logo detection and recognition in document images. In *Proceedings of the International Conference on Pattern Recognition*, pages 2716–2719, 2010.

[148] J. Liang, L. Phillips, and R. Haralick. An optimization methodology for document extracture on Latin character documents. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **23**[7]:719–734, 2001.

[149] C. Liou and H. Yang. Handprinted character recognition based on spatial topology distance measurement. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **18**[9]:941–944, 1996.

[150] J. Liu, X. Ding, and Y. Wu. Description and recognition of form and automated form data entry. In *Proceedings of the sixth International Conference on Document Analysis and Recognition*, pages 579–582, 1995.

[151] W. Liu and D. Dori. A protocol for performance evaluation of line detection algorithms. *Machine Vision And Applications*, **9**:240–250, 1997.

[152] Y. Liu and S. Srihari. Document image binarization based on texture features. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **19**[5]:540–544, 1997.

[153] D. Lopresti and G. Nagy. Automated table processing: An (opinionated) survey. In *Proceedings of the Third International Workshop on Graphics Recognition*, pages 109–134, 1999.

[154] D. Lopresti and G. Nagy. *A tabular survey of automated table processing*, **1941**, pages 93–120. Springer-Verlag, 2000.

[155] D. Lopresti, G. Nagy, and E. Barney Smith. Document analysis issues in reading optical scan ballots. In *Proceedings of the 9th International Workshop on Document Analysis Systems*, pages 105–112, September 2010.

[156] G. Louloudis, B. Gatos, N. Stamatopoulos, and A. Papandreou. ICDAR 2013 writer identification contest. In *Proceedings of the 2013 12th International Conference on Document Analysis and Recognition*, pages 1397–1401, 2013.

[157] G. Louloudis, N. Stamatopoulos, and B. Gatos. ICDAR 2011 writer identification contest. In *Proceedings of the 19th International Conference on Document Analysis and Recognition*, pages 1475–1479, 2011.

[158] S. Lu and C. Tan. Binarization of badly illuminated document images through shading estimation and compensation. In *Proceedings of the 2007*

International Conference on Document Analysis and Recognition, pages 312–316, 2007.

[159] H. Ma and D. Doermann. Word level script identification word level script identification for scanned document images. In *Proceedings of the Document Recognition and Retrieval XVIII (IST/SPIE International Symposium on Electronic Imaging)*, 2004.

[160] S. Mahmound. Arabic character recognition using Fourier descriptors and character contour encoding. *Pattern Recognition*, **27**[6]:815–824, 1994.

[161] J. Mantas. An overview of character recognition methodologies. *Pattern Recognition*, **19**[6]:425–430, 1986.

[162] K. Mardia and T. Hainsworth. A spatial thresholding method for image segmentation. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **10**[6]:919–927, 1988.

[163] U. Marti and H. Bunke. Using a statistical language mdoel to improve the performance of an HMM-based cursive handwriting recognition system. *International Journal of Pattern Recognition and Artificial Intelligence*, **15**[1]:65–90, 2001.

[164] U. Marti and H. Bunke. The IAM-database. *International Journal of Document Analysis and Recognition*, **5**:39–46, 2002.

[165] J. Matas, C. Galambos, and J. Kittler. Robust detection of lines using the progressive probabilistic Hough transform. *Computer Vision and Image Understanding*, **78**:119–137, 2000.

[166] Y. Min, S. Cho, and Y. Lee. A data reduction method for efficient document skew estimation based on Hough transformation. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 732–736, 1996.

[167] M. Mohammed and P. Gader. Handwritten word recognition using segmentation-free Hidden Markov modeling and segmentation-based dynamic programming techniques. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **18**[5]:548–554, 1996.

[168] G. Monagan and M. Roosli. Appropriate base representation using a run graph. In *Proc. of International Conference on Document Analysis and Recognition*, pages 623–626, 1993.

[169] B. Montgomery. The Iraqi secret police files: a documentary record of the Anfal geonocide. *Archivaria*, **52**:69–99, 2001.

[170] B. Montgomery. Returning evidence to the scene of the crime: why the Anfal files should be repatriated to Iraqi Kurdistan. *Archivaria*, **69**:143–171, 2010.

