2011

Iterated Classification of Document Images

Chang An
Lehigh University

ITERATED CLASSIFICATION OF DOCUMENT IMAGES

by

Chang An

Presented to the Graduate and Research Committee

of Lehigh University

in Candidacy for the Degree of

Doctor of Philosophy

in

Computer Science

Lehigh University

May, 2011

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

_____
Date

_____
Henry S. Baird, Dissertation Director

_____
Accepted Date

_____
David S. Doermann

_____
Xiaolei Huang

_____
Daniel P. Lopresti

_____
Roger N. Nagel

Acknowledgments

I would like to express my gratitude to my advisor, Professor Henry Baird, whose guidance and insight continue to motivate me throughout this research. I am grateful for the constructive criticism and advice from the members of my committee Dr. David Doermann and Professors Xiaolei Huang, Daniel Lopresti, and Roger Nagel. I would like to thank the other students of the Pattern Recognition Research Lab at Lehigh University for their collaboration and discussion. My thanks to the friends who always stand by me however far away we are. I wish to thank my wife Ying Zhou and my aunt Changyu Zhang for taking care my loving daughter Audrey, so I can concentrate on my research. Finally, I owe my deepest thanks to my father Ping An, for his support, understanding, and encouragement all the way. In memory of my mother, Changxiu Zhang.

Contents

Acknowledgments **iv**

Abstract **1**

Principal Contributions of this Dissertation **3**

Frontispiece **4**

1 Introduction **7**

2 Literature Review **12**

2.1 Cascading Classifiers 12

2.2 Conditional Random Fields 13

2.3 Matched Wavelets 14

2.4 Texture Based Segmentation 15

2.5 Constrained Connectivity Paradigm 16

2.6 Hierarchical Threshold Segmentation 16

2.7 Mathematical Morphology 17

2.8 Competing Ground-Truth Policies 18

2.9 Summary and Discussion of Iterated Classifiers 21

3 The Proposed Approach: Iterated Classification 22

3.1 Post-Classification 24

3.2 Repeated Classification Trained Only on First Stage Results 27

3.3 Iterated Classification 33

4 Classification Algorithms 35

4.1 Brute-force k-Nearest Neighbors 35

4.2 Hashed k-D Tree Classifier 35

4.3 Decimation of Training Data 37

4.4 Feature Extraction for DICE 37

4.4.1 Feature Combination 41

4.5 Feature Extraction for Iterated Classifiers 42

4.6 Systematic Exploration of Scale of Features 43

5 Investigation of Ground-Truth Policies 44

5.1 What to Ground Truth? 44

5.2 Blank Space 46

5.3 Overlapping Content 47

5.4 Machine Print in Photographs 47

5.5 Difficult Shapes 48

5.6 The Effect of Tighter Zoning 49

5.7 Ground-truth Policies 49

5.8 Morphological Expansions of Pixel-accurate Truth 50

5.9 Experiment Design 50

5.10 Experimental Results 51
5.11 Problems with Pixel-Accurate Ground Truth 56
5.12 Conclusion 59

6 Experiments **60**
6.1 Instability and Workarounds 62
6.2 Preliminary Experiment with Loose Ground Truth 67
6.3 Preliminary Experiment with Tight Ground Truth 69
6.4 Statistical Significance of Claimed Improvements 74
6.5 Final Experiment 78
6.5.1 Motivation 78
6.5.2 Results 79
6.5.3 Run Times 81
6.5.4 Conclusions 81
6.6 ICDAR 2009 Page Segmentation Competition 85

7 Performance Analysis of Iterated Classification **87**
7.1 Analysis of the Second-Stage Classifier 89
7.2 Analysis of Classifiers Following Second-Stage 90
7.3 Comparison of Iterated Classication with Repeated One 92

8 High Recall Document Content Extraction **95**
8.1 Iterated Classifiers Improve Recall and Precision 95
8.2 Proposed Testing Policy Changes 96

8.2.1 Multistage Voting Rule 97
8.2.2 Truthing and Scoring Policy for Blank Pixels 97
8.3 Experimental Results 98
8.4 Discussion 103

9 Conclusion **107**

Bibliography **111**

Vita **121**

List of Tables

4.1 Error Rate Table: Varying the Size of Feature Extraction Window . 43

5.1 Accuracy Table of Ground-Truth Policies 55

5.2 Error Rate As a Function of Stages of Different Ground-Truth Policies 56

6.1 Results of T-test 78

List of Figures

1 Frontispiece: Schematic Methodology of Iterated Classification . . . 5

2 Frontispiece: Iterated Classification Example of An Entertainment Magazine Image 6

1.1 An Example of DICE Classification 10

3.1 Examples of DICE Classification Errors 23

3.2 Collage of Subset of Test Images 25

3.3 Collage of Classification Results 26

3.4 Schematic Methodology of Post-Classification 28

3.5 Schematic Methodology of Repeated Classification 29

3.6 Example of Repeated Classification Improving Handwriting 31

3.7 Example of Repeated Classification Expanding Handwriting Erroneously 32

3.8 Schematic Methodology of Iterated Classification 34

4.1 Example: Effect of Decimation 38

5.1 Error Rates and Recall of Different GT Policies 52

5.2 Precision and the number of GT pixels of Different GT Policies . . 53

5.3 Error Rate As a Function of Stages of Different Ground-Truth Policies 57

6.1 Example of Instability 63
6.2 Instability: Errors as a Function of Stages 64
6.3 Instability Cause Analysis 65
6.4 Example of Workarounds 66
6.5 Iterated Classification Example (on Loose Truth): A Newspaper Image 68
6.6 Iterated Classification Example (on Loose Truth): A Magazine Image 70
6.7 Error Rate of Iterated Classification: Loose Ground Truth 71
6.8 Iterated Classification Example (on Tight Truth): A Sports Magazine Image 73
6.9 Iterated Classification Example (on Tight Truth): An Entertainment Magazine Image 75
6.10 Error Rate of Iterated Classification: Tight Ground Truth 76
6.11 Error Rate of Iterated Classification: Experiment with 157 Images . 80
6.12 Iterated Classification Example: An ICDAR2009 Page Segmentation Competition Image 82
6.13 Iterated Classification Example: A Minute Page 83
6.14 Iterated Classification Example: A DIBCO09 Test Image 84
6.15 F-measure Report of the ICDAR 2009 Page Segmentation Competition 86

7.1 Illustration of Analysis: Iterated Classification Converges Linear Boundary to Ground Truth 88

7.2 Simulation of Repeated Classification 94

8.1 Recall and Precision of Each Content Type as a Function of Stages . 96

8.2 Recall and Precision: Accepting Blank Pixels 101

8.3 Recall and Precision of Handwritten: Obtained by Different Policy Changes 102

8.4 Example of Improving Recall: A Complex Magazine Image 1 . . . 104

8.5 Example of Improving Recall: A Complex Magazine Image 2 . . . 105

Abstract

We investigate a family of pattern classification methodologies for image processing using *iterated classification*, that is, using a sequence of classifiers, each trained separately on the training-data results of the preceding classifier, and each guided by the same ground truth. We apply iterated classification to improve document image content extraction: that is, the location and segmentation of handwriting, machine-printed text, photographs, and blank space. We try to achieve high-accuracy pixel-accurate segmentation: that is, each pixel in a document image is assigned a class; there is no "region" model, and so results are not constrained to arbitrary region shapes such as rectangles (a restriction which dominates most of the R&D literature). Because classification is pixel-accurate, the output image is "false color", where colors represent content classes: thus, both the input and output of our algorithms are images. We describe large-scale experiments which reveal that iterated classifiers can increase recall of all content types, with little loss of precision. We also introduce two policy changes: (1) a multi-stage voting rule; and (2) a scoring policy that considers blank pixels to be a "don't care" class. These changes are

realistic and improve both recall and precision, achieving 89% recall and 87% precision (at least) among three content types: machine-print, handwriting, and photographs. We have found that iterated classification is sensitive to the ground-truth policy, such as “loose”, “tight”, and pixel-accurate policies. We have compared the accuracy of all three truthing policies, and report that tight truth supports higher accuracy than loose truth, and pixel-accurate truth yields the highest accuracy. Experiments on a diverse and highly challenging test set of 83 document images show that tighter ground-truth reduces per-pixel classification errors by 45% (from 38.9% to 21.4%). Latest experiment on a test set of 157 document images shows that iterated classifiers continue to drop per-pixel classification errors by 24.5% (from 20.2% to 15.2%). Evidence from both experiments and simulation suggests that iterated classification converges to the ground-truth; we have analyzed special cases suggesting reasons why iterated classifiers tend to converge to the ground truth.

Principal Contributions of this Dissertation

- We present a strategy for document image segmentation using a series of post classifiers: “iterated classification.”
- Our iterated classification approach allows pixel-accurate classification and minimizes arbitrary manually chosen rules.
- We have carried out large-scale experiments on classifier systems implementing this approach, showing that they are capable of reducing errors – and, increase both recall and precision – significantly on difficult test sets.
- We present a formal analysis of iterated classification, comparing it with “repeated classification,” and supported by simulations, which suggest reasons why it tends, over a series of classification stages, to converge towards ground truth.
- We compare and contrast the effects of choices of competing ground-truth policies on classifier performance for document image classification.

Frontispiece

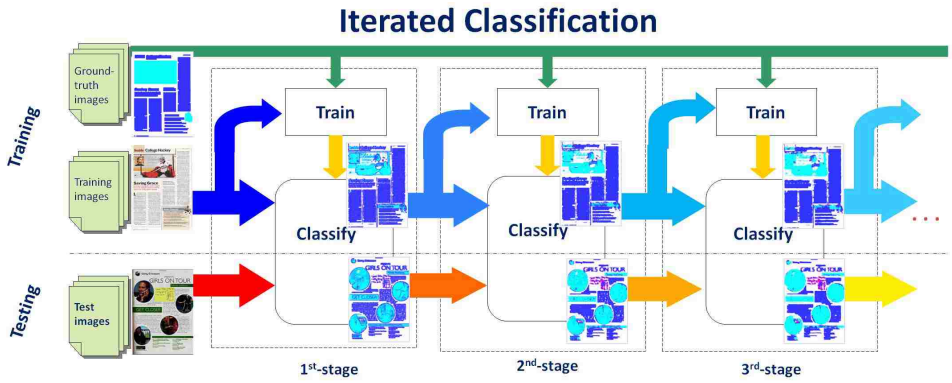


Figure 1: Schematic methodology of iterated classification. The same ground truth is passed to every training phase. Classification results are passed from one classifier to its successor for training and classification. Note that each classifier is, in general, different from one another.

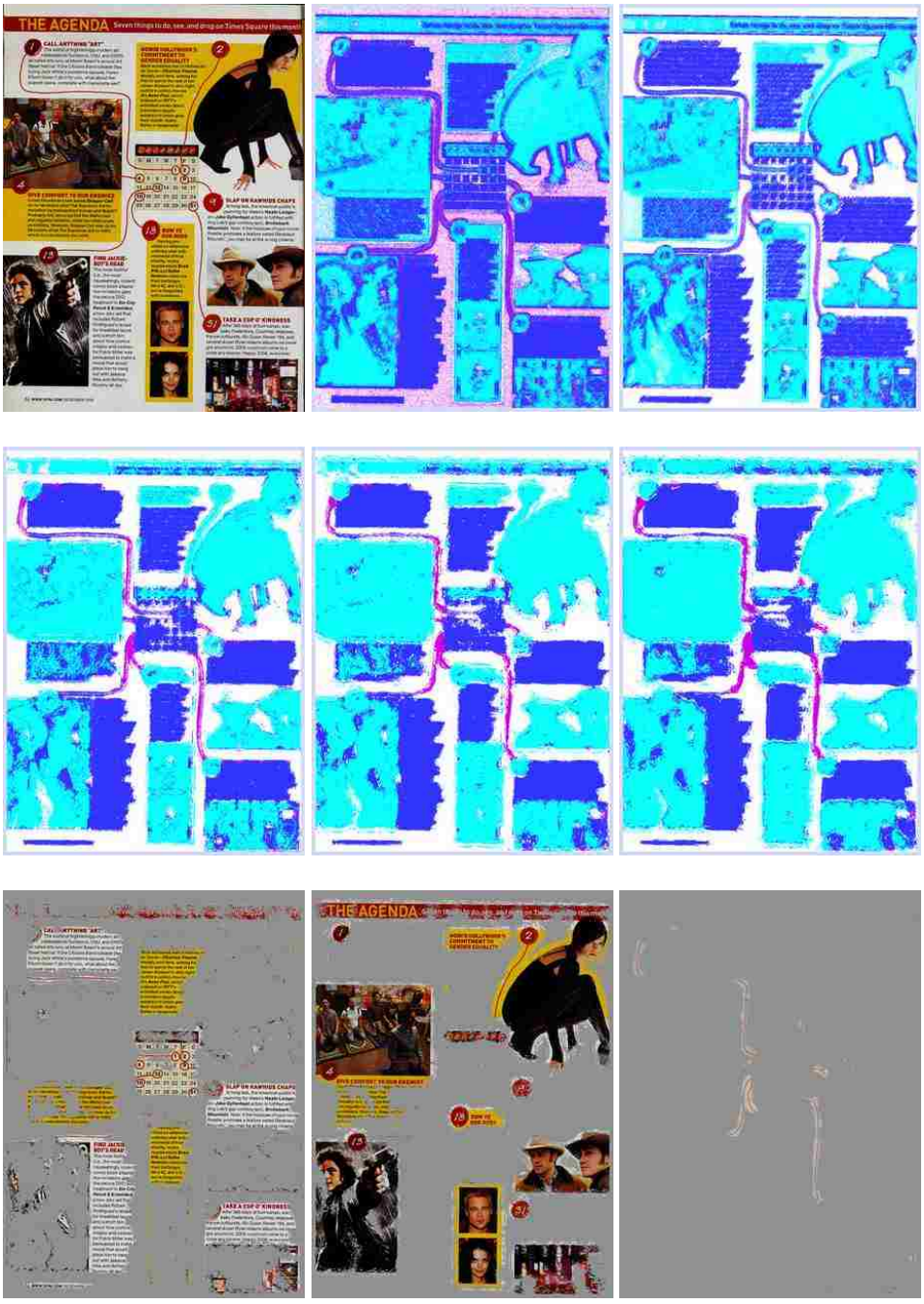


Figure 2: Iterated classification example of an entertainment magazine image. Upper left is the full color original image, followed by the results of iterated classification. The last three images are final MP, PH and HW masks.

Chapter 1

Introduction

Image processing is signal processing for which the input signal is an image, such as photographs or video frames; the output of image processing can be either an image or a set of characteristics or parameters related to the image. Most image-processing techniques involve treating the image as a two-dimensional signal and applying signal-processing techniques to it.

Image segmentation is one type of image processing. The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze [SS01]. Commonly, a segmentation is defined as a partition of the image into disjoint subsets of pixels (subareas of the image) such that each subset is of a single type, function, or content [HP74, HP76].

The vast and rapidly growing scale of document image collections has been compellingly documented [WMB99]. Information extraction [Ish01] and retrieval [MC00] from document images is an increasingly important R&D field at the interface between document image analysis (DIA) and information retrieval (IR).

Our research has focused on investigating versatile algorithms for *document image content extraction*, that is segmenting the images into machine printed text, handwriting, photographs, etc.

The **document image content extraction** problem can be defined as:

> *Given* an image of a document,
>
> *find* subsets of pixels containing machine-printed text, handwriting, photographs, etc.

We approach this problem in its full generality, attempting to cope with the richest diversity of documents and image types. Colleagues in our lab have reported preliminary results in the development of highly versatile [BMN+06] and voracious [CB06, Cas06] classifiers for this problem domain. Types of document images that we accept include color, grey-level, and bilevel (black-and-white); also, many sizes or resolutions (digitizing spatial sampling rates); and in many of a wide range of file formats (TIFF, JPEG, PNG, etc). We convert all image file formats into a PNG file in the HSL (Hue, Saturation, and Luminance) color space; bilevel and greylevel images convert to HSL images with fixed values for hue and saturation. We have access to a database of over 9000 sample page images containing the following types of content: machine print (MP), handwriting (HW), photographs (PH), line Art (LA), math notation (MT), maps (MA), engineering drawings (ED), chemical drawings (CD), "junk" (JK, *e.g.* margin and gutter noise), and blank (BL). These include samples of each content type across a wide range of languages (including English, Chinese and Arabic) and image qualities and from several historical periods. The wide range of images is illustrated in Figure 3.2.

We have adopted the policy of classifying individual *pixels*, not *regions* as most

previous document segmentation research have done. This avoids the arbitrariness and restrictiveness of limited families of region shapes, as illustrated in Figure 1.1. We are also strongly motivated by the work of Shafait, Keysers and Breuel on pixel-accurate representation of segmentation results [SKB06].

In Figure 1.1, a test image is shown on the left and the results of classification next on the right where the content classes are shown in color: machine print (MP) in dark blue, handwriting (HW) in red, photographs (PH) in light blue-green, blank (BL) in white, and unclassified in light grey. Each image possesses a thin border of unclassified pixels (difficult to see at this resolution) due the fact that feature extraction requires a region of a minimum size. Some other pixels remain unclassified due to sparsity of training data.

Both training and test datasets consist of pixels labeled with their ground-truth class (one of MP, HW, PH, BL). Each pixel sample is represented by scalar features extracted by image processing of a small region centered on that pixel; these features are discussed in detail in Section 4.4. Our work is built on two automatically trainable classification technologies developed by other students in our lab: brute-force 5-Nearest Neighbors (5NN) and fast approximate 5NN using hashed k-d trees [CB06, Cas06, BMA07].

My work is built directly on the results of work by several other researchers in our lab, earlier researchers' contributions to my entire system are listed as below:

- Brute-force 5-Nearest Neighbors (5NN) Classifiers by Don Delorenzo [CB06];
- Approximate 5NN using hashed k-d trees by Matthew Casey [Cas06];
- Random decimation by Michael Moll [BMA07];

(a) color test image

(b) "false color" classification result "image"

Figure 1.1: A document image with a complex non-rectilinear page layout. Our policy of classifying pixels has the advantage of adapting to arbitrary layouts with non-rectilinear region shapes (here, regions with circular-arc boundaries). The original image (a) is in full color. In the results of classification (b), machine print (MP) is dark blue, handwriting (HW) red, photographs (PH) light blue-green, blank (BL) white, and unclassified pixels are shown in light grey.

- Feature Selection Focused within Error Clusters [WB08];
- Bin decimation by Dawei Yin [YBA10, YAB10].

The remainder of this dissertation is organized in the following manner: Chapter 2 contains the literature review relevant to the motivation of our work. Chapter 3 discusses our proposed approach and details of its scheme. The details of algorithms and implementation is described in Chapter 4. The investigation of competing ground truth policies are discussed in Chapter 5. The results of most of our experiments are discussed in Chapter 6. In Chapter 7, we present a formal analysis of the performance of iterated classification. Chapter 8 is a discussion of policy changes to enhance recall. Finally Chapters 9 highlight conclusions and future work.

Chapter 2

Literature Review

In this Chapter we review other approaches that are relevant to the ones we investigated.

2.1 Cascading Classifiers

Our technique of iterated classification is similar in broad outline to cascading classifiers [AK98, KA00]. Cascading classifiers, introduced by Alpaydin and Kaynak, are a sequence of classifiers ordered in terms of increasing complexity and specificity such that early classifiers are simple and general whereas later ones are more complex and specific, being localized on patterns rejected by its preceding classifier. An example of a cascading system is as follows. The first classifier is a single layer perceptron (SLP) and the next classifier, is a multilayer perceptron (MLP), which is trained by focusing on training patterns not covered by the SLP. The remaining few patterns will be treated as exceptions and covered by an expensive instance-based technique, *e.g.* kNN. The cascading algorithm was tested on eight different

databases from the UCI repository [BM], The result showed that MLP reached accuracies of 76.4%, 89.1% and 95.2% on recognition of letters, optical-based hand-written digits and pen-based hand-written digits respectively; kNN increased the accuracies to 93.4%, 96.5% and 97.7%.

Cascading algorithm has the strength of increasing accuracy without the concomitant increase in complexity and cost. However, determining the confidence threshold needed for each stage is heuristic. Our iterated classifiers have these differences with cascading classifiers: we train on the results of classification, not on the original images; and we reclassify every sample, not merely rejected samples.