[171] D. Montgomery, E. Peck, and G. Vining. *Introduction to linear regression analysis (4e)*. John Wiley & Sons, Hoboken, New Jersey, 2006.

[172] V. Nagasamy and N. Langrana. Efficient diagram understanding with characteristic pattern detection. *Computer Vision, Graphics and Image Processing*, **30**[30]:84–106, 1985.

[173] V. Nagasamy and N. Langrana. Engineering drawing processing and vectorization system. *Computer Vision, Graphics and Image Processing*, **49**[3]:379–397, 1990.

[174] G. Nagy. Chinese character recognition: a twenty-five-year retrospective. In *Proceedings of the 1988 International Conference on Pattern Recognition*, pages 163–167, 1988.

[175] G. Nagy. Twenty years of document image analysis in PAMI. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**[1]:36–62, 2000.

[176] G. Nagy. Preprocessing document images by resampling is error prone and unnecessary. In *Proceedings of the Document Recognition and Retrieval XVIII (SPIE International Symposium on Electronic Imaging)*, page 86580U, 2013.

[177] G. Nagy. Invariant representation for rectilinear rulings. *Journal of Electronic Imaging*, **23**[6]:063011, 2014.

[178] G. Nagy and S. Seth. Hierarchical representation of optically scanned documents. In *Proceedings of the International Conference on Pattern Recognition*, pages 347–349, 1984.

[179] G. Nagy, S. Seth, and M. Viswanathan. A prototype document imaeg analysis system for technical journals. *Computer*, **25**[7]:10–22, 1992.

[180] J. Neumann, H. Samet, and A. Soffer. Integration and global shape analysis for logo classification. *Pattern Recognition Letters*, **23**[12]:1449–1457, 2002.

[181] W. Niblack. *An introduction to digital image processing.* Prentice Hall, 1986.

[182] L. O'Gorman. The document spectrum for page layout analysis. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **15**[11]:1162–1173, 1993.

[183] L. O'Gorman and R. Kasturi. *Document image analysis.* IEEE CS Press, 1995.

[184] U. Pal and B. Chaudhuri. Indian script character recognition: a survey. In 37, editor, *Pattern Recognition*, **9**, pages 1887–1899, 2004.

[185] U.. Pan and B. Chaudhuri. An imporved document skew angle estimation. *Pattern Recognition Letters*, **17**[8]:899–904, 1996.

[186] J. Parker. Gray level thresholding in badly illuminated images. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **13**[8]:813–819, 1991.

[187] T. Pavlidis and J. Zhou. Page segmentation by white streams. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 945–953, 1991.

[188] X. Peng, S. Setlur, V. Govindaraju, and R. Sitaram. Markov Random Field based binarization for hand-held devices captured document images. In *Proceedings of the Seventh Indian Conference on Computer Vision, Graphics and Image Processing*, pages 71–76, 2010.

[189] X. Peng, S. Setlur, V. Govindaraju, and R. Sitaram. Overlapped text segmentation using Markov Random Field and aggregation. In *Proceedings of the International Workshop on Document Analysis Systems*, pages 129–134, 2010.

[190] X. Peng, S. Setlur, V. Govindaraju, R. Sitaram, and K. Bhuvanagiri. Markov Random Field based text identification from annotated machine printed documents. In *Proceedings of the 2009 10th International Conference on Document Analysis and Recognition*, pages 431–435, 2009.

[191] D. Pérez, L. Tarazón, N. Serrano, F. Castro, O. Ramos Terrades, and A. Juan. The GERMANA database. In *Proc. of the International Conference on Document Analysis and Recognition*, pages 301–305, 2009.

[192] T. Pham, M. Delalandre, and S. Barrat. A contour-based method for logo detection. In *Proceedings of the International Conference on Pattern Recognition*, pages 718–722, 2011.

[193] I. Phillips and A. Chhabra. Empirical performance evaluation of graphics recognition systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **21**[9]:849–870, 1999.

[194] R. Plamondon and G. Lorette. Automatic signature verification and writer identification – the state of the art. *Pattern Recognition*, **22**:107–131, 1989.

[195] R. Plamondon and S. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. *Proc. IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**:63–84, 2000.