2.2 Conditional Random Fields

Conditional random fields [JLP01] (CRFs) are a framework for building probabilistic models to segment and label sequence data. A CRF can be viewed as a undirected graphical model that defines a single log-linear distribution over label sequences given a particular observation sequence [Wal04]. Formally, let G = (V,E) be a graph such that there is a node v ϵ V corresponding to each of the random variables representing an element Y_v of **Y**, and **X** be the random variable representing observation sequences. If each random variable Y_v obeys the Markov property with respect to G, then (**X,Y**) is a conditional random field.

Other researchers have attacked this problem of fine-grain classification without restricting region shape. Nicolas and Dardenne *et al* [SNH07] adapted and applied conditional random fields (CRFs) to document image segmentation. In the phase of feature extraction, they defined three feature functions: a local feature function

that takes only into account features extracted on the observed image, a contextual feature function that takes only into account the local conditional probability densities on the label field in a neighborhood, and a global feature functions that extract the global label configuration over a larger neighborhood than that taken by the contextual feature function. They used Multilayer Perceptron (MLP) to model each feature function because they are fast and provide good generalization properties even in high dimensional spaces. They also investigated the use of MLP as a combination function, and used the backpropagation algorithm to train all the MLP to determine the weights of each MLP.

Their insights of taking into account of neighborhood contextual information is similar to ours. Another similarity is that they also extracted features on pixel level, but they classified 3x3 region to decrease the computation. The drawback of their method is the prohibitive time required for training the MLP. One limitation is that they only experimented on handwritten drafts of Flaubert, not on versatile images containing several content types. Still, we can learn from their method.

2.3 Matched Wavelets

Kumar and Gupta *et al* [KGK+07] used matched wavelets to develop the globally matched wavelet filters. Their method works in two phases. In the first phase, the matched wavelets scheme is extended for the segmentation of document images into text, background and picture components. They used three Fisher filters, each optimized for a two-class classification problem. In the second phase, to refine the obtained segmentation results, they exploited the contextual information by using

a Markov random field (MRF) formulation-based pixel labeling scheme; and they attained MRF energy minimization using the alpha-expansion algorithm proposed in [YZ04, KZ04].

They coped with not only grey images but also color images, and classified per-pixel. Another similarity with our work is that they did not need any information about the font size or format of the text in the image. They tested their method on a test set of 33 images taken from scanned images and different website. The accuracy of their method is high – it reaches 93.8% in one of the images. The average time of their approach on the test set is 158 seconds, which is considered fast. However, because each filter they use only deals with two-class problem, as the number of classes increases, the number of filters increases. We can learn from their approach.

2.4 Texture Based Segmentation

Etemad and Doermann *el al* [EDC97] proposed a texture based algorithm for layout-independent document page segmentation. They regarded text, image and graphics regions in a document image as three classes of textures. In their approach a wavelet packet tree is built for texture based multiscale feature extraction, and six features are selected. A multilayer neural network is trained and decision integration is utilized to obtain “soft classification”, that is, image subblocks can be classified as belonging to multiple classes. Their approach performs well on complex document layouts, and is robust to noise and page skew. This texture based document segmentation scheme may be more complex than other methods, but it has a wider range of

applicability. This approach has the advantage that a majority of its calculation and decisions are made independently and in parallel without iterative stages. Therefore it can be well adapted to distributed and parallel architecture.

2.5 Constrained Connectivity Paradigm

Pierre Soille introduced an image partitioning and simplification method based on the constrained connectivity paradigm [Soi08]. According to this paradigm, two pixels are said to be connected if they satisfy a series of constraints defined in terms of simple measures such as the maximum gray-level differences over well-defined pixel paths and regions. The resulting connectivity relation generates a unique partition of the image definition domain.

Soille's method has the strength of avoiding the arbitrariness of region shapes, on which aspect it is similar to our method. However, this method only offers a "low-level" answer in the sense that they generate a partition of the image into "puzzle pieces" that still need to be assembled for the purpose of detecting specific objects defined in the context of application.

2.6 Hierarchical Threshold Segmentation

Peak and Tag presented a technique called hierarchical threshold segmentation (HTS) to solve the problem of segmentation of satellite cloud images [PT94]. Their approach applied artificial intelligence to reasoning about the sizes and shapes of the emergent regions during the segmentation process. Two key features are used: the first one is the number of pixels on the perimeter; the second is the ratio of the

number of pixels inside the region to the number of boundary pixels.

Their idea of split and merging of regions is relevant to our problem in that we need to obtain pure regions by reclassifying mixed pixels of different content classes. However, their problem is simpler than ours: the shape and the size of clouds are restricted, while the shape and the size of image partitions are arbitrary.

2.7 Mathematical Morphology

Since we classify every pixel, our classifiers are similar to many image processing methods, such as mathematical morphology [SS94]. Morphological processing refers to certain operations where an object is *hit* with a *structuring element* and thereby reduced to a more revealing shape [Jai89]. Most morphological operations can be defined in terms of two basic operations, *erosion* and *dilation* [Ser82]. Useful morphological transforms that are derived from the basic erosion and dilation operations include *hit-miss*, *opening*, *closing*, *boundary convex hull*, *skeletoning*, *thinning*, *thickening* and *pruning*.

Park and Lee investigated features in 1-D signal on many scales and proved a "causal" property of scale-space (*i.e.* no new feature points are created as scales get larger) for each of the morphological operations: opening, closing, and alternating sequential filtering [PL96]. In order to prove that, they refined the standard definition of zero-crossings so as to allow signals with a certain singularity, and use them to define feature points. They claimed and explained that morphological opening can not satisfy causality for two-dimensional gray-scale images.

Morphological transforms are sometimes considered "low-level" [Dou92b]. The

qualifier “low-level” means that the implementation of transformations are served as elementary steps when solving practical image analysis problems. This does not mean that these transformations are simple; on the contrary, some of the operations are complex. However, from a user’s perspective, these transformations share the characteristics of being easily and intuitively understandable.

2.8 Competing Ground-Truth Policies

The availability of a good ground-truth policy for evaluation is crucial to the success of image analysis. Although many ground-truth policies have been proposed, agreement on details has been hard to reach. Nevertheless, some researchers agree that for different tasks, different ways for measuring overall performance are desired [LL10]. In this section, we briefly summarize recent debates on competing ground-truth policies.

Clavelli et al presented a nice survey of evaluation problems for the text extraction [ACL10]. Several algorithms for text extraction from complex color images have been described in the literature [LZ00, KA07, PGM+04, KJK03, RM07]. Some papers use a subjective “eye-ball” standards for lack of ground-truthed datasets and corresponding performance evaluation methods [LZ00, KA07]. Some other papers rely on optical character recognition (OCR) error rate to evaluate text extraction [PGM+04]; of course any defects in the OCR system are also reflected in error rate, thereby affecting the evaluation of the text extraction.

Basically, text extraction approaches can be sorted into two main categories: texture-based [KJK03] and connected-component (CC) based [RM07]. Texture

based approaches aim to locate text zones in image. Performance evaluation of texture based approaches is typically based on calculating the overlapping ratio between detected text zones and the ground truth [LPS+05, WJ06, RM07]. This evaluation scheme was originally conceived for layout analysis algorithms [LPS+05, LPH97, AKB06]. Since the results of text extraction are not necessarily in the form of bounding boxes, it is difficult to quantify the effects of each step of an algorithm, as pointed out in [RM07]. Connected component approaches aim to produce a pixel-level segmentation of the image by separating text items (*e.g.* characters, words, text-lines, etc) into CCs. The performance evaluation should be able to assess not only the text location step but also the post-processing steps towards text extraction. Nevertheless, in [RM07] the authors have to revert to bounding box overlapping measures that assess only the text location performance.

Clavelli et al then proposed a comprehensive framework for the evaluation of text extraction methods at multiple levels: pixel-level segmentation, character restoration, text localization, word and text-line extraction.

Existing performance evaluation frameworks for text extraction generally do not work with pixel-accurate ground truth. Nitrogiannis et al [NGP08] proposed a framework for the purpose of evaluating thresholding results. In [MBA08] we discussed the challenges and difficulties of defining a pixel-accurate ground truth for the purpose of document-image segmentation. We showed in large-scale experiment that "tight" truth support better classification result than "loose" truth. A PARC research team agreed with this notion, and built an efficient manual truthing tool, PixLabeler [SLS09].

Utilizing PixLabeler, Barney Smith explored the variability that inevitably arises

when images are ground truthed by humans and how this might affect the evaluation of automated binarization algorithms [Smi10]. A series of experiments were run using test images of Document Image Binarization Contest (DIBCO 2009) [GNP09], the ground truth images used in DIBCO, the re-ground-truthed images generated at BSU, and the results of five competing binarization algorithms submitted to DIBCO. The semi-automatically generated ground truth images used in DIBCO were compared with BSU's fully manually ground truthed images. The two sets did not match as closely as might have been expected. For a single image that was ground truthed multiple times, a larger variability was observed among different ground truthing operators. Four direct evaluation metrics were used in this study: F-measure (FM), negative rate metric (NRM), peak signal-to-noise ratio (PSNR) and normalized cross correlation (NCC). The variable and inconsistent performance of NRM indicates that it should not be used for this type of evaluation. It was observed that the manually ground truthed images were on average comparable to the top DIBCO competing results. On certain images the competing automatic binarization alogrithms agreed with the DIBCO ground truth more closely than the BSU manual ground truthed images. Barney Smith summarized this: "This may indicate that in a contest, no differentiation among algorithms can be made above a certain level of fit."

We face similar issues as Barney Smith does when we use PixLabeler to extract pixel-accurate ground truth. The first difficulty is the binarization of grey images and color images, especially when ground-truthing handwriting images: it is sometime hard to decide where the boundary between text and back ground is. The second is the overlapping pixels of different foreground content types: a typical

case is text (machine-print or handwriting) printed on photographs. One thing to point out is that in color images (*i.e.* magazine pages), foreground pixels are not necessarily darker than background pixels.

2.9 Summary and Discussion of Iterated Classifiers

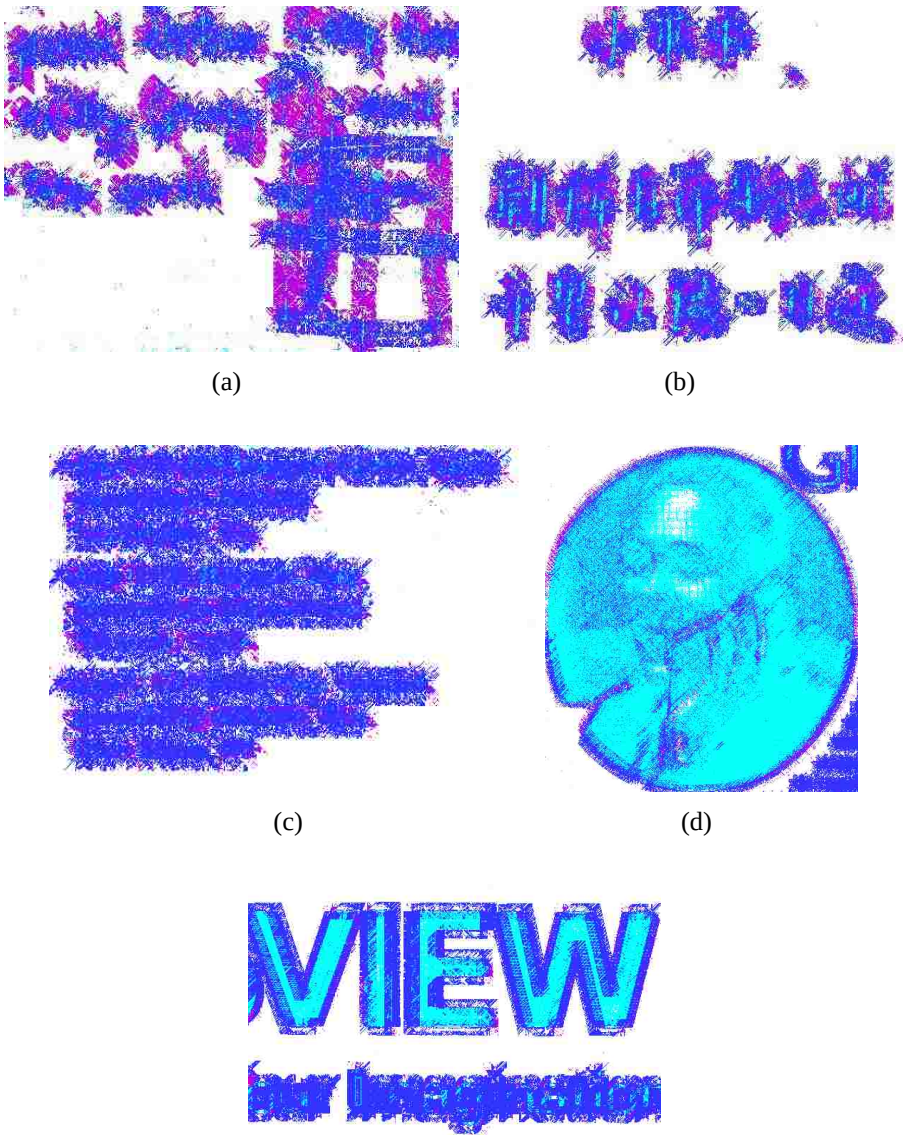
In summary, few previous approaches attack the image segmentation problems challenged by (a) versatility (*i.e.* content classes) and (b) arbitrariness (*i.e.* region shapes and boundary shapes) against a wide range of documents (*i.e.* black & white, grey and color), (c) in a automatic way (*i.e.* trainable and data-driven). Pixel accurate segmentation is growing in popularity because it avoids the arbitrariness and restrictiveness of shapes. Our proposed iterated classification method gains its strength from pixel-accurate segmentation, and has been tested on an extensively wide range of documents; and it is data driven.

Chapter 3

The Proposed Approach: Iterated Classification

We explain in this Chapter the motivation for and the design of iterated classification.

We showed in the Introduction that our document image content extraction (DICE) algorithm is able to avoid arbitrariness and restrictiveness of region shapes. Examples of typical DICE classification errors are shown in Figure 3.1. In these examples, the ground truth is all of one content class, but the DICE classification "mixes" content types together. Clearly, further improvement is desirable and should be possible. For these kinds of errors, researchers might suggest image processing technique of average voting within a window. We object to this approach on the grounds that it requires engineering choices of arbitrary region shapes and manually crafted rules such as window sizes. This observation motivates us to find other approaches to improve the accuracy yielded by DICE algorithm.



(a) (b) (c) (d) (e)

Figure 3.1: Examples of DICE classification errors. Example (a) shows clusters of HW pixels that are heavily misclassified as MP. Example (b) shows HW pixels misclassified as both MP and PH. Example (c) shows MP pixels lightly misclassified as both HW and PH. Example (d) shows PH pixels located in the center of a photo are sparsely misclassified as MP; and those lying on or near the boundaries are heavily misclassified as MP. Example (e) shows MP pixels of large font size letters are heavily misclassified as PH, except for those lie on the boundaries.

Basically, we can approach the problem from three directions: one is to choose better features, which Sui-Yu Wang had investigated in her research work [WB08]; another is to improve the algorithm of the classifiers, which Dawei Yin has investigated [YBA10, YAB10]; a third is to use post-classification, which is my principal contribution, and will be introduced in detail in this Chapter.

3.1 Post-Classification

We decided to choose post-classification as our approach for two reasons. The first reason is that we want to avoid restricting arbitrary region shapes; therefore, we want the post-processing method to continue to produce pixel-accurate results. The second reason is that we want to avoid unsuitable extra manual decisions when recomputing pixel-accurate content classes. Since we want the post-processing method to be scalable to large datasets, we want it to require as few engineering intervention as possible. In another word, we want this method to be data-driven as much as possible.

We define the post-classification problem as follows:

Given: pixel-accurate classification results for a document image.

find: a reassigned labeling that yield higher accuracy.

We have designed a trainable post-classifier that operates on the output of the preceding classifier, guided by the same ground truth. That is, we want the behavior of the system to be determined by training data alone. Of course we can not escape all engineering choices, *e.g.* we have to choose features, and the size of windows within which features are extracted. A diagram of the post-classifier is shown in

3.1. POST-CLASSIFICATION

Figure 3.2: Examples of documents that are tested in our experiments.

3.1. POST-CLASSIFICATION

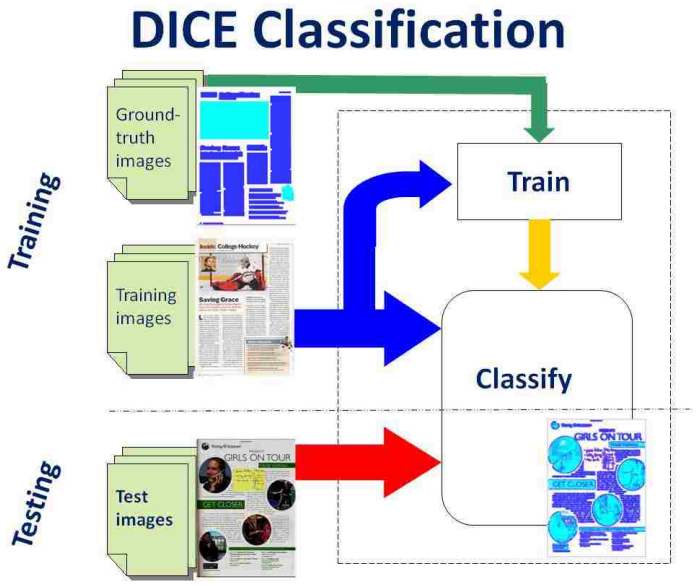
Figure 3.3: Examples of "false-color" classification results of the wide range of documents that are tested in our experiments.

Figure 3.4.

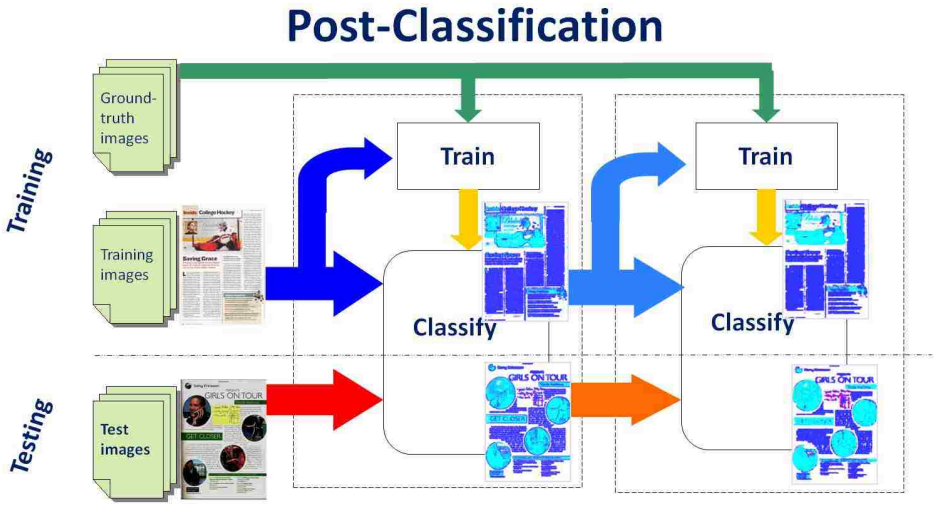
3.2 Repeated Classification Trained Only on First Stage Results

As discussed in section 3.1, we have built a post-classifier that operates on the output of the DICE classifier, guided by the same ground truth, and reclassifies the test images. The post-classifier takes "false color" images as input and yields also "false color" images as output. This characteristic suggested to us that we repeat the post-classification process: put the output of the post-classifier back as its input, as perform another round of classification. Of course this process can be repeated indefinitely: we call this approach *Repeated Classification* (RC). We will call the output of the DICE classifier the *first stage* output; the output of the immediately following post-classifier is called the *second stage* output, followed by the *third stage* output, etc. Note that the second stage classifier and the third stage classifier are the same, as well as the following stage classifiers. A diagram of repeated classification is shown in Figure 3.5.