[196] W. Postl. Detection of linear oblique structures and skew scan in digitized documents. In *Proceedings of the International Conference on Pattern Recognition*, pages 687–689, 1986.

[197] I. Pratikakis, B. Gatos, and K. Ntirogiannis. ICDAR 2011 document image binarization contest (DIBCO 2011). In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1506–1510, 2011.

[198] I. Pratikakis, B. Gatos, and K. Ntirogiannis. Icfhr 2012 competition on handwritten document image binarization (H-DIBCO 2012). In *Proceedings of the International Conference on Frontiers in Handwriting Recognition*, pages 817–822, 2012.

[199] I. Pratikakis, B. Gatos, and K. Ntirogiannis. ICDAR 2013 document image binarization contest (DIBCO 2013). In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1471–1476, 2013.

[200] S. Rice, J. Kanai, and T. Nartker. A report on the accuracy of OCR devices. Technical report, University of Nevada Las Vegas, 1992.

[201] S. Rice, G. Nagy, and T. Nartker. *OCR: an illustrated guide to the frontier*. Kluwer Academic Publishers, 1999.

[202] J. Richarz, S. Vajda, and G. Fink. Towards semi-supervised transcription of handwritten historical weather reports. In *Proceedings of the 10th International Workshop on Document Analysis Systems*, pages 180 – 184, 2012.

[203] J. Roach and J. Tatem. Using domain knowledge in low-level visual processing to interpret handwritten music: an experiment. *Pattern Recognition*, **21**[1]:33–44, 1988.

[204] J. Rocha and T. Pavlidis. Character recognition without segmentation. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **17**[9]:903–909, 1995.

[205] M. Rusinol and J. Llados. Logo spotting by a bag-of-words approach for document categorization. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 111–115, 2009.

[206] H. Said, T. Tan, and K. Baker. Personal identification based on handwriting. *Pattern Recognition*, **33**:149–160, 2000.

[207] P. Sarkar, G. Nagy, J. Zhou, and D. Lopresti. Spatial sampling of printed patterns. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 1998.

[208] E. Saund. Scientific challenges underlying production document processing. In *Proceedings of the Document Recognition and Retrieval XVIII (IS&T/SPIE International Symposium on Electronic Imaging)Document Recognition and Retrieval Conference*, **7874**, 2011.

[209] A. Schlapbach and H. Bunke. Off-line writer identification using Gaussian Mixture Models. In *Proc. of the 18th International Conference on Pattern Recognition*, pages 992–995, 2006.

[210] A. Schlapbach and H. Bunke. A writer identification and verification system using HMM based recognizers. *Pattern Analysis and Application*, **10**:33–43, 2007.

[211] L. Schomaker and M. Bulacu. Automatic writer identification using connected-component contours and edge-based features of uppercase western script. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **26**:787–798, 2004.

[212] L. Schomaker and L. Vuurpijl. Forensic writer identification: a benchmark data set and a comparison of two systems. Technical report, Nijmegen, 2000.

[213] S. Seiden, M. Dillencourt, S. Irani, R. Borrey, and T. Murphy. Logo detection in document images. In *Proceedings of the International Conference on Imaging Science, System and Technology*, pages 446–449, 1997.

[214] F. Shafait and R. Smith. Table detection in heterogeneous documents. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, pages 65–72, 2010.

[215] H. Shamalian, H. Baird, and T. Wood. A retargetable table reader. In *International Conference on Document Analysis and Recognition*, pages 158–163, 1997.

[216] Z. Shi, S. Setlur, and V. Govindaraju. A model based framework for table processing in degraded document images. In *Proc. 12th International Conference on Document Analysis and Recognition*, pages 963 – 967, 2013.

[217] M. Shiridhar and F. Kimura. *Segmentation based cursive handwriting recognition*, pages 123–156. World Scientific Publishing Company, 1997.

[218] I. Siddiqi and N. Vincent. A set of chain code based features for writer recognition. In *Proc. the 10th international Conference on Document Analysis and Recognition*, pages 981–985, 2009.

[219] F. Slimane, S. Kanoun, H. El Abed, A. Alimi, R. Ingold, and J. Hennebert. ICDAR 2011 Arabic recognition competition: multi-font multi-size digitally represented text. In *Proceedings of the 2011 11th International Conference on Document Analysis and Recognition*, pages 1449–1453, 2011.