Repeated classifier can sometimes succeed in fixing errors from earlier stage. An example of improved results on handwriting is shown in Figure 3.6. The test image contains only handwriting text. In stage 1, we can see that many pixels are misclassified as MP, especially these lying on the strokes. Most of their surrounding neighbors are classified as HW. In stage 2, we can see the clusters of MP shrink, that is, many pixels that are classified as MP in stage 1 are reclassified as HW. This correction continues in following stages. By stage 5, almost all MP pixels are

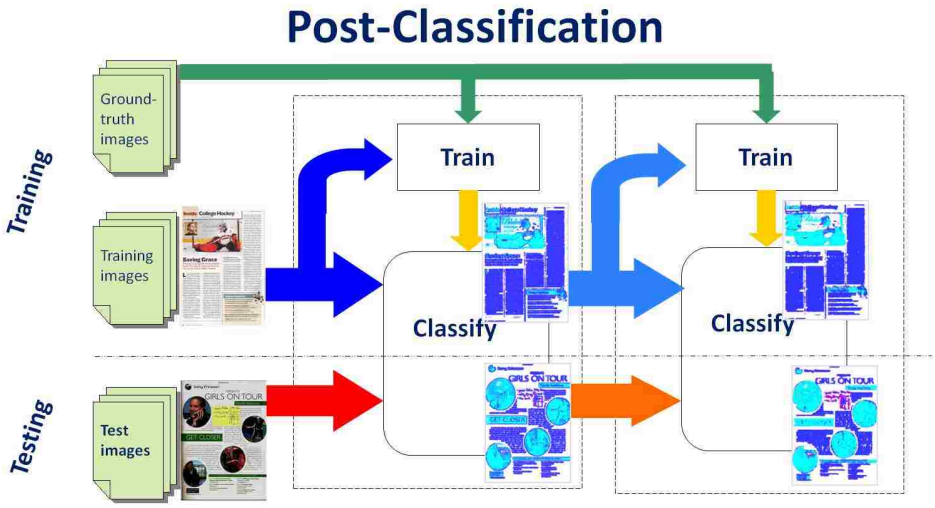


(a) DICE Classification

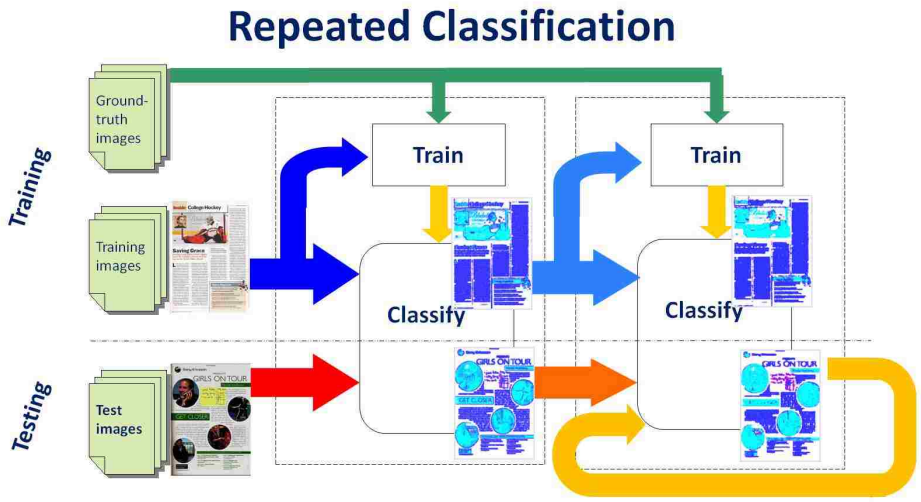


(b) Post-Classification

Figure 3.4: Schematic methodology of post-classification. To enable post-classification, DICE classifier is modified in the way that not only the test images but also the training images are classified. The results, along with the ground truth, are then passed to the post-classifier as input.



(a) Post-Classification



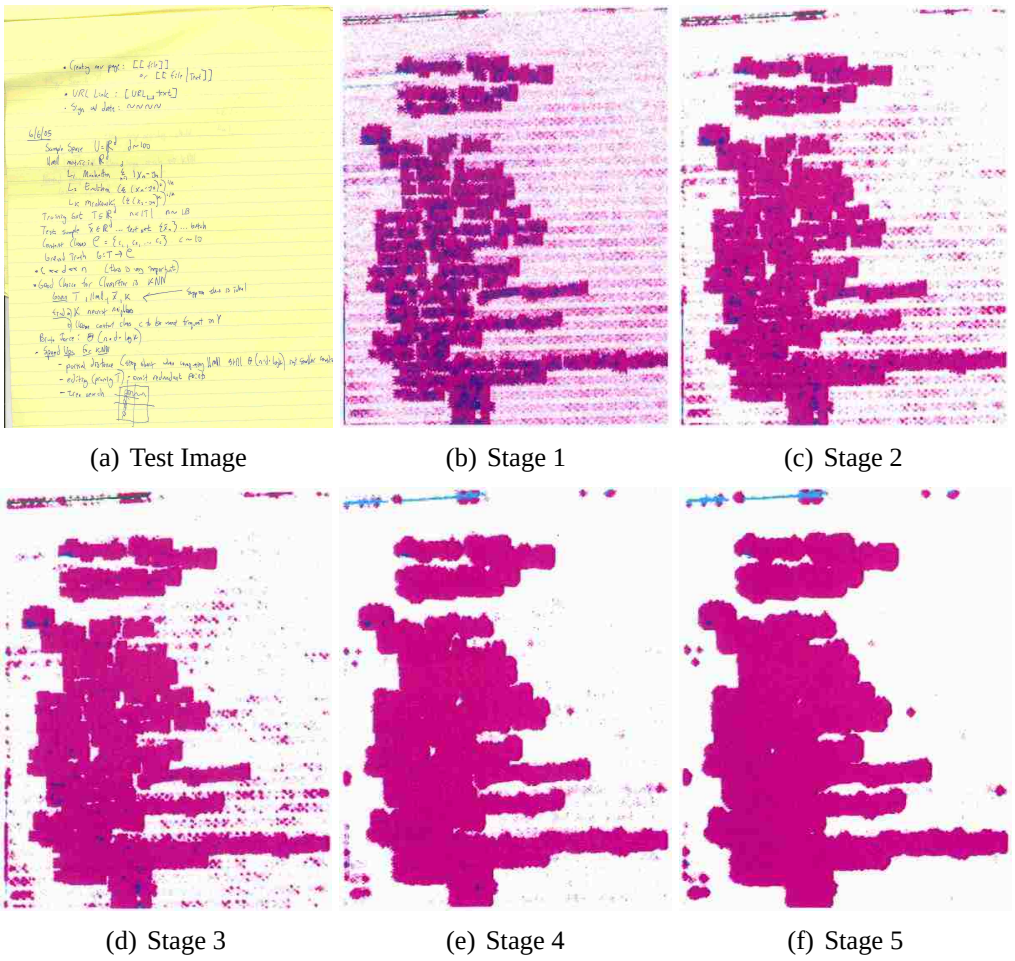
(b) Repeated Classification

Figure 3.5: Schematic Methodology of repeated classification. On the top is the post-classifier, which operates on the output of DICE classifier. On the bottom is the repeated classifier, which is trained on the results of the post-classifier. The output of the post-classifier is put back as its input, and reclassified repeatedly.

corrected as HW. Note that as MP pixels are fixed, BL pixels that are misclassified as HW are also fixed gradually. Repeated classification allows subsets of pixels that are dominated by one content (here, HW) class to expand.

However, repeated classification can also fail; especially it allows subsets of pixels that are dominated by incorrect content classes to expand. An example of incorrect expansion of handwriting is shown in Figure 3.7. In stage 1, we can see that most pixels lying within the text zone are correctly classified as MP. A few pixels within the text zone are misclassified as HW; this type of error happens to occur more severely on the boundaries of the text zone. The bottom and right margins of the image, which are blank, are erroneously dominated by HW. In stage 2, we can see that HW pixels within the text decrease. However, the HW clusters on the boundaries and margins expand. This continues in following stages. By stage 5, almost all HW pixels well inside the text are corrected as MP; but the HW clusters around the edge have expanded significantly. (Pixels in the square graphic located at the beginning of the text are misclassified MP almost all the time.)

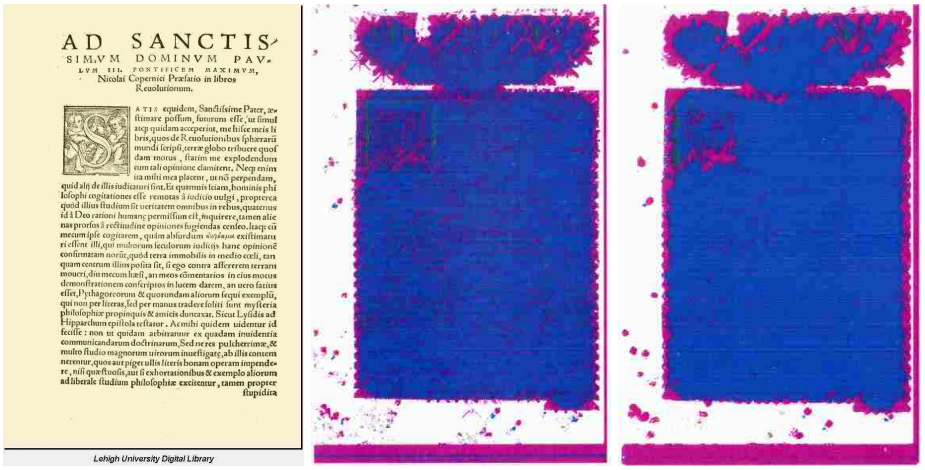
This expansion of erroneous clusters suggested to us that repeated classification is unstable in improving the results. It can reduce the errors in earlier stages, but might increase the error in later stages. After careful examining of the "false-color" images in the test set, and comparing them with the training set, we noticed that repeated classifiers may not improve at all stages. Repeated classifiers implicitly assume that the errors made by the DICE classifier occur, unchanged, again and again at every stage of post-classification. If this assumption were correct, repeatedly applying the same post-classifier would work fine at all stages. However, this assumption does not hold in reality. Post-classifiers can make different types



(a) Test Image (b) Stage 1 (c) Stage 2

(d) Stage 3 (e) Stage 4 (f) Stage 5

Figure 3.6: Illustration of repeated classification improving on handwriting. The test image contains only handwriting text. In stage 1, we can see that many pixels are misclassified as MP, especially these lying on the stokes. Most of their surrounding neighbors are classified as HW. In stage 2, we can see the clusters of MP shrink, that is, many pixels that are classified as MP in stage 1 are reclassified as HW. This correction continues in following stages. By stage 5, almost all MP pixels are corrected as HW.



(a) Test Image (b) Stage 1 (c) Stage 2

(d) Stage 3 (e) Stage 4 (f) Stage 5

Figure 3.7: Illustration of repeated classification expanding handwriting erroneously. The test image contains only machine-print text. In stage 1, we can see that most pixels lying within the text zone are correctly classified as MP. A few pixels within the text zone are misclassified as HW. This type of error occurs more severely on the boundaries of the text zone, and the bottom and right margins of the image, which are blank, are erroneously dominated by HW. In stage 2, we can see that HW pixels within the text decrease. However, the HW clusters on the boundaries and margins expand. This continues in following stages. In stage 5, almost all HW pixels within the text are corrected as MP; and the HW clusters expand significantly.

of errors at each stage. Thus repeatedly classifying the test image using the same post-classifier maybe unable to adjust to the changing situation. This thought motivates us to try re-training the classifier again at each stage, and leads to the design of iterated classification.

3.3 Iterated Classification

In previous section, we introduced repeated classification – we trained the second-stage classifier using the first-stage classification results, and continued using the same classifier for all following stages of classification. The flaw of repeated classification inspired us to try *iterated* classification: a sequence of post-classifiers, each trained separately on the training-data results of its preceding classifier, guided, as always, by ground truth. We will call the initial stage classifier (the DICE classifier) the *first stage* classifier, the immediately following post-classifier is the called the *second stage* classifier, followed by the *third stage* classifier, etc. A diagram of iterated classification is shown in Figure 3.8.

Our strategy has been to extract features from small local regions, *e.g.* circular windows of radius 9, so that no single classification stage affects a large area. It it worth emphasizing that we train each of the post-classifiers separately on the results from the training set of the previous stage. As we will show, this strategy appears to prevent the clusters of wrongly classified pixel to expand, while allowing those dominated by the correct class to expand slowly.

For the classification technology, we use approximate 5NN using hashed k-d trees [Cas06]. The features for the post-classifiers are discussed in Section 4.5.

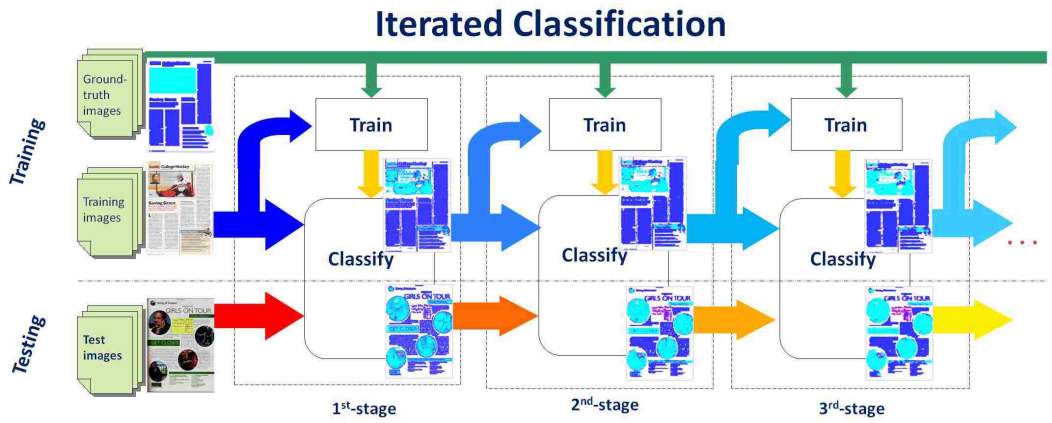


Figure 3.8: Schematic methodology of iterated classification. The same ground truth is passed to every training phase. Classification results are passed from one classifier to its successor for training and classification. Take the 2nd-stage classifier for example, the classifier is trained on ground truth and the first-stage classification results of the training images, and classifies both the training and test images. Note that each classifier is, in general, different from one another.

Chapter 4

Classification Algorithms

In this Chapter, we briefly introduce two automatically trainable classification technologies, developed by other students in our lab: brute-force k-Nearest Neighbors (kNN) and fast approximate kNN using hashed k-d trees. We describe in detail the features for DICE and iterated classification.

4.1 Brute-force k-Nearest Neighbors

We have implemented 5NN under the Infinity Norm using a brute-force algorithm. We regard this as our "gold standard" and compare other faster but usually less accurate methods to it.

4.2 Hashed k-D Tree Classifier

We have implemented our non-adaptive k-D tree classifier using fixed cuts, sped up by hashing bit-interleaved addresses [CB06] [Cas06], which runs up to several

hundred times faster than brute-force 5NN with only a small loss in accuracy in this domain. The experimental results described here were achieved using this classifier, hashing 24 bits of bit-interleaved address. We also sped it up by a technique of "inverted classification" ("filtering" in [Cas06]), in which test data are read first and hashed into the k-D tree; as the training data is read, data that hashes to an empty cell (*i.e.* one that contains no test data) can be discarded, while those that hash into occupied cells are of course used to "annotate" the relevant testing points with their class and distance (each testing point owning a list of up to k nearest neighbors so far). The principal advantage of this technique is that it allows us to constrain memory usage to $O(m)$, where m is the testing set size, with no sacrifice in accuracy and with the same computational cost (measured in numbers of distance computations). As test and training sets grow, inverted classification scales well since the test set can, with little or no loss in accuracy, be split into separate test sets as needed to maintain memory footprints small enough to avoid thrashing.

Since inverted classification allowed us to avoid thrashing, observed runtime was roughly proportional to the number of distance computations performed. For example, given a testing set of 3.3 million samples and a training set of 35,247 samples (this training set is small due to the decimation). a brute-force kNN classifier would perform over 110 billion computations, whereas the hashing classifier performed only 7.5 billion, a speed-up of a factor of 15.5. This allowed the classifier to run to completion in 47 CPU minutes, permitting frequent experiments which allowed a more thorough investigation of effective combinations of features. These results were typical of our experiments with the hashing inverted classifier.

4.3 Decimation of Training Data

Nearest Neighbor classification can be sped up simply by randomly throwing away most of the training pixels. Experiments carried out by Michael Moll and reported in [BMA07], showed that the loss of accuracy was significant by acceptable. to support that this sacrificed an acceptable accuracy: Figure 4.1 shows, on the left, a test page image and, on the right, five results of classification (using the brute-force 5NN classifier) with fewer and fewer training samples. With a decimation factor of 1000 (999 out of 1000 pixels omitted), per-pixel accuracy had fallen from 80% to 67% with a speed-up of a factor of over 350. As we scaled up our experiments, we increased the decimation factor to maintain an acceptable trade-off between accuracy and run-time. In experiments for iterated classification, when the size of the training set reached 33 images, we chose a decimation factor of 9000.

4.4 Feature Extraction for DICE

Each pixel (the "target pixel") is represented by scalar features extracted by image processing of a small region centered on that pixel.

We have investigated more than 60 features, all extracted from the luminosity channel (ignoring the hue and saturation channels): we selected twenty-six of these for the experiments reported here, for reasons summarized below. All feature values are scaled to lie within the (convenient but otherwise arbitrary) integer range 0-255.

- **Average Region Luminosity** (a group of four features): the average luminosity values of NxN-subregions centered on the target pixel (for N=1,3,9,27). The algorithm makes five successive passes. In the first pass, it simply copies

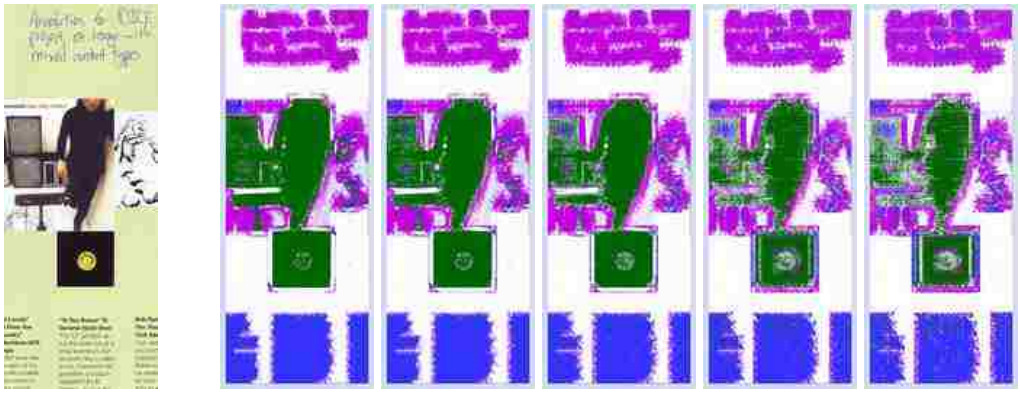


Figure 4.1: The results of classifying an image when decimating the training file used for classification. The first image on the left is the source image and from left to right after that are the results of using every training pixel (80.4% correct), every 10th pixel (72.9%, speedup of 7.9x), every 100th pixel (76.2%, 57.9x), every 500th pixel (70.0%, 212.5x), and every 1000th pixel (66.6%, 354.2x). Machine print and handwriting segmentation quality is generally good even though the number of unclassified (grey) pixels increases noticeably; but confusions between handwriting and photographs are sometimes glaring.

all the pixel luminosity values in an array. In the second through fourth passes, it calculates the sum of the luminosity values needed for the successively larger boxes by taking the sums of the values held in the smaller boxes (so 9 smaller boxes are added to create an larger one, at each pass). A final pass outputs the values to the feature array, dividing the values in each of the boxes by the number of pixels summed in that box.

On small, relatively specialized, training and test datasets, these features discriminated handwriting from machine print well, but their effectiveness lessened as the training set grew and diversified. Unsurprisingly perhaps, the larger the NxN region the less discriminating they were.

- **Region Luminosity Difference** (a group of sixteen features): each is the difference in total luminosity between halves of NxN regions cut in four directions: horizontal, vertical, and the two diagonals.

  These are effective in discriminating between BL (blank) and other content classes, with the (still somewhat mysterious) exception of HW (handwriting).

The following five groups of features extract features from straight lines of pixels centered on the target pixel, at each of the four directions. The length of these lines (in pixels) is an essential parameter of course: we'll give specifics of these choices at the end of this section.

- **Average Line Luminosity**: the average of luminosity values along the line.

  These assist in discriminating between handwriting and machine print. However, the diagonal features proved less effective than the horizontal and vertical features and were discarded.

4.4. FEATURE EXTRACTION FOR DICE

- **Line Luminosity Average Difference**: The average of absolute differences of luminosity between adjacent pairs of pixels along the line.

  The diagonal variants of these proved to be effective especially in combination with the average line luminosity features. But the horizontal and vertical variants were less effective and were discarded,

- **Line Luminosity Max Difference**: the maximum among absolute differences in luminosity between each pair of adjacent pixels along the line.

  These are effective especially in combination with "Average Line Luminosity" and "Line Luminosity Average Difference". They help discriminate BL (blank) from other classes.

- **Distance to Max-difference Pair**: the distance from the target pixel to the closest pair of pixels that possess a maximum luminosity difference.

- **Distance to Max-difference Pixel**: the distance from the target pixel to the closest one with a maximum absolute luminosity difference with the target pixel. Early experiments suggested that the two groups of features (immediately above) were not helpful, but they improved when revised as discussed next.

- **Revised Distance to Max-difference Pair**: these are features the same as above computed in eight directions radiating out from the target pixel, rather than in four directions centered on it.

- **Revised Distance to Max-difference Pixel**: same revision as discussed above.

These two revised features were not effective unless used together. Used together, they were the best features for discriminating between PH and other classes.

- **Difference Between Two Distances**: these are the differences between corresponding features "Revised Distance to Max-difference Pair" and "Revised Distance to Max-difference Pixel". They did not assist classification (and in fact increased the error rate). We tried other ways to combine these two features, including encoding the luminosity max-difference into the distance by multiplication, but there was no improvement.

4.4.1 Feature Combination

Having tested many (but, of course, not all possible) combinations and variations of the features described above, we gradually converged on the following twenty-six:

Region luminosity average: 1x1 (pixel) region;

Line luminosity average: horizontal and vertical, line-length 25 pixels;

Line average difference: line-length 25;

Line luminosity average difference: diagonals only; line-length 25;

Line luminosity max difference: four directions, line-length 41;

Revised distance to max-difference pair: eight directions, line-length 41; and

Revised distance to max-difference pixel: eight directions, line-length 41.

4.5 Feature Extraction for Iterated Classifiers

Each pixel (the "target pixel") sample is represented by scalar features extracted by image processing of a small region centered on that pixel[BMA07].

- **Pixel Class:** This feature is the content type value assigned by the earlier-stage classifier to the pixel. Presently there are four content types:HW, MP, PH and BL.

- **Disk Class:** This is a group of four features: each is the total number of pixels of a special content type within the circle of radius 5 centered on the target pixel.

- **Disk Edge Detection:** A group of thirty-two features: each is the total number of pixels of a content type within each half of circle of radius 5 cut in four directions: horizontal, vertical, and the two diagonals.

- **Disk Class Euclidean Distance Sum:** A group of four features: each is the sum of all distances from the target pixel to pixels of a content type within a circle of radius 6.

- **Pixel Content Type:** A group of four features: extension of Pixel Class feature. For example, if the preceding classifier label the pixel MP, then the MP feature is set to a non-zero value (here we use 186 based on our experiments result), otherwise it is set to zero.

- **Encoded Disk Edge Detection:** A group of sixteen features: extension of Box Edge Detection feature, each is the difference between two halves of the circle radius 5.

	3	5	7	9	11	13
2nd-stage	0.162	0.158	0.148	0.151	0.166	0.174
3rd-stage	0.144	0.141	0.138	0.137	0.143	0.160
4th-stage	0.136	0.135	0.133	0.134	0.141	0.150

Table 4.1: Error rates for training set of each stage using different scale of features, that is in radius of 3, 5, 7, 9, 11 and 13 pixels. Guided by the classification results for the training set, we chose radius of 7 for the second stage classification, 9 for the 3rd-stage, and 7 for the 4th-stage.

- **Neighbor Disk Class:** A group of sixteen features: extension of Box Class feature, each is extracted from the circular regions tangential to the center circle in the direction of horizontal and vertical.

4.6 Systematic Exploration of Scale of Features

In earlier experiments, we extracted features from circles of radius 5 pixels. Our experiment show that the classification results are sensitive to the radius. We have explored this sensitivity over a range of scales for each classifier stage separately. The experiments show that the best scale of features changes from stage to stage, as shown in Figure 4.1. Guided by the classification results for the training set, we chose radius of 7 for the second stage classification, 9 for the 3rd-stage, and 7 for the 4th-stage. The differences are not always statistically significant, but it is clear that the sweet spot is somewhere between 6 and 10 pixels radius for these features.

Chapter 5

Investigation of Ground-Truth Policies

Discussions of methods for obtaining ground truth to support pixel-accurate segmentations are reported in [AGB07, AKB06, PCHH93, SS03]. In [MBA08] we showed in large-scale experiments that the tighter (the more pixel-accurate) the truth, the better the resulting classifiers. Further discussion on pixel-accurate ground-truth policies are recently reported in [ACL10, Smi10, CAB10]. A PARC research team agreed [SLS09] with this observation, and built an efficient manual truthing tool, PixLabeler. We have applied this to our full-color and greyscale images, and compared it to earlier truthing policies we used.

5.1 What to Ground Truth?

The initial discussion of our ground-truth policy began with what classes we wanted to classify. In the context of our problem of document image content extraction, we

started with this initial list of content types: machine printed text (MP), handwriting (HW), photograph (PH), blank (BL), line art (LA), math (MT), engineering drawings (ED), chemical diagrams (CD), maps (MP) and junk (JK). We used this list to drive a systematic collection of document images for our database, containing each content type in bitonal, greyscale and color formats, in a variety of languages (when applicable). However, for initial testing of our classifier we tested on a smaller set of content types and we realized that some of these classes were possibly subsets of others. Therefore, initial ground truth only labeled MP, HW, PH and BL content.

As mentioned previously, manual ground truthing makes pixel-accurate ground truth infeasible, leading to a policy decision of what to classify. While we use overlapping rectangles, this also applies to any other scheme that uses polygons or any other shapes. Considering any form of text, handwritten or machine printed, the next level up from pixel accurate ground truth would be at the character level, then the line level and finally the paragraph level. Since our classifier is labeling each pixel based on a small window around each pixel, combined again with the infeasibility of manual labor, we chose not to pursue character level ground truth. Character level zoning also presents a challenge in determining where a character begins and ends, as discussed in [KC94]. Some of the white pixels in between and around the black pixels of a character must also be considered part of a character and sometimes these regions may overlap. We chose to ground truth at the paragraph level initially as this was the most efficient policy time wise and as we were improving the classifier this yielded acceptable results. We will discuss alternatives to this policy decision in later sections.

5.2 Blank Space

As mentioned before, we chose to treat blank space as a unique class and therefore we must also ground truth blank space like any other content class. An initial idea was to label any pixel not explicitly "zoned" by the user in our tool as blank, however there were multiple reasons for not making this policy. Some documents may have types of content that we are currently not testing and we would like to intentionally not ground truth or there may be ambiguous areas of the document that contain multiple content types or that the user is unsure of how to label. For the purpose of training data, these pixels can be left unlabeled and will be ignored in training the classifier. Finally, since we treat blank space as an equal class to the other classes we should use the same policy for obtaining ground truthed data as we do for the other classes.

Our ground-truth policy however, created some problems for our classifier in classifying blank space. At any level other than pixel accurate ground truth, some amount of blank space will be included in the areas zoned as other content types (*i.e.* the white space between lines of text, the white space inside the letter o, etc). If ground truth is particularly sloppy or loose, this can introduce what appears to be noise in areas classified as blank space. Experiments with our classifier show that this problem occurs most frequently with confusing blank space for handwriting and in more limited cases also for machine print. This is due to the more free form layout of handwriting samples, compared to the more uniform layout of machine print. Experiments discussed later confirm the idea that more careful, tighter ground truth of handwriting samples, lead to less mistakes in classifying blank space.

5.3 Overlapping Content

One problem that we have dealt with from the beginning of this dissertation and have yet to find a satisfying policy for is that of how to zone areas that contain overlapping content areas. Part of our research goal is for our classifier to do well on images with difficult, complex layouts. This includes images that have complicated backgrounds, possibly photographs, with machine print over them. Other common forms of this problem are machine printed documents with handwriting annotations.

Our policy has been to try to as tightly as possible zone the foreground pixels (the MP over the PH, the HW over the MP) before labeling the background pixels. However, since we are not adopting a pixel-accurate ground-truth policy this has the potential of introducing some "noise"' pixels to the ground truth for that class. Current experiments have not shown any serious problems with this policy for the classifier, however more experiments should be conducted using training sets consisting of much larger amounts of overlapping data. An alternative policy would be to assign two class labels to overlapping areas. No experiments have been completed with this policy yet.

5.4 Machine Print in Photographs

A special form of the above problem is specifically how to handle machine print and photographs when they overlap. The above mentioned example of a magazine article with a photograph as background with a story printed over it or a caption on a photograph seems straight forward. We try to tightly zone the MP and then zone the PH around it. However, a unique case is that of a photograph that contains

machine print. For example, an image taken from a handheld camera of a street sign or even a newspaper article with a photograph of a football player showing his name on his jersey. While the case of the street sign in the image obtained from a digital camera seems straight forward, to label the text as machine print it quickly becomes less clear if the street sign is not the focus of the photograph or the case of the newspaper article with a photograph. In this case we consistently do not label the text as machine print.

5.5 Difficult Shapes

We chose to use overlapping rectangles for our zoning to make implementation of our zoning tool simpler, as well as simplifying the zoning process for the user. Many of the documents we collected to train and test on contain difficult, non-rectangular layouts. Even with a tool for zoning that uses polygons instead of simple rectangles would have an imperfect representation of the actual layout in the ground truth. The policy we use for these areas are trying to capture as much of the detail and as little noise as possible using many small rectangles. This is unfortunately a very time consuming process for the person doing the zoning, and at best is still imprecise. An alternative in our research program is to leave images like this out of the training set, as our classifier does not learn from the layout of a page, but from the content of a page. However, this is obviously not an acceptable policy for all research. This also creates an evaluation problem that will be discussed later, as it will force pages with these difficult layouts to be scored worse than they should be using some evaluation metrics.

5.6 The Effect of Tighter Zoning

Given the problems encountered with using a non-pixel-accurate ground-truth policy for a pixel level classifier, we began to experiment with using a tighter ground-truth policy to try and reduce errors to improve overall classification. As discussed and illustrated before, our initial ground-truth policy was designed to drive development of the classifier and running experiments with very large numbers of training and test images. This required a ground-truth policy that was not extremely labor intensive and relatively simple for new people in our lab to adopt. However, as performance of the classifier became more stable and test set sizes started growing less slowly, we realized one area of our program that could potentially lead to great increases in performance was our ground-truth policy.

5.7 Ground-truth Policies

In this section, we discuss the differences among three ground-truth policies.

We have developed a web-based user interface to zone document images in PNG format, using overlapped rectangles. Using this, we can capture loose and tight ground truth. Loose ground truth is obtained by sweeping rectangles to enclose entire block of a particular content type. This policy inevitably encloses blank space that is inter-column, inter-paragraph, at the end of a paragraph. Therefore, loose ground-truthing is an efficient manual task. Tight ground truth requires more care to enclose individual textlines, sometime even hand-written strokes or large letters. As a result, tens of times more rectangles, need to be swept. Pixel-accurate truth, in which only foreground pixels are labeled, is obtained by applying the PARC

PixLabeler [SLS09] tool; in our experience this tool was faster and easier than loose truthing methods.

5.8 Morphological Expansions of Pixel-accurate Truth

To better understand the effectiveness of pixel-accurate ground truth, we generated morphological expansions on it. We expanded foreground pixels by applying morphological dilation operations [Dou92a] with a circular disk structuring element, that is, background pixels within a distance d of a foreground pixel are labeled as the same class as the foreground pixel. This choice of circular disk structuring element is justified by the notion that a circle does not imply bias to any particular directions. We have generated four morphological expansions on pixel-accurate ground truth, using Matlab®, labeled by radii, in pixels, of the disks: d=1, d= 2; d= 4, and d=8. The choice of any specific radii does not matter much here, as long as there are several of them and obtained expansions are somewhere between pixel-accurate and tight truth.

5.9 Experiment Design

We have compared the effectiveness of loose and tight ground truth in iterated classification [AB08], and found that tight truth reduces per-pixel classification errors by 45% (from 38.9% to 21.4%). Now we add experiments on pixel-accurate ground truth and its morphological expansions. We use a training set of 33 images and a distinct test set of 83 images, which are the same images we used in [AB08]. Together the two sets contain machine-print (MP), handwriting (HW), photograph

(PH) and blank ("don't care") (BL). The training data was decimated randomly by selecting one out of every 9000th training sample.

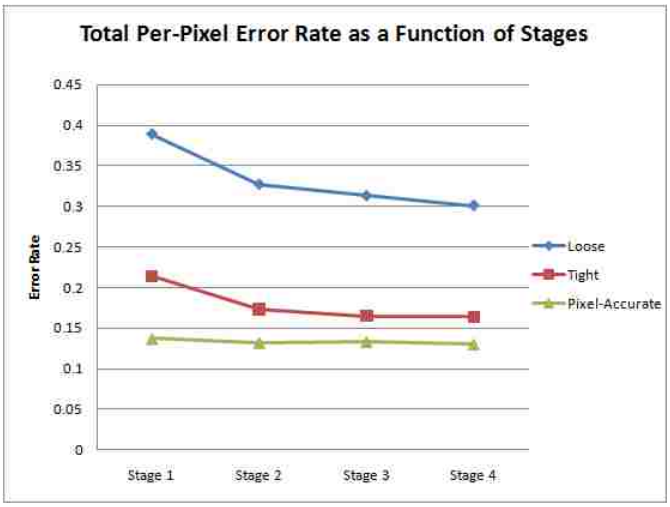
We evaluated performance using per-pixel accuracy, precision and recall. Per-pixel accuracy is the fraction of all pixels in the document image that are correctly classified. Unclassified pixels are counted as incorrect. Precision is defined as the number of pixels correctly classified as belonging to a positive class divided by the total number of pixels classified as belonging to the positive class. Recall is defined as the number of pixels correctly classified as belonging to a positive class divided by the total number of pixels that actually belong to the positive class.

For each truthing policy, we trained a classifier using that kind of ground truth, then tested, and evaluated performance on the same type of ground truth.

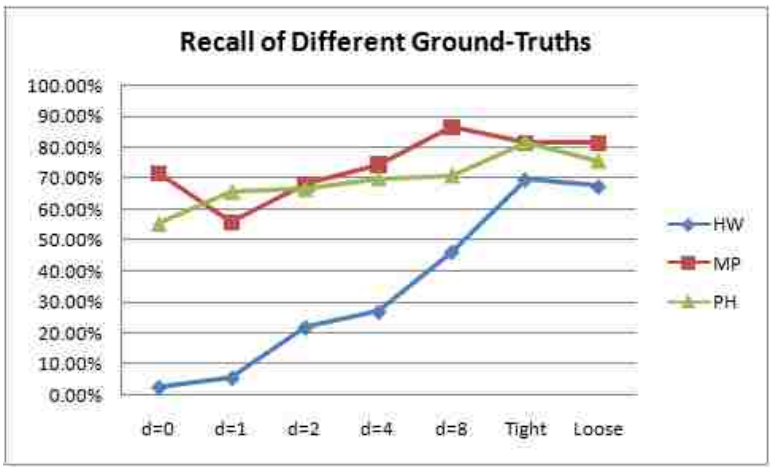
5.10 Experimental Results

In this section, experiments comparing different ground-truth policies are presented.

The results of loose, tight and pixel-accurate truth are shown in the form of per-pixel error rate in Figure 5.1(a), which indicates that in each of the classification stages, total error rate (averaging over all classes) decreases as the truth goes from "loose" to "tight", and finally to pixel-accurate (via PixLabeler). Notice that the drop of error rate from tight to pixel-accurate is less significant than that from loose to tight. The figure also show that the error rate decreases monotonically as a function of stages. For the fourth stage, the difference between error rates of tight and pixel-accurate is moderate. It seems like a clear win for pixel-accurate ground truth.

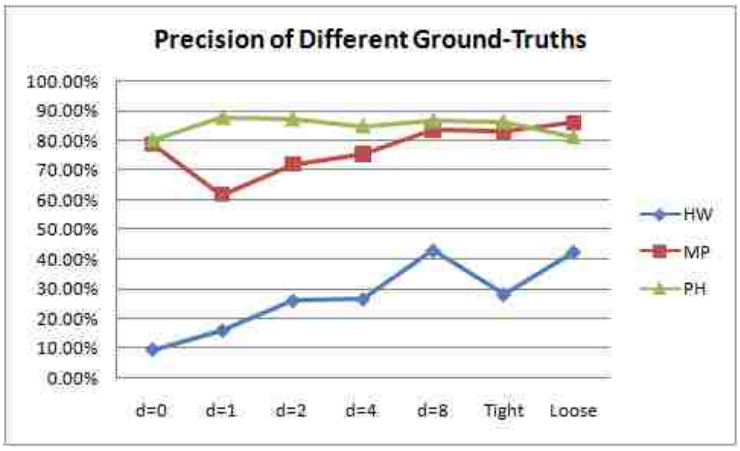


(a) Error rate

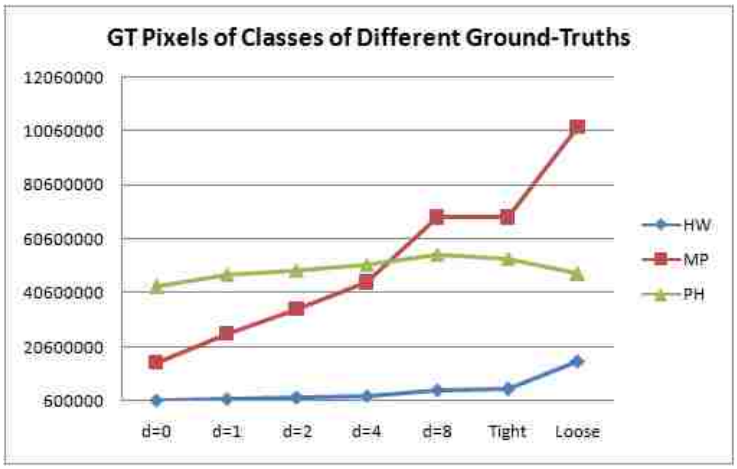


(b) Recall

Figure 5.1: On the left, (a) shows the total per-pixel error rate of loose, tight and pixel-accurate ground truth as a function of stages of iterated classification. In each stage, the error rate decreases as the truth goes from loose to tight, and finally to pixel-accurate. For each ground-truth policy, the error rate also decreases monotonically as a function of stages. On the right, (b) shows the recall for each content class as a function of truthing policy: pixel-accurate (d=0); morphological expansions by d=1,2,4,8; tight; loose.



(a) Precision



(b) GT Pixels

Figure 5.2: On the left, (a) shows the precision for each content class as a function of truthing policy, including morphological expansions on pixel-accurate ground truth. Pixel-accurate ground truth is represented by "d=0". On the right, (b) shows the number of pixels in each content class by truthing policy.

However, a closer look at each content class reveals problems. For one thing, the overall error rate drops only slightly (from 16% to 14%) from tight to pixel-accurate. More seriously, some classes suffer a catastrophic fall in recall in the final move to pixel-accurate ground truth: for examples, handwriting recall drops from 70% to 2%.

The results of precision and recall for morphological expansions on pixel-accurate truth, along with loose, tight and pixel-accurate, are shown in Figure 5.1(b), Figure 5.2(a) and Figure 5.2(b). In all the three figures, pixel-accurate truth is labeled as "d=0".

In Figure 5.1(b), the recall on handwriting (HW) increases as the pixel-accurate truth expanded, and reaches the highest with tight truth. The recall on machine-print (MP) and photograph (PH) also increases as pixel-accurate truth is expanded (*i.e.* d=1,2,4,8). Figure 5.1(b) also suggests that one of the pixel-accurate truth's morphological expansions, labeled as "d=8", yields results similar to tight truth.

In Figure 5.2(a), we can see that some morphological expansions yield higher precision on machine-print (MP) and photograph (PH) than pixel-accurate ground truth. Note that precision on handwriting (HW) increases monotonically from pixel-accurate truth to the morphological expansion of "d=8". Figure (a) also suggests that the morphological expansion of "d=8" is nearly equivalent to tight truth on MP and PH. In Figure 5.2(b), notice that the number of MP pixels increases the most in both quantity and percentage, and HW the least.