[220] E. Barney Smith. Scanner parameter estimation using bilevel scans of star charts. In *Proceedings of the 6th International Conference on Document Analysis and Recognition*, pages 1164–1168, 2001.

[221] R. Smith. A simple and efficient skew detection. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1145–1148, 1995.

[222] A. Spitz. Style-directed document recognition. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 611–619, 1991.

[223] A. Spitz. Skew determination in CCITT Group 4 compressed document images. In *Proceedings of the Symposium on Document Analysis and Information Retrieval*, pages 11–25, 1992.

[224] A. Spitz. Determination of the script and language content of document images. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **19**[3]:235–245, 1997.

[225] S. Srihari, S. Cha, H. Arora, and S. Lee. Individuality of handwriting. *Journal of Forensic Science*, **47**:1–17, 2002.

[226] S. Srihari and V. Govindaraju. Analysis of textural images using the Hough transform. *Machine Vision and Applications*, **2**[3]:141–153, 1989.

[227] P. Stathis, E. Kavallieratou, and N. Papamarkos. An evaluation survey of binarization algorithms on historical documents. In *Proceedings of the International Conference on Pattern Recognition*, pages 1–4, 2008.

[228] B. Su, S. Lu, and C. Tan. Combination of document image binarization techniques. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 22–26, 2011.

[229] P. Suda, C. Bridoux, B. Kammerer, and G. Maderlechner. Logo and word matching using a registration. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 61–65, 1997.

[230] H. Tamura. A comparison of line thinning algorithms from digital geometry viewpoint. In *Proc. of Fourth International Joint Conference on Pattern Recognition*, pages 715–719, 1978.

[231] T. Tan. Rotation invariant texture features and their use in automatic script identification. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **20**[7]:751–756, 1998.

[232] T. Taxt, P. Flynn, and A. Jain. Segmentation of document images. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **11**[12]:1322–1329, 1989.

[233] S. Theodoridis and K. Koutroumbas. *Pattern Recognition*. Academic Press, 2009.

[234] Ø. Trier and A. Jain. Goal-directed evaluation of binarization methods. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **17**[12]:1191–1201, 1995.

[235] Ø. Trier, A. Jain, and T. Taxt. Feature extraction methods for character recognition - a survey. *Pattern Recognition*, **29**[4]:641–662, 1996.

[236] Ø. Trier and T. Taxt. Evaluation of binarization methods for document images. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **17**[12]:312–314, 1995.

[237] Ø. Trier and T. Taxt. Improvement of "integrated function algorithm" for binarization of document images. *Pattern Recognition Letters*, **16**[3]:277–283, 1995.

[238] Ø. Trier, T. Taxt, and A. Jain. Recognition of digits in hydrographic maps: binary versus topographic analysis. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **19**[4]:399–404, 1997.

[239] L. Tseng and R. Chen. The recognition of form documents based on three types of line segments. In *Proc. of the 4th international Conference on Document Analysis and Recognition*, pages 71–75, 1997.

[240] S. Uchida and H. Sakeo. A survey of elastic matching techniques for handwritten character recognition. *Transactions on the Institute of Electronics, Information and Communication Engineers*, **88**[D8]:1781–1790, 2005.

[241] Luc Vincent. Google book search: Document understanding on a massive scale. In *Proceedings of the 9th International Conference on Document Analysis and Recognition*, pages 819–823, 2007.

[242] A. Vinciarelli. A survey of off-line cursive word recognition. *Pattern Recognition*, **35**[7]:1433–1446, 2002.

[243] A. Viterbi. Error bounds for convolutional codes and an asymtotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, **13**:260–269, 1967.

[244] F. Wahl, K. Wong, and R. Casey. Block segmentation and text extraction in mixed text/image documents. *Graphical Models and Image Processing*, **20**[4]:375–390, 1982.

[245] H. Wang and Y. Chen. Logo detection in document images based on boundary extension of feature rectangles. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1335–1339, 2009.

[246] S. Wang, H. Baird, and C. An. Document content extraction using automatically discovered features. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1076–1080, 2009.

[247] X. Wang. *Tabular abstraction, editing, and formatting.* PhD thesis, University of Waterloo, 1996.

[248] S. Watt and L. Dragan. Recognition for large sets of handwritten mathematical symbols. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 740–744, 2005.

[249] J. White and G. Rohrer. Image thresholding for optical character recognition and other applications requiring character image extraction. *IBM Journal of Research and Development*, **27**[4]:400–411, 1983.

[250] P. Xiu and H. Baird. Whole-book recognition using mutual-entropy-driven model adaptation. In *Proc. Document Recognition and Retrieval XVIII (IS&T/SPIE International Symposium on Electronic Imaging)*, 2008.

[251] P. Xiu and H. Baird. Whole-book recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **34**[12]:2467–2480, 2012.

[252] P. Xiu, D. Lopresti, H. Baird, G. Nagy, and E. Barney Smith. Style-based ballot mark recognition. In *Proceedings of the 2009 10th International Conference on Document Analysis and Recognition*, pages 216–220, 2009.

[253] Y. Yamazaki, T. Nagao, and N. Komatsu. Text-indicated writer verification using Hidden Markov Models. In *Proc. International Conference on Document Analysis and Recognition*, pages 329–332, 2003.

[254] H. Yan. Skew correction of document images using interline corss-correlation. *CVGIP: Graphical Models and Image Processing*, **55**[6]:538–543, 1993.

[255] S. Yanowitz and A. Bruckstein. A new method for images segmentation. *Computer Vision, Graphics and Image Processing*, **1**[82-95], 46.

[256] X. Ye, M. Cheriet, and C. Suen. A generic method of cleaning and enhancing handwritten data from business forms. *The International Journal on Document Analysis and Recognition*, **4**[7]:1184–1194, 2001.

[257] F. Yin, Q. Wang, X. Zhang, and C. Liu. ICDAR 2013 Chinese handwriting recognition competition. In *Proceedings of the 2013 12th International Conference on Document Analysis and Recognition*, pages 1464–1470, 2013.

[258] J. Yoo, M. Kim, S. Han, and Y. Kwon. Line removal and restoration of handwritten characters on the form documents. In *Proc. of International Conference on Document Analysis and Recognition*, pages 128–131, 1997.

[259] B. Yu. Automatic understanding of symbol-connected diagrams. In *Proceedings of the 3rd International Conference on Document Analysis and Recognition*, pages 803–806, 1995.

[260] B. Yu and A. Jain. A generic system for form dropout. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **18**[11]:1127–1131, 1996.

[261] B. Yu and A. Jain. A robust and fast skew detection algorithm for generic documents. *Pattern Recognition*, **29**[10]:1599–1629, 1996.

[262] B. Yu, A. Jain, and M. Mohiuddin. Address block location on complex mail pieces. In *Proceedings of the 4th International Conference on Document Analysis and Recognition*, pages 897–901, 1997.

[263] R. Zanibbi, D. Blostein, and J. Cordy. A survey of table recognition: models, observations, transformations, and inferences. *International Journal on Document Analysis and Recognition*, **7**[1]:1–16, 2003.

[264] Y. Zheng and D. Doermann. Background line detection with a stochastic model. In *Proc. of the Computer Vision and Pattern Recognition Workshop*, pages 424–431, 2003.

[265] Y. Zheng, H. Li, and D. Doermann. Machine printed text and handwriting identification in noisy document images. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **26**[3]:337–353, 2004.

[266] Y. Zheng, H. Li, and D. Doermann. A parallel-line detection algorithm based on HMM decoding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **27**[5]:777–792, 2005.

[267] Y. Zheng, C. Liu, and X. Ding. Form frame line detection with directional single-connected chain. In *Proc. of the International Conference on Document Analysis and Recognition*, pages 699–703, 2001.

[268] J. Zhou and D. Lopresti. Improving classifier performance through repeated sampling. *Pattern Recognition*, **30**[10]:1637–1650, 1997.

[269] G. Zhu and D. Doermann. Automatic document logo detection. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 864–868, 2007.

[270] E. Zois and V. Anastassopoulos. Morphological waveform coding for writer identification. *Pattern Recognition*, **33**:385–398, 2000.