Consider this scenario: given a particular fixed test policy, and several competing training policies, it is conceivable that one of these training policies would score

the best. It would be interesting to see which training policy wins such a competition. Since we have several ground truth policies available, as well as the results, we computed the scores and the results are shown in Table 5.1. Note that it makes sense to read the table row by row, that is, given a test policy, how well and different the classifier performs when trained with different training policies. When read column by column, the table presents the results with different meaning, that is, given the output of particular classifier, how it scores when evaluated by different ground truth. This does not make much sense though. This table indicates that for each test policy, the highest accuracy is achieved by using the same policy for training. It also suggests that the more similar the training policy is to the test policy, the higher the accuracy.

		← Training Policies →						
		Pixel-Accurate	**W=1**	**W=2**	**W=4**	**W=8**	**Tight**	**Loose**
↑ Test Policies ↓	**Pixel-Accurate**	**0.864**	0.833	0.804	0.759	0.645	0.637	0.440
	W=1	0.823	**0.832**	0.826	0.795	0.695	0.686	0.495
	W=2	0.782	0.820	**0.831**	0.815	0.730	0.719	0.534
	W=4	0.734	0.790	0.817	**0.825**	0.767	0.754	0.578
	W=8	0.616	0.680	0.723	0.770	**0.836**	0.817	0.681
	Tight	0.623	0.677	0.720	0.761	0.822	**0.836**	0.685
	Loose	0.349	0.406	0.453	0.505	0.630	0.639	**0.699**

Table 5.1: Row entries represent test policies, and column entries represent training polices. For example, the numbers shown in the first row are accuracies of classifiers trained on different GT policies, and evaluated with pixel-accurate GT. In each row, the highest accuracy is shown in bold. This table indicates that for each test policy, the highest accuracy is achieved by using the same policy for training. It also suggests that the each test policy, the more similar of the training policy to the test, the higher the accuracy.

So far we compared the error rates, precision and recall of different ground-truth policies; and all results are from the fourth stage. We are also interested to examine the effects of different ground-truth policies have on iterated classification. The results, in the form of error rate as a function of stages of iterated classification,

	1st-stage	2nd-stage	3rd-stage	4th-stage
Pixel-Accurate	0.137	0.132	0.133	0.130
d = 1	0.171	0.154	0.168	0.168
d = 2	0.188	0.164	0.164	0.169
d = 4	0.211	0.173	0.178	0.175
d = 8	0.220	0.168	0.161	0.164
Tight	0.214	0.173	0.165	0.164
Loose	0.389	0.327	0.313	0.301

Table 5.2: Error rate as a function of iterated classification stages, and using different ground-truth policies. Generally, iterated classification always reduce the error rate for all ground-truth policies. For ground truth of "d=1", the error rate decreases by 1.7% from the 1st stage to the 2nd stage, but then bounces back by 1.4%. For ground truth of "d=8", the error rate decreases from 22.0% to 16.4%, a drop of 25.5%, which is the most significant drop of all ground-truth policies.

are shown in Table 5.2 and Figure 5.3. For all ground-truth policies, the error rate is reduced from the first stage to the fourth stage, although always monotonically. This suggests that *iterated classifiers yield no worse results than the DICE classifier no matter what the ground-truth policy is.* Notice that the drop of error rate is slight for pixel-accurate ground truth and its expansion of "d=1"; the drop becomes more significant as the ground truth expands, the drop reaches the most significant when the ground truth is loose.

5.11 Problems with Pixel-Accurate Ground Truth

Several things might have caused the problem we saw with pixel-accurate ground truth: (1) imbalance in the training set (handwriting pixels were significantly fewer than others); (2) confusion between foreground pixels and background pixels that

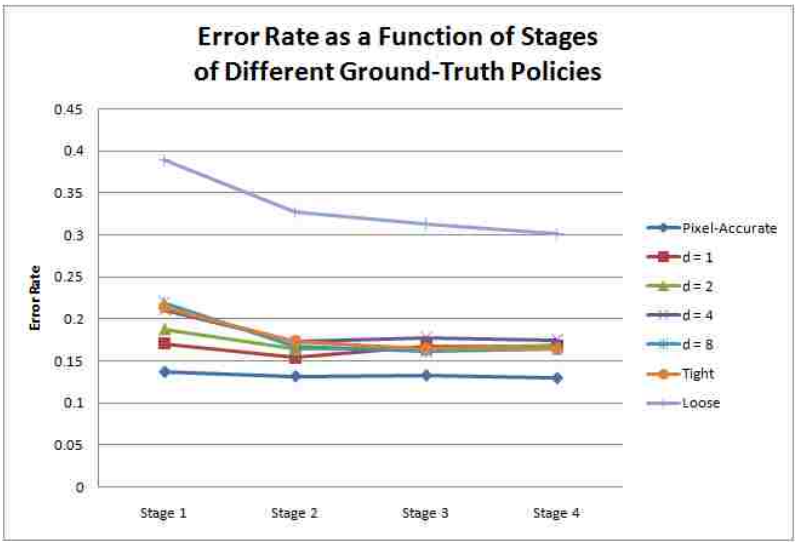


Figure 5.3: Error rate as a function of stages of different ground-truth policies.

are inter-character, inter-word and inter-line, etc; and (3) bad fit with the features (*e.g.* the radius of the feature extraction window is badly chosen).

Figure 5.1(b), Figure 5.2(a) and Figure 5.2(b) show that a moderate increase (less than eight times) of ground-truthed HW pixels leads to a significant improvement (more than 23 times) of recall (from 2% to 46%).

Random decimation, as discussed in Section 4.3 does not take into consideration imbalances in the training set. In separate collaboration with Dawei Yin [YAB10], we have explored another speed-up method, impurity-decimation that automatically rebalances training data and estimates concentration in each K-d hash bin separately, which then controls how many samples should be kept in each bin of the hashed k-D tree. We do not include that research in this dissertation.

Our algorithms extract features from a circular window centered on target pixels. It is possible that the background pixels (inter-character and inter-word) have

similar feature values as foreground pixels, *e.g.* handwriting. Such background pixels would be hashed with foreground pixels into the same bin, thereby causing classification mistakes.

We can not avoid choosing a test policy; but it is not clear which one is the "best" for the purpose of evaluation and diagnosis. See earlier section 2.8 for detailed discussion of competing ground-truth policies. The accuracy table 5.1 can not, of course help us choose the "best" test policy.

From the perspective of training, the accuracy table indicates how different ground-truth policies can affect classification results. Most noticeably, it suggests a way to choose among different training policies when a test policy is given, that is, train the classifier with the same policy.

Figure 5.3 might suggest that the closer the ground truth is to pixel-accurate, the less iterated classification improves results. However, this is not as straightforward as it seems. Recall that the feature extraction window we used in this experiment has a radius of 7 pixels; and the most three significant drops in error rate occur on the loose, the tight, and the expansion of "d=8". It seems likely the iterated classifiers perform better when the radius of the feature extraction window is smaller than the expansion parameter of pixel-accurate ground truth. This does not mean we should choose ground truth for testing to fit the feature extraction window of the classifier.

5.12 Conclusion

We have compared performance among three ground-truth policies, loose, tight and pixel-accurate, along with morphological expansions on pixel-accurate truth. Experiments suggest that pixel-accurate ground truth can improve overall accuracy. One consideration to take into account when comparing loose, tight and pixel-accurate ground truth is the ground truthing effort (*e.g.* in hours of manual or semi-automatic labor). Roughly, using our zoning tool, we averaged about 3 minutes per image for loose ground truthing, 10 minutes per image for tight ground truthing; and using PARC's PixLabeler, we averaged about 2 minutes per image for pixel-accurate ground truthing. With automatic morphological expansion, pixel-accurate ground truth can cheaply provide variations that are similar to loose truth or tight truth. Choice of ground-truth policies do not significantly affect runtime, either for training or for testing. Therefore, we prefer PARC's PixLabeler to our zoner. For the choice of ground-truth policy for training using our classifier, based on the comparison and experimental results, we recommend morphological expansion of "d=8".

Chapter 6

Experiments

This Chapter presents the results of a series experiments, and generally a larger and larger training and test set.

Our data sets contain MP, HW, PH, and BL content. Their text includes English, Arabic, Chinese, Hindi, and Korean characters each represented by bilevel, greylevel, and color document image examples. The selection of test and training pages was random except that for each test image there was at least one similar (from the same source), but not identical, training image. Thus these experiments test the discriminating power of the features and weak generalization (to similar data) of the classifiers, but they do not test strong generalization to substantially different cases.

We evaluated performance in two ways:

Per-pixel accuracy: the fraction of all pixels in the document image that are correctly classified: that is, whose class label matches the class specified by the

ground truth labels of the zones. Unclassified pixels are counted as incorrect. This is an objective and quantitative measure, but it is somewhat arbitrary due to the variety of ways that content can be zoned. Some content—notably handwriting—often cannot be described by rectangular zones. This in some cases will lead to a per-pixel accuracy score being worse than an image may subjectively appear to be. The overall per-pixel accuracy, O, is calculated as shown in Equation 6.1, where $N_{i,j}$ denotes the number of pixels that are ground truthed as class i and classified as j ($i, j \epsilon C, C = \{MP, HW, PH, BL\}$), $N_{i,i}$ is the number of pixels that are correctly classified, that is, classified as its ground truth class.

$$O = \frac{\sum_{i \epsilon C} N_{i,i}}{\sum_{i \epsilon C} \sum_{j \epsilon C} N_{i,j}} \tag{6.1}$$

Precision and recall: Precision P_i of class i is calculated as shown in Equation 6.2, where $N_{i,i}$ is the number of pixels that are correctly classified i, and $\sum_{j \epsilon C} N_{j,i}$ is the number of pixels that are classified as belonging to class i.

$$P_i = \frac{N_{i,i}}{\sum_{j \epsilon C} N_{j,i}} \tag{6.2}$$

Recall R_i of class i, is calculated as shown in Equation 6.3, where $N_{i,i}$ is the number of pixels that are correctly classified as i, and $\sum_{j \epsilon C} N_{i,j}$ is the number of pixels that actually belong to class i.

$$R_i = \frac{N_{i,i}}{\sum_{j \epsilon C} N_{i,j}} \tag{6.3}$$

6.1 Instability and Workarounds

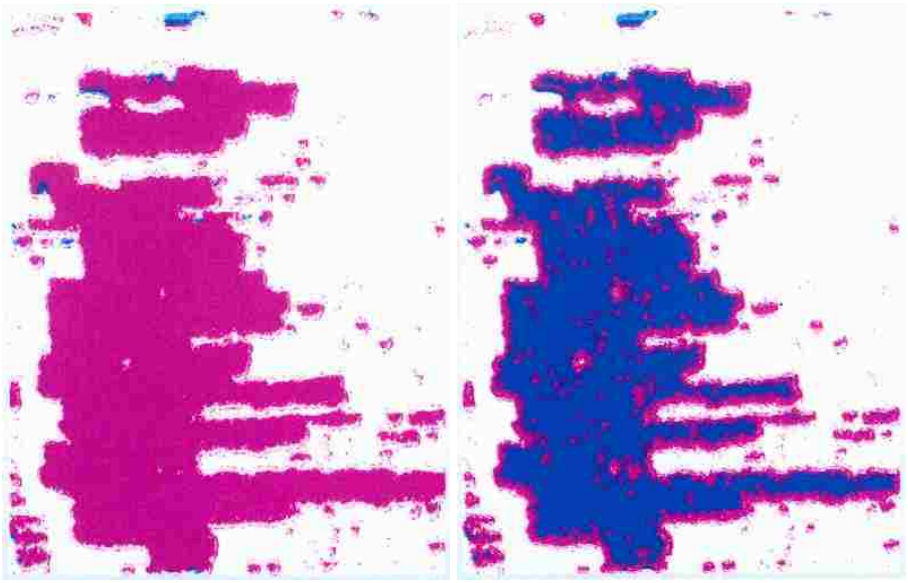
In this Section, we discuss an instability that occurred in an early small experiment.

In this experiment, we tested with 17 images: 8 images were placed in the training set, and 9 in the test set. We ran ten iterated classification stages. For the first eight stages, the total error rate decreased almost monotonically from 34.4% to 24.4%, except for a slight bounce of 0.2% at the seventh stage (from 24.8% to 25.0%). Then at the ninth stage, errors increased by 26.7% over the eighth stage (from 24.4% to 30.9%). Large solid clusters of hand-writing were suddenly misclassified as machine-print, an example is shown in Figure 6.1.

After locating the misclassification on test images, we examined the iterated classification results for training images in order to find if similar mistake took place among them. We found that the error rate of training set also increased significantly at the ninth stage (from 19.8% to 38.5%), cause by the misclassification of MP over HW. The error rates on each stage for both test and training sets are shown in Figure 6.2.

Could we be notified of this kind of error in real application where no ground truth for test set is unavailable? Given the representative property between test set and training set, that is, each test image is represented in training set. we are assured that we can by tracking the error rate of training set.

The cause appears to be as follows as we take a close check up with training images. As iterated classification proceeds, isolated pixels are relabeled, and clusters become increasingly uniform. This effect aggregate as stage furthers. In one training image, a thin “gutter” cluster separating MP blocks, which was in fact BL, but was, for convenience, manually ground-truthed MP. This cluster was classified



(a) mnotes in stage 8 (b) mnotes in stage 9

Figure 6.1: Classification "false color" images of a handwritten notes, which contains only handwriting and blank. On the left is the result of the eighth stage, most of the HW pixels are correctly classified. On the right is the result of the ninth stage, HW pixels with in large solid clusters are misclassified as MP, except for those lying on boundaries or within small clusters.

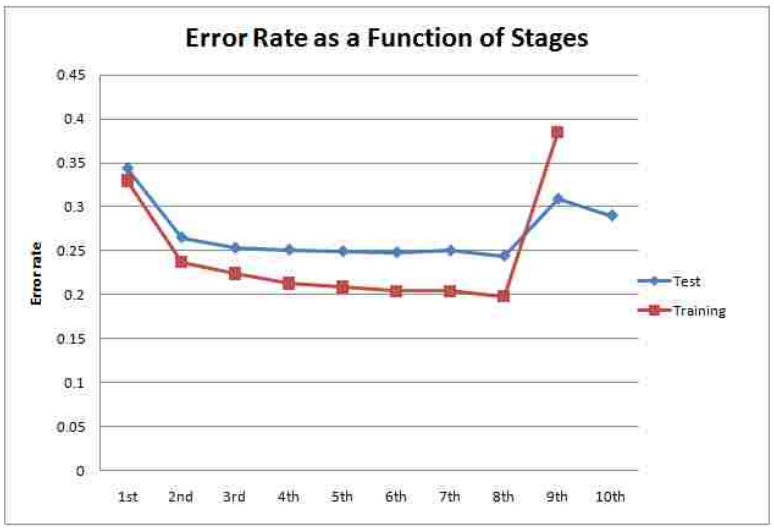
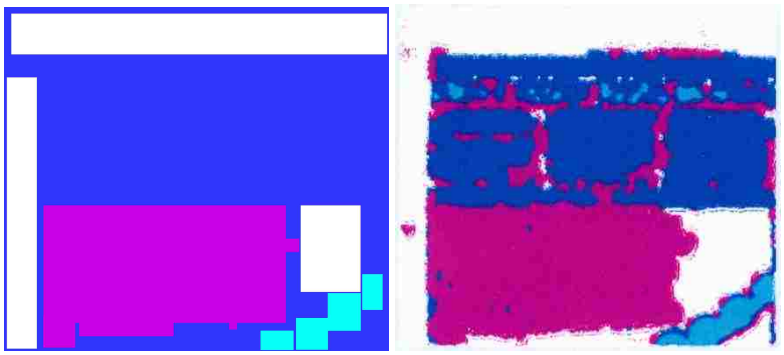


Figure 6.2: Error rates as a function of stages for both test and training sets. Note that for the first eight stages, the error rate of test set decreases as the error of training set does. The test set's curve is similar to the training set's. At the ninth stage, the errors of both sets increase significantly. The error rate of training set increases twice as much as test set does.



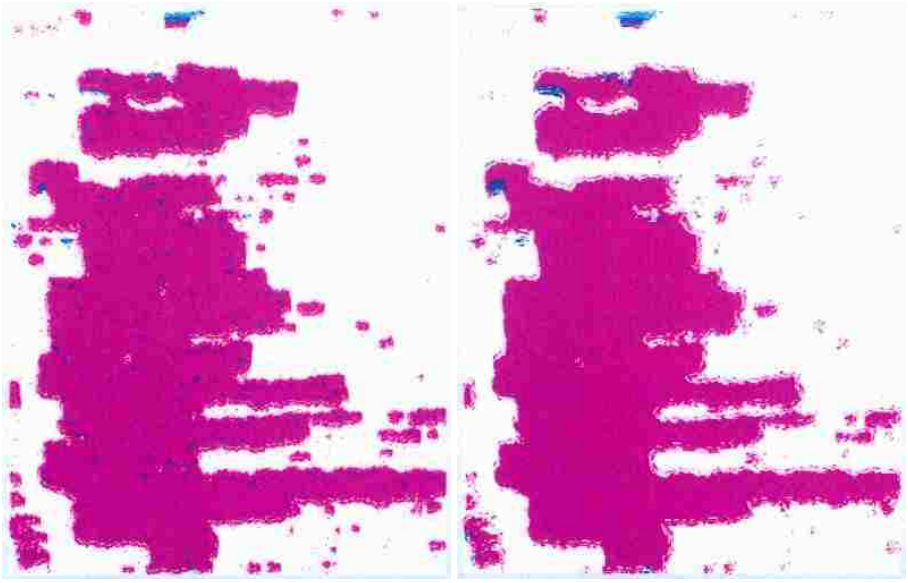
(a) GT of bw10 (b) bw10 in stage 8

Figure 6.3: Ground truth and classification "false color" images of a newspaper crop in the training set. On the left is GT image, note that the thin "gutter" blank space between the title and the paragraphs are ground-truthed MP. On the right is the result of the 8th stage, the thin "gutter" blank space are misclassified as HW.

HW by the eighth classifier, as shown in Figure 6.3. Thus the incorrectly classified samples whitin the gutter fall at exactly the same point in feature space as correctly classified MP. This led the NN classifier to mistake large clusters of MP for HW. The essential problem is that incorrectly classified clusters, even small in area, once purified to a certain threshold, can compete with large clusters that are consists of correctly classified pixels.

We have found two engineering workarounds to reduce the incidence of this instability. The first is to drop a training image out of the training set whenever its classification error rate rises. The second is to increase the radius of the features. The result of these to workarounds are shown in Figure 6.4.

We do not have a full enough understanding of this problem to propose guaranteed solutions. This problem is due to aggregation of several factors: the loose



(a) Effect of Dropping bw10 (b) Effect of Increasing Radius

Figure 6.4: Illustration of two workarounds. In both figures, the misclassification occurred in stage 9 of earlier experiment disappears. The figures indicates that increasing radius of feature window outperforms the act of dropping the training image of bw10.

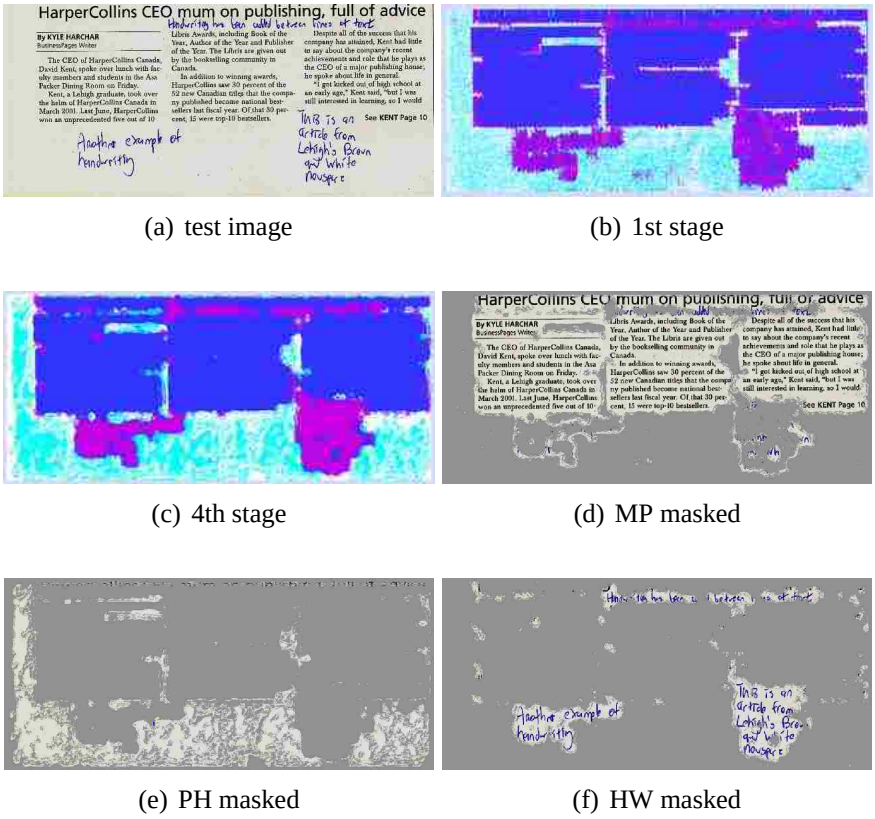
ground truth, classification error, and iterated classification. A small patch of erroneous ground truth plus a nuance error, which continues increasing as iterated classification marches, can flip the dominating correct result. However, this instability issue suggests that defects in ground truth can lead to failure to the classifier.

6.2 Preliminary Experiment with Loose Ground Truth

In this experiment, we selected a training set of 33 images and a distinct test set of 83 images. Each content type was zoned manually (using closely cropped isothetic rectangles, overlapped where needed to fit non-rectangular regions) and the zones were ground-truthed. The training data was decimated randomly by selecting only one out of every 9000th training sample.

Our results are illustrated in Figures 6.5 and 6.6. Each figure contains six images of three types: (a) the original image; two classification images from stages one (b) and four (c); and three *mask* images for MP (d), PH (e), and HW (f) content classes. In the mask images—say, for example, the MP (machine-print) mask image, only the regions that are classified as machine-print are extracted and displayed using their original color pixel values; the pixels of other classes are shown as light grey.

Figure 6.5 shows results on a color image of a newspaper page containing non-rectilinear handwriting regions. The first stage classifier locates handwriting fairly precisely, but mixes it with many machine-print misclassifications. We could read most of the handwriting extracted by the handwriting mask image. The light blue texture in the background is uniform from the start and does not worsen under post-classification.



(a) test image (b) 1st stage

(c) 4th stage (d) MP masked

(e) PH masked (f) HW masked

Figure 6.5: A color image containing rectilinear machine-print regions and non-rectilinear hand-writing annotations. The error of the 1st-stage classifier is 37%; the error of the 2nd-stage classifier is 36.4%; and the error of the 4th-stage classifier is 34.2%. The MP mask extracts almost all of the MP except for a little near the (unclassifiable) page boundary. Almost all of the HW is extracted correctly, except for patches where MP crowds it.

Figure 6.6 shows results on a color image of a magazine page with a block of handwriting on a yellow ruled background. The iterated post-classifiers cleans much of the sparse light blue texture in the background, without causing the thicker light blue texture to expand, in fact some of it shrinks, which is good. Note that it cleans most of the red texture, both sparse and thick ones, in both the machine print and photo regions. Meanwhile, the curvelinear boundaries of those large regions are accurately detected, as well as the blank regions between paragraphs. The post-classifiers also eliminate most of the erroneous handwriting areas in the yellow ruled background while enhancing the handwriting regions by removing the machine-print texture within them. The mask images are highly promising in representing handwriting, machine-print and photo layers.

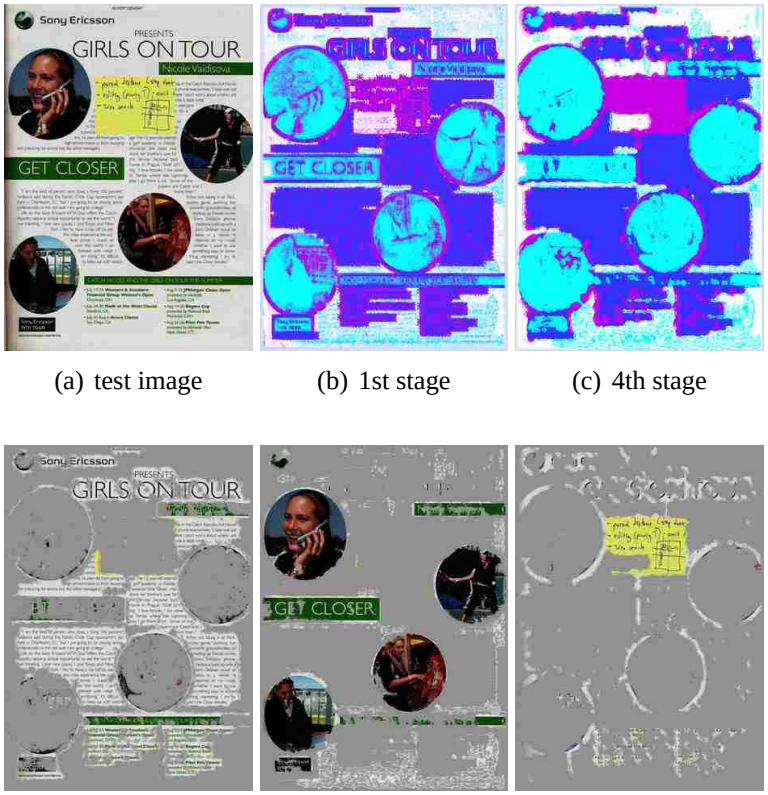
Figure 6.7 gives the total error rate as a function of stages of classification. Iterated classifiers reduce the error rate by 22.6%.

6.3 Preliminary Experiment with Tight Ground Truth

In this experiment, each content type was zoned manually (using closely cropped isothetic rectangles) with careful and the zones were ground-truthed. The training data was decimated randomly by selecting only one out of every 9000th training sample.

We have experimented with 116 page images: 33 images were placed in the training set, and the rest in the test set.

Experiments show great improvement on tighter ground truth. With loose ground truth, for both training and testing, the error rate for the first stage of classification



(a) test image (b) 1st stage (c) 4th stage

(d) MP masked (e) PH masked (f) HW masked

Figure 6.6: A magazine image with a complex non-rectilinear page layout. The per-pixel classification error of the 1st-stage classifier is 36.7%; and the error of the 4th-stage classifier is 27.4%. The final MP, PH, and HW masks extract their content types well, as shown in (d)-(f), with the exception a few small patches of HW misclassified as MP.

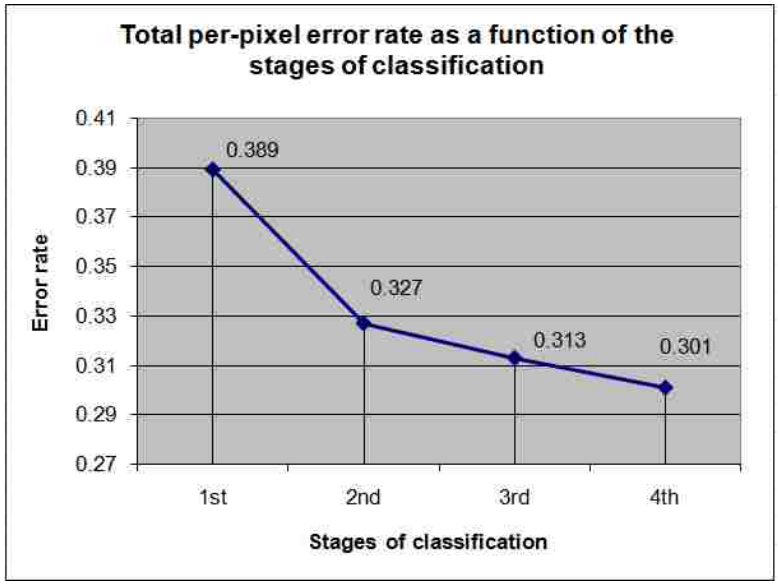


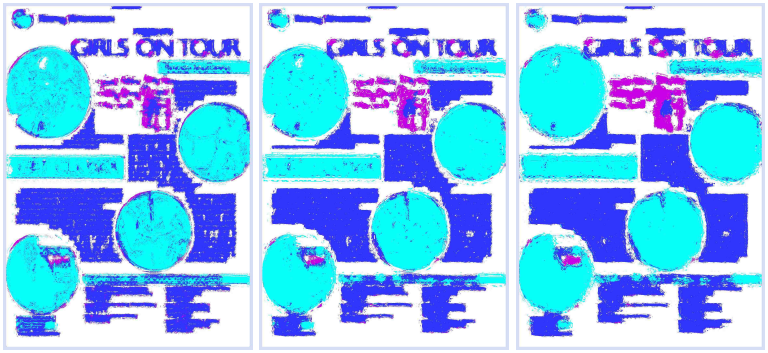
Figure 6.7: Total error rate averaged over the test set, as a function of stages of classification. After four stages of classification, the error rate decreases from 0.39 to 0.30, a drop of 23%.

was 38.9%; With tight ground truth, for both training and testing, the error rate for the first stage of classification has decreased to 21.4%, a drop of 45%.

Our results are illustrated in Figure 6.8 and Figure 6.9. Each figure contains nine images of four types: (a) the original image; classification images from stage one using loose ground truth (b), classification images using tight ground truth from stages one (c), two (d), three (e), and four (f); and three *mask* images for MP(g), PH(h), and HW(i) content classes. In each of these two figures, the original images are shown on the upper left. The results of classification are shown in (b)-(f), as classification images where the content classes are shown in color: machine print (MP) in dark blue, handwriting (HW) in red, photographs (PH) in light bluegreen, and blank (BL) in white.

Figure 6.8 shows results on a color image of a sports magazine page containing complex non-rectilinear regions. With tight ground truth, the per-pixel error of the first-stage classifier is 22.9%; Figure 6.8(b) shows the result obtained with loose ground truth: the per-pixel classification error of the first-stage classifier is 36.7%. Note that BL regions are mixed with PH pixels, MP and PH regions are mixed with HW pixels, HW regions are mixed with MP pixels. Figure 6.8(c) shows the result obtained with tight ground truth: the per-pixel error of the first-stage classifier is 22.9%; compared to Figure 6.8(b), BL regions are much purer, MP and PH regions have less HW pixels in them, but HW regions are mixed with more MP pixels. the error of the second-stage classifier is 17.2%; the error of the third-stage classifier is 15.4%; and error of the fourth-stage classifier is 14.4%. The final MP, PH and HW masks extract their content types well, as shown in (g)-(i), except for some small patches of HW misclassified as MP, and some small patches of PH misclassified as

(a) test image (b) 1st stage classification (with loose GT) (c) 1st stage classification (with tight GT)



(d) 2nd stage classification (e) 3rd stage classification (f) 4th stage classification

(g) MP masked (h) PH masked (i) HW masked

Figure 6.8: A document image with a complex non-rectilinear page layout, contains content of MP, HW, PH and BL. Tighter ground truth drops the error rate of this image from 36.7% to 22.9%, a drop of 38%. The final MP, PH and HW masks extract their content types well, as shown in (g)-(i).

MP or HW.

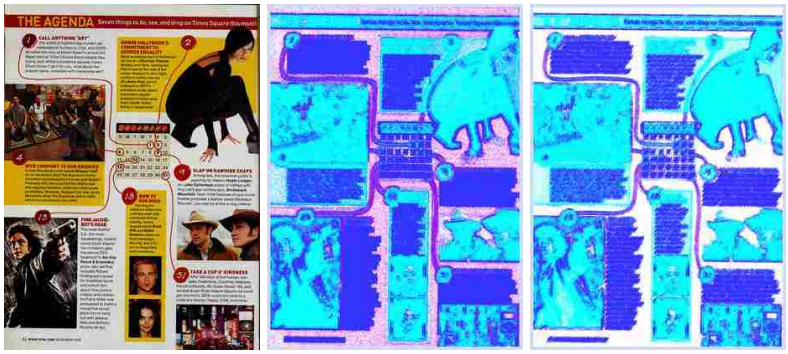
Figure 6.9 shows results on a color image of a movie magazine page containing complex non-rectilinear regions. With loose ground truth, the per-pixel classification error of the first-stage classifier is 32.5%. And the background is mixed with HW. With tight ground truth, the per-pixel error of the first-stage classifier is 25.2%; the error of the second-stage classifier is 18.9%; the error of the third-stage classifier is 17.7%; and error of the fourth-stage classifier is 17.7%. This error is possibly due to the lack of training sample of MP written in red color on a yellow background. For curvature preservation, notice the small red circles containing numbers: their curvature changes slightly.

Figure 6.10 gives the representation of total error rate as a function of stages of classification. The post-classifiers reduce the error rate by 23.4%.

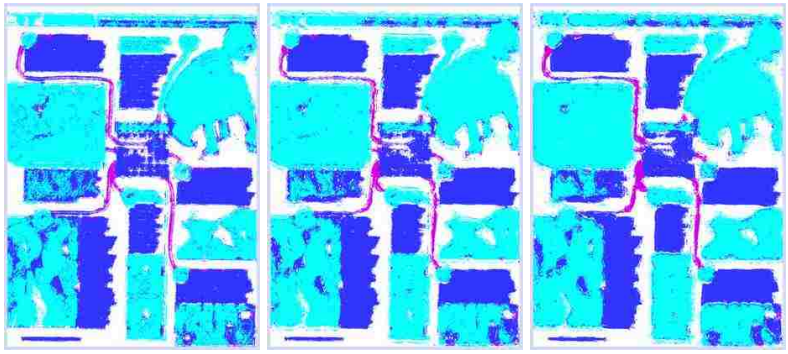
6.4 Statistical Significance of Claimed Improvements

We verify the statistical significance of improvements due to iterated classification using a statistic two sample *t*-test [BD77]. The *t*-test is commonly applied to assess whether the means of two populations are statistically significantly different from each other, using an estimate of standard deviation based on sample size. The null hypothesis of a *t*-test is that the two populations have equal means. When the two populations are of the same size, as is true in our test, the *t* statistic is calculated as follows:

$$t = \frac{\overline{X_1} - \overline{X_2}}{\sqrt{\frac{S^2_{X_1} + S^2_{X_2}}{n}}} \tag{6.4}$$



(a) test image (b) 1st stage classification (with loose GT) (c) 1st stage classification (with tight GT)



(d) 2nd stage classification (e) 3rd stage classification (f) 4th stage classification

(g) MP masked (h) PH masked (i) HW masked

Figure 6.9: A magazine page with a complex non-rectilinear page layout, containing content of MP, PH and BL. The MP mask extracts its content class well, except for three patches misclassified PH. For curvature preservation, notice the small red circles containing numbers: their curvature changes slightly.

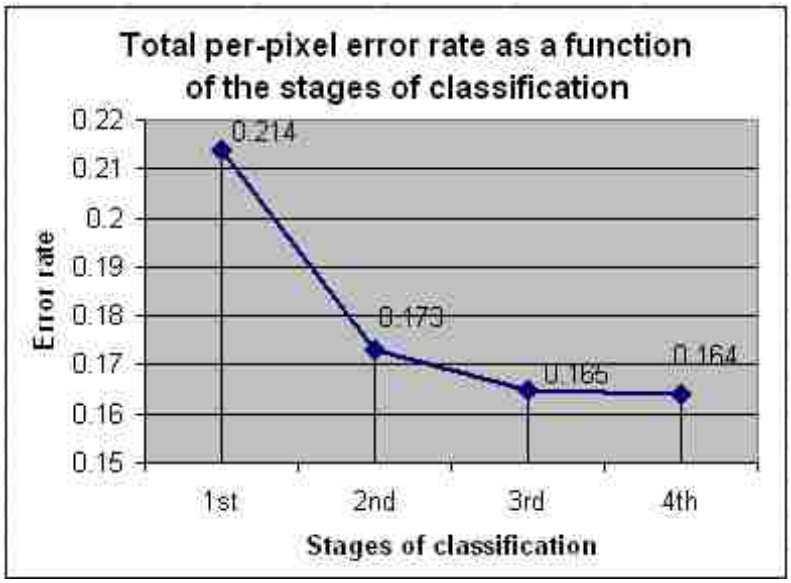


Figure 6.10: Total error rate averaged over the test set of 83 images, as a function of the stages of classification. After four stages of classification, the error rate has fallen from 0.214 to 0.164, a drop of 24%.

where $\overline{X_1}$, $\overline{X_2}$ are the means of the two populations X_1, X_2, respectively; and S_{X_1}, S_{X_2} are the sample standard deviations of X_1, X_2. Once a *t* value is determined, a p-value can be found using a table of values from Student's t-distribution. The *p*-value is the probability, under the null hypothesis, of observing a value as extreme or more extreme than the test statistic. We perform the *t*-test using MATLAB®, and report *p*-values here. The results of the *t*-test comparing the first stage and the fourth stage of iterated classification, using various ground-truth policies, are shown in Table 6.1. This table shows that the improvement is statistically significant when iterated classifiers are trained with ground truth policies of "d=2", "d=4","d=8", tight, and loose, but not for pixel-accurate and "d=1".

This training instability suggests that iterated classifiers are sensitive to ground truth, and do not work for pixel-accurate ground truth. We are not convinced that the current feature extraction window is a good fit to pixel-accurate ground truth. Our proposition of iterated classification starts with loose ground truth. The scale of features we have been using is systematically chosen (see Section 4.6) under the guidance of ground truth that includes some blank pixels as part of foreground. The success of the loose, tight, and morphological expansion ground truth shows the effectiveness of such features. However, for iterated classifiers to work with pixel-accurate ground truth, the scale of features should be re-explored. This can be done using the the same method discussed in Section 4.6.

	h	p-value
Pixel-Accurate	0	0.1795
d = 1	0	0.7946
d = 2	1	1.2740e-005
d = 4	1	1.2513e-009
d = 8	1	1.4500e-017
Tight	1	4.2537e-009
Loose	1	5.5349e-008

Table 6.1: The result of the t-test is returned in h: h = 1 indicates a rejection of the null hypothesis at the 5% significance level. h = 0 indicates a failure to reject the null hypothesis at the 5% significance level. This table shows that the improvement is statistically significant when iterated classifiers are trained with ground truth policies of "d=2", "d=4","d=8", tight, and loose; but not for pixel-accurate and "d=1".

6.5 Final Experiment

6.5.1 Motivation

The final experiment scaled our training and test set sizes up by nearly twice with a goal of including images from more sources and writing systems, and challenging iterated classifiers with more difficulties. We upgraded our ground truth technique from swapping overlapping rectangles using zoner to labeling using Parc's PixLabeler [SLS09] and expanding foreground pixel-accurately. This choice is justified by the notion that the current feature set for iterated classifiers is a good fit for the morphological expansion of "d=8". The ground truth techniques and policies are discussed in detail in Section 5.7 and 5.8.

We have experimented with 219 page images: 62 images were placed in the training set, and 157 images in the test set. The scale of this experiment is nearly twice as large as the previous experiment. We included images from more writing

systems: we used to include English, Arabic and Chinese, now we added Hindi and Korean.

In this experiment, the ground truth was obtained in two steps: in the first step, we applied the PARC PixLabeler tool to generate pixel-accurate ground truth, in which only foreground pixels are labeled; in the second step, we expanded foreground pixels by applying morphological dilation operations [SS94] with a circular disk structuring element, that is, background pixels within eight pixels of a foreground pixel are labeled as the same class as the foreground pixel. The training data was decimated randomly by selecting only one out of every 9000th training sample, as usual.

6.5.2 Results

As shown in Figure 6.11 the overall per-pixel error rate of the fist-stage classifier for this experiment is 20.2%; the error of the second-stage classifier is 15.7%; the error of the third-stage classifier is 15.4%; the error of the fourth-stage classifier is 15.4%; the error of the fifth-stage classifier is 15.2%. We see that the error decreases monotonically, with a drop of 24.5% from the first stage to the fifth stage. We carried out a statistical two-sample *t*-test (discussed in Section 6.4) to verify the statistical significance of improvement in the final experiment. The *p*-value is $7.8736e^{-16}$, which indicates that the improvement is statistically significant.