Vita

Jin Chen, the son of Weiliang Chen and Caimei Wang, was born in Changsha, Hunan Province, China on August 24th, 1984. He was at the middle school attached to Hunan Normal University for middle and high school. After that, he became an undergraduate at Nanjing University of Science and Technology from 2002 to 2006. Then, he served as a server end software engineer at Shanda Network for a massive-play online game from 2006 to 2007. Since 2007, he joined Lehigh as a Ph.D student working with Professor Daniel Lopresti on handwriting biometrics and document image analysis. He enjoys research and communication with colleagues, friends, and researchers at meetings, conferences, and seminars.

Publications

1. J. Chen and D. Lopresti, "Ruling-Based Table Analysis for Noisy Handwritten Documents", *International Workshop on Multilingual OCR*, 2013.

2. J. Chen and D. Lopresti, "Alternatives for Page Skew Compensation in Writer Identification", *International Conference on Document Analysis and Recognition*, 2013.

3. J. Chen and D. Lopresti, "Model-Based Ruling Line Detection in Noisy Handwritten Documents", *Pattern Recognition Letters*, available online September 2012. DOI:10.1016/j.patrec.2012.08.008.]

4. J. Chen and D. Lopresti, "Exploiting Ruling Line Artifacts in Writer Identification", *International Conference on Pattern Recognition*, Japan, 2012.

5. J. Chen and D. Lopresti, "Model-based Tabular Structure Detection and Recognition in Noisy Handwritten Documents", *International Conference on Frontiers in Handwriting Recognition*, Italy, 2012.

6. J. Chen B. Zhang, H. Cao, R. Prasad, P. Natarajan, "Applying Discriminatively Optimized Feature Transform for HMM-based Off-line Handwriting Recognition", *International Conference on Frontiers in Handwriting Recognition*, Italy, 2012.

7. J. Chen and D. Lopresti, "Table Detection in Noisy Off-line Handwritten Documents",*International Conference on Document Analysis and Recognition*, China, 2011.

8. J. Chen and D. Lopresti, "A Model-based Ruling Line Detection Algorithm for Noisy Handwritten Documents", *International Conference on Document Analysis and Recognition*, China, 2011.

9. J. Chen , D. Lopresti, and Bart Lamiroy, "A Real-world Noisy Unstructured Handwritten Notebook Corpus for Document Image Analysis Research", *the Joint Workshop on Multilingual OCR and Analytics for Noisy Unstructured Text Data*, China, 2011.

10. J. Chen, W. Cheng, D. Lopresti, "Using Perturbed Handwriting to Support Writer Identification in the Presence of Severe Data Constraints", *Document Recognition and Retrieval XVIII (IS&T/SPIE International Symposium on Electronic Imaging*, USA, 2011.

11. E. Kavallieratou, D. Lopresti, and J. Chen, "Ruling Line Detection and Removal", *Document Recognition and Retrieval XVIII (IS&T/SPIE International Symposium on Electronic Imaging*, USA, 2011.

12. J. Chen, D. Lopresti, and E. Kavallieratou, "The Impact of Ruling-lines on Arabic Off-line Writer Identification", *International Conference of Frontier Handwriting Recognition*, India, 2010.

13. J. Chen, H. Cao, R. Prasad, A. Bhadwaj, and P. Natarajan, "Gabor Features for Arabic Handwriting Recognition", *International Workshop on Document Analysis System*, USA, 2010.

14. J. Chen and D. Lopresti, "On the Usability and Security of Pseudo-signatures", *Document Recognition and Retrieval XVII (IS&T/SPIE International Symposium on Electronic Imaging)*, USA, 2010.

15. J. Chen, D. Lopresti, and F. Monrose, "Toward Resisting Forgery Attacks using Pseudo-signatures", *International Conference on Document Analysis and Recognition*, Spain, 2009.

16. J. Chen, D. Lopresti, L. Ballard, and F. Monrose, "Pseudo-signatures as a Biometric", *International Conference on Biometrics: Theory, Applications, and Systems*, USA, 2008.

17. L. Ballard, J. Chen, D. Lopresti, and F. Monrose, "Biometric Key Generation using Pseudo-signatures", *International Conference of Frontier Handwriting Recognition*, Canada, 2008.