The following Figures 6.12, 6.13 and 6.14 are some sample test images from this set, highlighting classifier successes and failures. In all the examples, the test image is shown on the upper left, followed by a sequence of five "false color" classification images in the order of stages. Figure 6.12 shows that iterated classifiers successfully

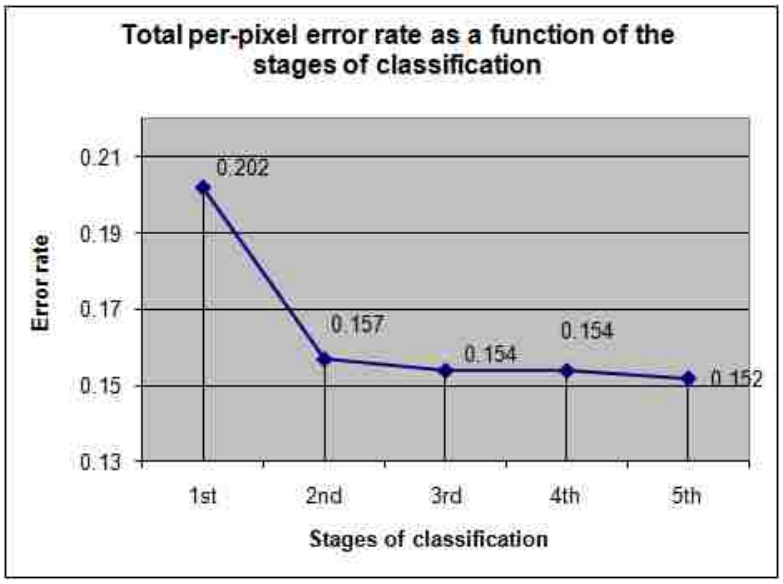


Figure 6.11: Total error rate averaged over the test set of 157 images, in a function of the stages of classification. After four stages of classification, the error rate has fallen from 0.202 to 0.152, a drop of 24.5%.

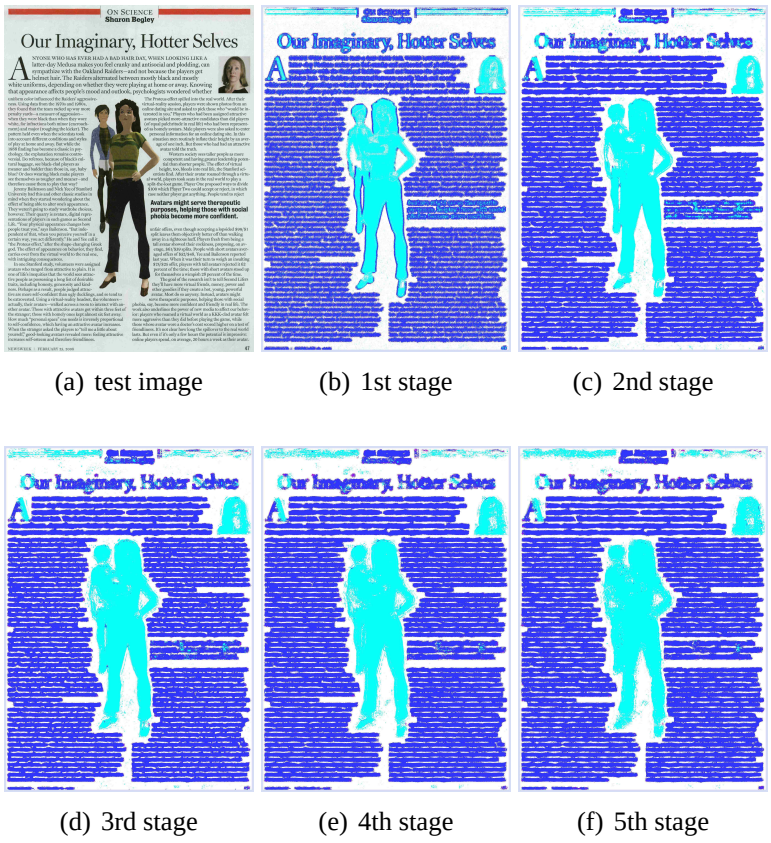
correct misclassification of MP over PH, and vice versa. Figure 6.13 shows that iterated classifiers improves the result on a hard case of misclassification between HW and MP. Figure 6.14 shows an example of iterated classifiers fails to correct the misclassification of MP over HW.

6.5.3 Run Times

Running our classifier on a test set with hundreds of test images is not feasible for a single machine and we have become dependent on the use of a Beowulf cluster of machines at Lehigh [HPC10] to complete large scale experiments. The Beowulf cluster we use currently consists of 320 64bit Xeon processors. For the run time of the final experiment, the classifier we have been using (discussed in Section 4.2) requires on average 220 CPU minutes per test image per iterated classification stage. The runtime can be reduced by a factor of as least 20 using Dawei Yin's bin-decimation classifier [YBA10, YAB10].

6.5.4 Conclusions

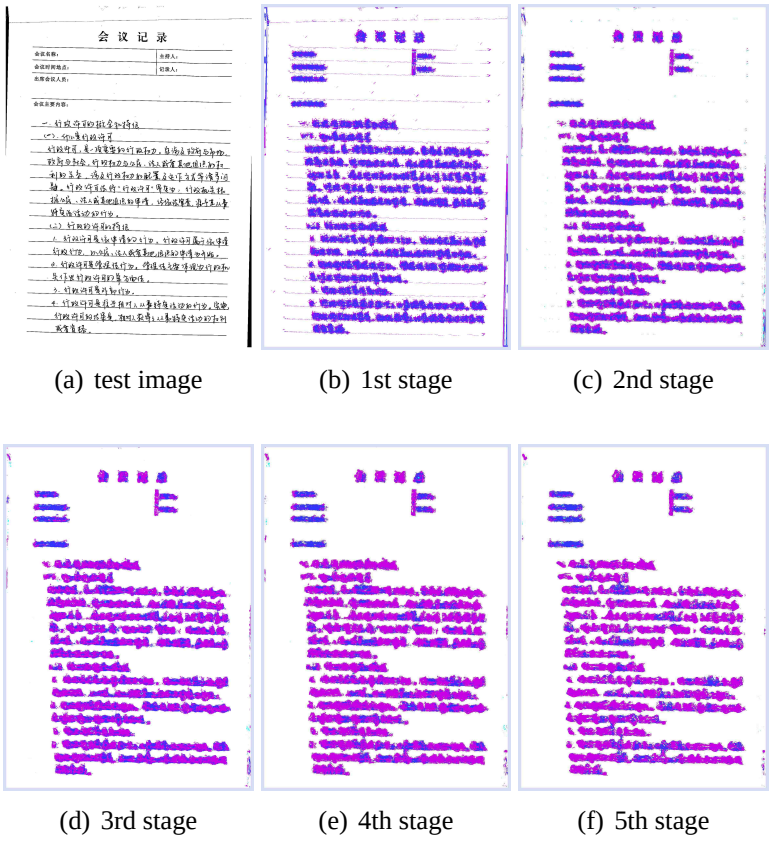
This experiment successfully scaled up the size of our experiments to train on nearly twice as many images and two more languages as previous experiments. Iterated classifiers continue to drop the overall error rate, by 24.5%. As always, Arabic handwriting and machine-print images introduce problems of mixing HW and MP. Although the classifiers failed on some hard cases, overall, they can improve the results. This increase our confidence that iterated classifiers can reduce the error rate by a range of 22% - 25.5%.



(a) test image (b) 1st stage (c) 2nd stage

(d) 3rd stage (e) 4th stage (f) 5th stage

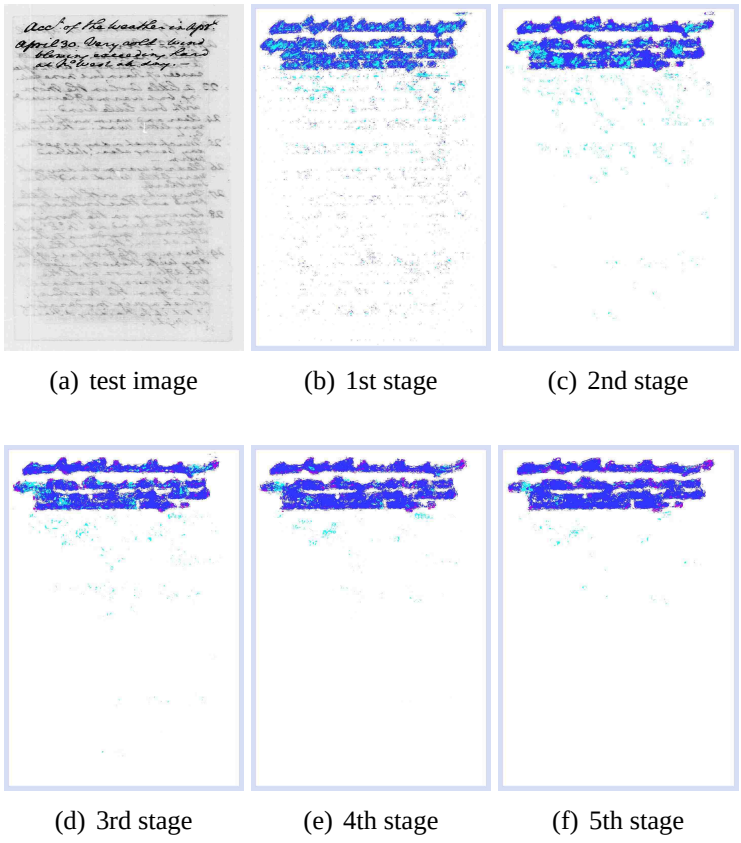
Figure 6.12: A magazine page from the test set of ICDAR2009 Page Segmentation Competition. The original image (a) is in full color. The results of classification are shown (b)-(f). At the first stage, some MP pixels, especially these lying within the titles or large font letters, are labeled PH. On the boundary of the photo in the center of the image, the PH pixels are labeled MP. By the fifth stage, most misclassified MP pixels are correct labeled; almost all PH pixels on the boundary are correctly labeled.



(a) test image (b) 1st stage (c) 2nd stage

(d) 3rd stage (e) 4th stage (f) 5th stage

Figure 6.13: A minute page written in Chinese, containing HW and MP contents. At the first stage, HW and MP text regions are located, but their pixels are labeled as mixture of HW or MP. This is difficult, even for human readers, to distinguish one content type from the other. By the fifth stage, most misclassified HW pixels are correct labeled, except for several clusters of characters; most misclassified MP pixels are corrected, except for these lying within the relatively large font title. Note that the pixels on rules are labeled HW in the first stage, and corrected as BL by the fifth stage.



(a) test image (b) 1st stage (c) 2nd stage

(d) 3rd stage (e) 4th stage (f) 5th stage

Figure 6.14: A handwriting script from the test set of DIBCO09 (Document Image Binarization Competition). At the first stage, most HW pixels are labeled MP or PH. Some BL pixels are labeled PH, which is probably cause by bleed-through from the other side. By the fifth stage, most HW pixels are labeled MP. The BL pixels are correctly labeled.

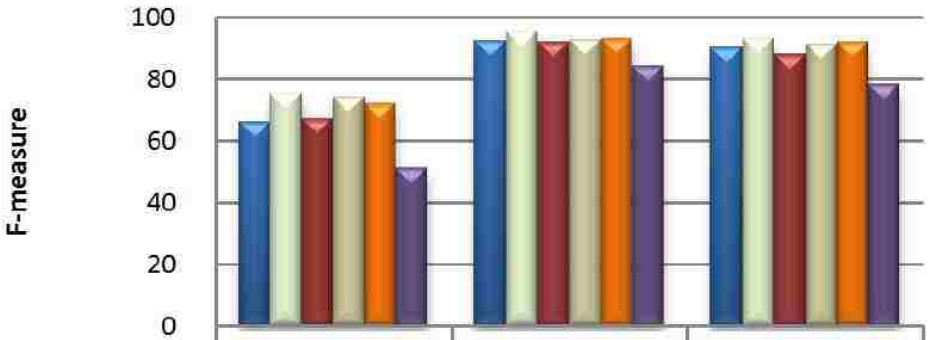
6.6 ICDAR 2009 Page Segmentation Competition

Our method (DICE and IC as a whole) was tested along with five others in the ICDAR2009 Page Segmentation Competition and the results was reported by Antonacopoulos *et al* in detail in [APBP09]. We summarize the results in this section.

Participating methods in this competition included the Fraunhofer Newspaper Segmenter [GDPP05, ZLDP01, Bre02, JY98], the REGIM-ENIS method, the Tesseract method [Smi09], ABBYY FineReader Engine®8.1, and OCRopus 0.3.1. It should be noted that our DICE system is designed as a first step in document analysis, intended to be executed before "classical" layout analysis methods which decompose text into blocks, determine reading orders, etc. Therefore, precision/recall metrics are appropriate for evaluating our algorithm, rather than higher-level metrics [AB07] that consider region structure and penalize violations of reading order. The F-measure was chosen as one of the evaluation metrics in the competition. In statistics, the F-measure is a measure of a test's accuracy. It considers both the precision and the recall of the test to compute the score. The traditional F-measure is the harmonic mean of precision and recall:

$$F = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{6.5}$$

The F-measure report is shown in Figure 6.15. This shows that our DICE system is competitive: our system scored 90.09, while the highest score was 93.14 and the lowest score 78.35.



	Non-text	Text	Overall
DICE	66.22	92.21	90.09
Fraunhofer	75.15	95.04	93.14
REGIM-ENIS	67.13	91.73	87.82
Tesseract	74.23	92.50	91.04
FineReader	71.75	93.09	91.90
OCRopus	51.08	84.18	78.35

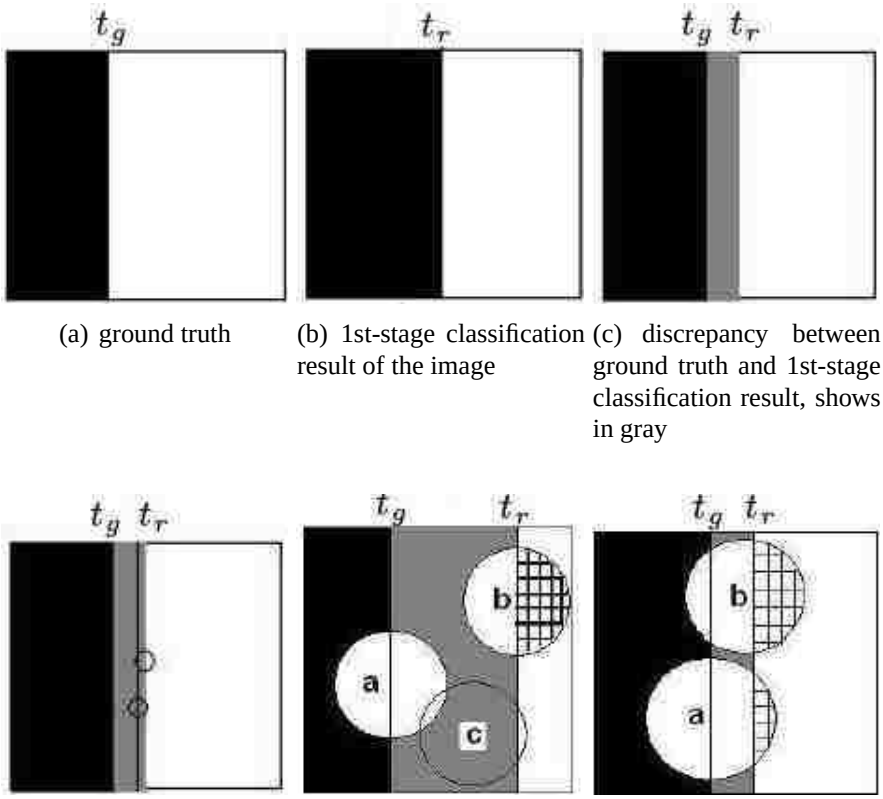
Figure 6.15: F-measure of the four methods submitted to the ICDAR 2009 Page Segmentation Competition and two state-of-the-art methods. (Courtesy of ICDAR 2009 Page Segmentation Competition.)

Chapter 7

Performance Analysis of Iterated Classification

One of our previous experiments shows that the post-classifiers reduce the per-pixel classification errors by 23%, running a four-stage classification on 83 test images. Another experiment with fewer test images shows that per-pixel errors can fall monotonically for as many as eight stages. We notice that, as uniformity improves in local regions, boundaries tend to remain stationary – that is, they do not drift. This observation leads us to try to prove that there exist iterated classifiers that are guaranteed to converge to the ground-truth boundary.

We begin the investigation by simulating an image containing two content-classes, say MP and BL, and we have a classifier trained and tested on this image. The ground truth and first-stage classification result for this image are shown schematically in Figure 7.1(a)-(b); MP pixels are colored black, BL pixels are colored white. In Figure 7.1(a)-(d), t_g marks the horizontal coordinate of the boundary



(a) ground truth

(b) 1st-stage classification result of the image

(c) discrepancy between ground truth and 1st-stage classification result, shows in gray

(d) feature extraction windows and new boundary

(e) the width of the discrepancy is greater than the radius of feature extraction windows

(f) the width of the discrepancy is smaller than the radius of feature extraction windows

Figure 7.1: Analysis of convergence of iterated classification. Black represents MP, white represents BL. In figure (c)-(e), the discrepancies between ground truth and the classification results for the image are colored gray. In figure (e) and (f), circles represent feature extraction windows. In each figure, t_g marks the horizontal coordinate of the boundary in the ground truth and t_r marks the horizontal coordinate of the boundary in the classification results for the image. Initially, $t_g < t_r$.

in the ground truth and t_r marks the horizontal coordinate of the boundary in the classification results for the image. In Figure 7.1(c)-(f), gray regions represent the discrepancies between ground truth and the classification results for the training image.

Given the ground truth and results of the first-stage classifier, we can analyze how the second-stage classifier performs. Recall that features are extracted within a local window (a circle of radius R) centered on the target pixel.

7.1 Analysis of the Second-Stage Classifier

We start by analyzing the case where the width of the discrepancy is greater than R, *i.e.* $t_r - t_g > R$, as shown in Figure 7.1(e). For the post-classifiers, we consider one feature that we have been using in experiments: the number of BL pixels within the right half of the feature extraction window. Recall that all features are extracted from the results of classification.

Consider these different cases of a target pixel depending on its ground-truth class, labeled class from classification results, and the number of BL pixels within the right half of the feature window.

Case I: Target pixel is ground-truthed MP, classified MP, and contains no BL pixels within the right half of its feature window.

Case II: Target pixel is ground-truthed BL, classified BL, and all pixels within the right half of its feature window are BL.

Case III: Target pixel is ground-truthed BL, classified MP, and contains no BL pixels within the right half of its feature window.

Case IV: Target pixel is ground-truthed BL, classified MP, and contains at least one BL pixel within the right half of its feature window.

For pixels that fall outside the discrepancy region, the classification is obvious: pixels in case I, *i.e* those in the black region in Figure 7.1(c), are still labeled MP; pixels in case II, *i.e* those in the white region in Figure 7.1(c), are still labeled BL.

For pixels within the discrepancy region (ground-truthed BL but classified MP by the first-stage classifier), part of them will be correctly classified using the feature, as follows:
If the right half of its feature extraction window contains any BL pixels – case IV – the target pixel is then classified BL, because its feature value is different from that of pixels in case I. For example: in Figure 7.1(e), the pixel centered on circle *b* and *c* is labeled BL. If the right half of its feature extraction window contains no BL pixel – case III – the pixel is still classified MP because its feature value is the same as that of pixels in case I. For example: in Figure 7.1(e), the pixel centered on circle *a* is labeled MP. Pixels that are less than R pixels left from the boundary t_r are in case IV, and are thereby are labeled BL.

After the second-stage classification, the horizontal coordinate of the resulting boundary would be $t_r - R$, which moves towards ground-truth boundary t_g by a distance of R pixels.

7.2 Analysis of Classifiers Following Second-Stage

As long as the width of the discrepancy is greater than R, each succeeding classifier must behave the same as the second-stage classifier and cause the boundary to move

again towards t_g by R.

When the the width of the discrepancy is smaller than R, *i.e.* $t_r - t_g < R$, we must consider more cases, as follows:

Case V: Target pixel is on boundary t_g, ground-truthed MP, classified MP, and contains a number, say B, of BL pixels within the right half of its feature window.

Case VI: Target pixel is ground-truthed MP, classified MP, and contains more than one but less than B of BL pixels within the right half of its feature window.

Case VII: Target pixel is within the discrepancy, ground-truthed BL, classified MP, and contains more than than B of BL pixels within the right half of its feature window.

Pixels that fall outside the discrepancy are classified in this way: pixels in cases I, V and VI are still labeled MP; pixels in case II are still labeled BL.

Pixels within the discrepancy will be classified BL: all of them are in case VII, and their feature values are different from that of ground-truthed MP pixels in cases I, V and VI, therefore the classifier must classify them BL. This is illustrated in Figure 7.1(f): the center pixel of circle *a* lies on the left boundary of the discrepancy area will be classified MP, following its ground-truthed content; circle *b* has more BL pixels in its right half than circle *a* does, therefore the center pixel of *b* can be discriminated and classified BL; for the same reason, pixels in the discrepancy, but not on its left boundary, are to be classified BL. Consequently, the boundary in the classification result moves towards t_g by $t_r - t_g$: *the boundary of the classification result has converged to ground truth*.

Simulation shows the same behavior as the analysis above suggests. We simulated a discrepancy of 174 pixels wide, and a feature extraction (circular) window

of radius 20. For the first eight stages, the boundary moved left by 20 pixels in each stage of classification. At the ninth stage, the boundary moved left by 14 pixels, which converged exactly to the ground-truth boundary.

In summary, analysis of special cases, experiments and simulations, behave as the classifiers appear often to do. That is, with proper choice of features and guidance by the ground truth, there exists a sequence of post-classifiers that refine the obtained results and force them to converge to ground truth. This further implies that post-classifiers can converge linear boundaries, oriented at any direction, to ground truth. We conjecture that for all region shapes, whose radius of curvature is bounded below, there exists a similar training methodology such that all boundaries converge to ground truth. Some of the experiments show that the post-classifiers also converged on regions with small radii of curvature. For example, in Figure 6.9 the small red circles containing numbers are preserved.

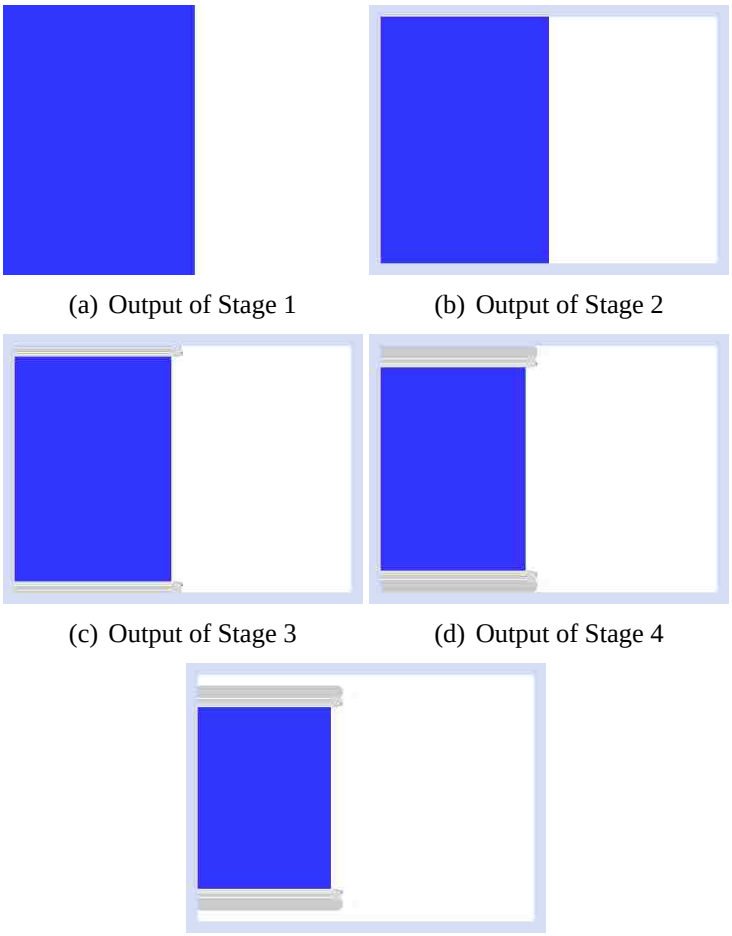
We also conjecture that to converge to ground truth, the number of post-classifiers needed is proportional to the width of the discrepancies, and inversely proportional to the radius of the feature extraction window.

7.3 Comparison of Iterated Classication with Repeated One

Recall that iterated classification is proposed as a refinement to repeated classification. Empirical results have shown that iterated classification is superior to repeated classification. In this section, we analyze the causes of this using simulation.

One of our earlier experiment showed that repeated classification is unstable in

improving the results of the DICE classifier. It can succeed in some images, but fail in others. We conceive that this failure occurs because repeated classification makes an unrealistic assumption: the same error will occur in the same context again and again. To illustrate this, we again simulated an image containing only MP and BL; the result and the ground truth had a discrepancy of 28 pixels wide. The feature extraction (circular) window was in radius of 13 pixels. The post-classifier was trained once, and then repeatedly applied to the result of the test image. The sequence of the result is shown in Figure 7.2. As the test image is reclassified, the boundary between MP and BL keeps moving left, each time by a width of 13 pixels. This simulation suggests that repeated classification, unlike iterated classification, may never converge to ground truth.



(a) Output of Stage 1 (b) Output of Stage 2

(c) Output of Stage 3 (d) Output of Stage 4

(e) Output of Stage 3

Figure 7.2: Illustration of repeated classification.

Chapter 8

High Recall Document Content Extraction

In this Chapter, we focus on obtaining high recall masks for each of three content types—machine-print (MP), handwriting (HW), and photo (PH)—in order to support downstream processing of each separately. We previously reported overall per-pixel accuracy of 85% [AB08]; here, we report achieving at least 89% recall and at least 87% precision among all three content types.

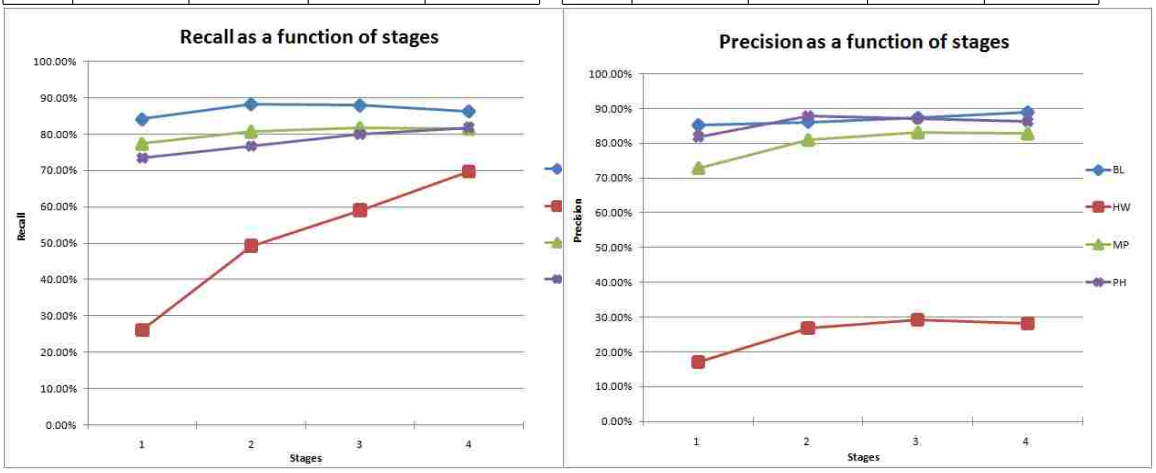
8.1 Iterated Classifiers Improve Recall and Precision

Recent experiments have indicated that iterated classifiers can increase accuracy stage by stage [AB08]. The following analysis of the results shows that iterated classifiers increase recall stage by stage without causing precision to decrease. In Figure 8.1, the table and figure of recall are shown on the left, and precision on the right. The recall of all three principal content types, MP, HW and PH, increases

stage by stage, except for the fourth-stage of MP, which decreases by 0.2%. Meanwhile, the precision of these content types also increases.

	1st-stage	2nd-stage	3rd-stage	4th-stage
BL	84.11%	88.28%	87.95%	86.33%
HW	26.06%	49.22%	59.01%	69.65%
MP	77.32%	80.64%	81.61%	81.41%
PH	73.46%	76.66%	79.89%	81.83%

	1st-stage	2nd-stage	3rd-stage	4th-stage
BL	85.13%	86.09%	87.28%	88.91%
HW	17.08%	26.84%	29.31%	28.09%
MP	72.77%	80.95%	83.13%	82.76%
PH	81.89%	87.84%	87.05%	86.29%



(a) Recall (b) Precision

Figure 8.1: Recall and precision of each content type obtained in each stage of iterated classification. The results suggest that the iterated classifiers improve the recall of all content types. Especially, the recall of HW is significantly improved. Note that the precision even improves a little when the recall increases as a function of stages.

8.2 Proposed Testing Policy Changes

In this section, we first discuss a straightforward testing policy change, a multi-stage voting rule, that can be utilized to increase recall. Afterwards, we introduce a testing policy of accepting blank ("don't care") (BL) pixels as pixels of each particular content types such as HW, MP and PH.

8.2.1 Multistage Voting Rule

We investigated multistage voting rule which votes on the results of stages of the iterated classification, that is, given a test sample *t* and a content class *i* ($i \epsilon \{MP, HW, PH, BL\}$), if in any stage *t* is classified as class *i*, then *t* is finally classified as class *i*.

This technique increases the recall of single content class moderately. As shown in our previous experiments, iterated classifiers gradually and slowly expand or shrink the areas of different content classes, without causing the boundaries shift back and forth. Empirically, later classifiers are more confident than earlier one. Another advantage of this technique is that voting from the results of each stage is simple and fast – it merely requires voting for each one of the content classes.

However, the only concern is that the result of biased voting might be insignificant compared to the result of the last stage of iterated classification. This concern results from the observation that a four-stage iterated classification drops the error rate by nearly a quarter. Therefore, we are not confident that this approach can lead to as high recall as we want.

8.2.2 Truthing and Scoring Policy for Blank Pixels

There is growing debating on pixel-accurate ground truth. As discussed in detail recently [MBA08, ACL10, Smi10], truthing policies for page segmentation seem often to possess ambiguities and inconsistencies.

Blank pixels especially have an ambiguous status during manual ground-truthing: blank regions often make up the “background” for MP and HW “foreground” pixels, and furthermore BL pixels often interpenetrate PH regions. Therefore, blank

pixels are usually called “don’t care” pixels. This seems inevitable when ground truth is loose (using, say, rectangular zones). Experiments with our earlier classifier show that blank space are often confused with handwriting and in more limited cases also with machine print.

We propose a modification in our training and testing methods applied to blank pixels. During the training phase, blank pixels are inevitably treated as a distinct class and so ground-truthed and trained separately from the other classes. However, during the testing phase, pixels which are classified as blank are viewed as “don’t care” pixels, and so are combined with each of the other classes: so, for example, all pixels classified as BL are accepted also as MP pixels (and similarly for HW and PH). Note that after this policy chagne, the masks obtained are no longer disjoint; that is, they can share pixels.

Our modified policy, in which blank pixels are assigned to all the other classes, leads to intuitive results, which look natural to the eye and, we believe, will not cause difficulties for any later stages of processing; this is illustrated in Figures 8.4 and 8.5.

8.3 Experimental Results

We use a training set of 33 images and a distinct test set of 83 images, which are the same images we used in [AB08]. Together the two sets contain machine-print (MP), handwriting (HW), photograph (PH) and blank (“don’t care”) (BL). The training data was decimated randomly by selecting one out of every 9000th training sample (as discussed in Section 4.3).

Remember that, using the original scoring police, we evaluated performance using per-pixel accuracy, precision and recall, defined as follows. Per-pixel accuracy is the fraction of all pixels in the document image that are correctly classified (Equation 6.1). Unclassified pixels are counted as incorrect. Recall of a content class is defined as the number of pixels that are correctly classified divided by the total number of pixels that are ground truthed as belonging that class (Equation 6.3). Precision of a content class is defined as the number of pixels that are correctly classified divided by the total number of pixels classified as belonging to that class (Equation 6.2).

Note that under the new "don't care" BL pixel scoring policy, precision and recall must now be computed differently. For example, the precision of HW is calculated as shown in Equation 8.1, where classifying HW pixels as BL or BL pixels as HW is not penalized, but classifying MP or PH pixels as HW or BL is penalized. Under this policy, precision P_i of a foreground class i ($i \epsilon \{MP, HW, PH\}$), is now calculated as shown in Equation 8.1, where $N_{i,j}$ denotes the number of pixels that are ground truthed as class i and classified as class j ($j \epsilon C = \{MP, HW, PH, BL\}$).

$$P_{HW} = \frac{N_{HW,HW} + N_{BL,BL} + N_{HW,BL} + N_{BL,HW}}{(N_{j,i} + N_{j,BL})}$$

$$P_i = \frac{N_{HW,HW} + N_{BL,BL} + N_{HW,BL} + N_{BL,HW}}{\sum_{j \epsilon C}(N_{j,HW} + N_{j,BL})} \tag{8.1}$$

Under this new policy, recall R_i of a foreground class i ($i \epsilon \{MP, HW, PH\}$),

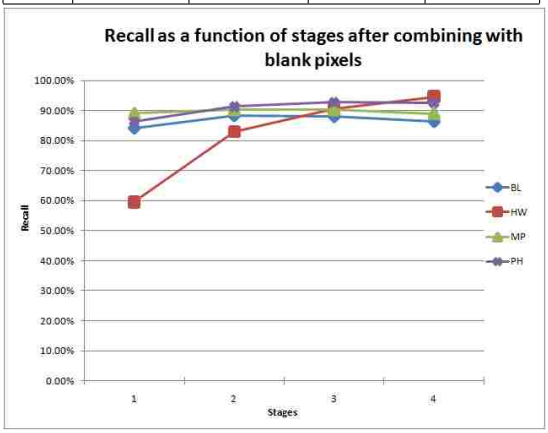
is now calculated as shown in Equation 8.2, where $N_{i,j}$ denotes the number of pixels that are ground truthed as class i and classified as class j ($j \epsilon C = \{MP, HW, PH, BL\}$).

$$R_i = \frac{N_{HW,HW} + N_{HW,BL}}{\sum_{j \epsilon C} N_{j,HW}} \tag{8.2}$$

After applying the new blank truthing policy and rescoring accordingly, recall improved inevitably on HW, as well as on the others, show in Figures 8.2 (the BL scores are shown for comparison, and are unchanged from Figures 8.1). We always knew that most confusions involving HW pixels also involved BL pixels; this new way of interpreting the results has revealed the extent to which HW was not significantly confused with either MP or PH. We were pleased that, at the same time, rather surprisingly, precision improved for all three of MP, PH, and HW.

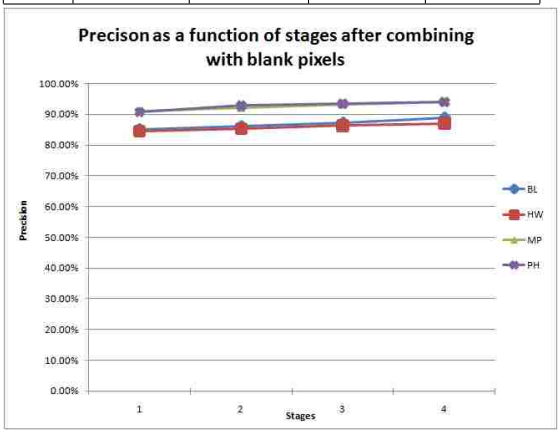
The second policy change was to the classifier, and involved voting among the stages of iterated classifiers. The particular voting policy we report here was the most aggressive one: if any classifier stage decided that a pixel was HW, then we finally classified it as HW; and similarly for MP and PH. This policy can be implemented independently of the blank pixel truthing policy, so there are four cases of combining these two changes: in Figures 8.3, we show the results on HW only; the four cases are (a) the original method (labeled simply "HW"), (b) aggressive voting among all stages ("H&S"), (c) accepting BL pixels as HW ("H&B"), and finally (d) adopting both multi-stage voting and BL pixel accepting("H&S&B"). Very pleasantly it turned out that the fourth technique achieved the highest recall, of 95.31%, far higher than any previous result we have achieved on HW, and nearly tied with the highest recall for either of MP and PH. Under the same conditions, precision on HW fell slightly from 87.1% to 83.5%.

	1st-stage	2nd-stage	3rd-stage	4th-stage
BL	84.11%	88.28%	87.95%	86.33%
HW	59.56%	82.91%	90.67%	94.63%
MP	89.09%	90.30%	90.32%	89.02%
PH	86.36%	91.31%	92.93%	92.53%



(a) Recall

	1st-stage	2nd-stage	3rd-stage	4th-stage
BL	85.13%	86.09%	87.28%	88.91%
HW	84.48%	85.34%	86.26%	87.07%
MP	90.91%	92.17%	93.24%	93.97%
PH	90.79%	92.82%	93.48%	93.93%

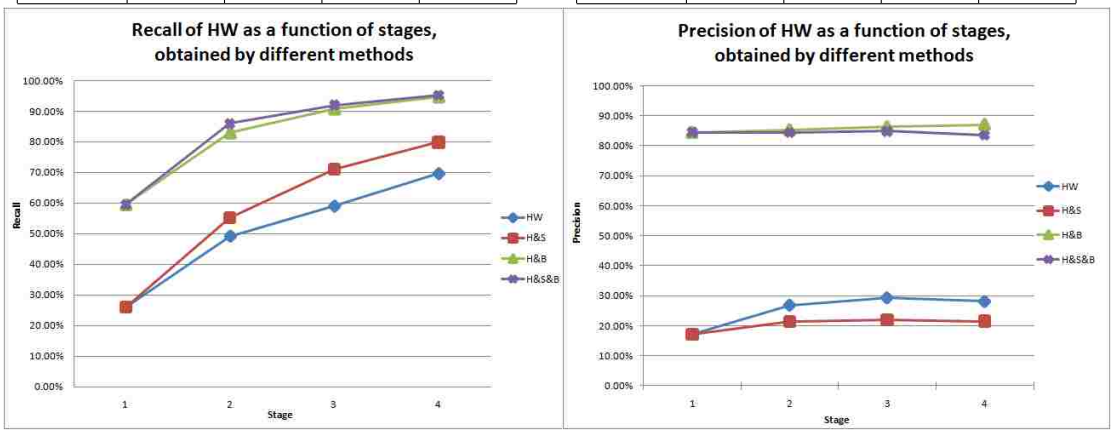


(b) Precision

Figure 8.2: Recall and precision of each content type obtained in each stage after combining with BL pixel; for example, the mask of HW contains all pixels that are classified HW or BL; accordingly, a pixel is correctly classified if it is ground-truthed HW /BL and classified HW/BL. The results suggest that the iterated classifiers improve the recall of regions of each content type to about 90%. The recall of handwritten regions is significantly improved to 94.63%. Note that the precisions are nearly 90% under this policy.

	stage 1	stage 2	stage 3	stage 4
HW	26.06%	49.22%	59.01%	69.65%
H&S	26.06%	55.29%	71.01%	79.83%
H&B	59.56%	82.91%	90.67%	94.63%
H&S&B	59.56%	85.96%	92.01%	95.27%

	stage 1	stage 2	stage 3	stage4
HW	17.08%	26.84%	29.31%	28.09%
H&S	17.08%	21.32%	21.97%	21.52%
H&B	84.48%	85.34%	86.26%	87.07%
H&S&B	84.48%	84.35%	84.84%	83.52%



(a) Recall (b) Precision

Figure 8.3: Recall and precision of handwritten regions obtained by different method as a function of stages. HW is simply the result of each stage, which is our baseline. H&S is a voting results of all available stages, take the 2nd-stage as an example, a pixel is classified HW if it is classified HW in either 1st-stage or 2nd-stage. H&B is described before, that is the results of combining HW with BL pixels. H&S&B is the results of combining H&S with blank pixels. Similar to previous results, the precision remains or improves a little where as the recall increases as a function of stages.

The results of two image examples are illustrated in Figure 8.4 and Figure 8.5. Note that neither of these images is well represented in the training set, but their content is still extracted with high recall and good precision.

Note that the masks obtained from the fourth stage of iterated classification are partitions of an original image. However, the new masks obtained by applying both policy changes are no longer partitions. They now share pixels, which consist of two types of pixels: (a) "don't care" blank pixels; and (b) pixels that are assigned to multiple classes.

Each figure contains nine images of three types: (a) the original image shown on the upper-left; the "false-color" classification results of 1st-stage (b) and 4th-stage (c), machine print (MP) is dark blue, handwriting (HW) red, photographs (PH) light blue-green, and blank (BL) white; *mask* images extracted from the 4th-stage for MP(d), PH(e), and HW(f) content classes, and those final *mask* images as a result of the combination of policy changes for MP(g), PH(h), and HW(i) content classes.

8.4 Discussion

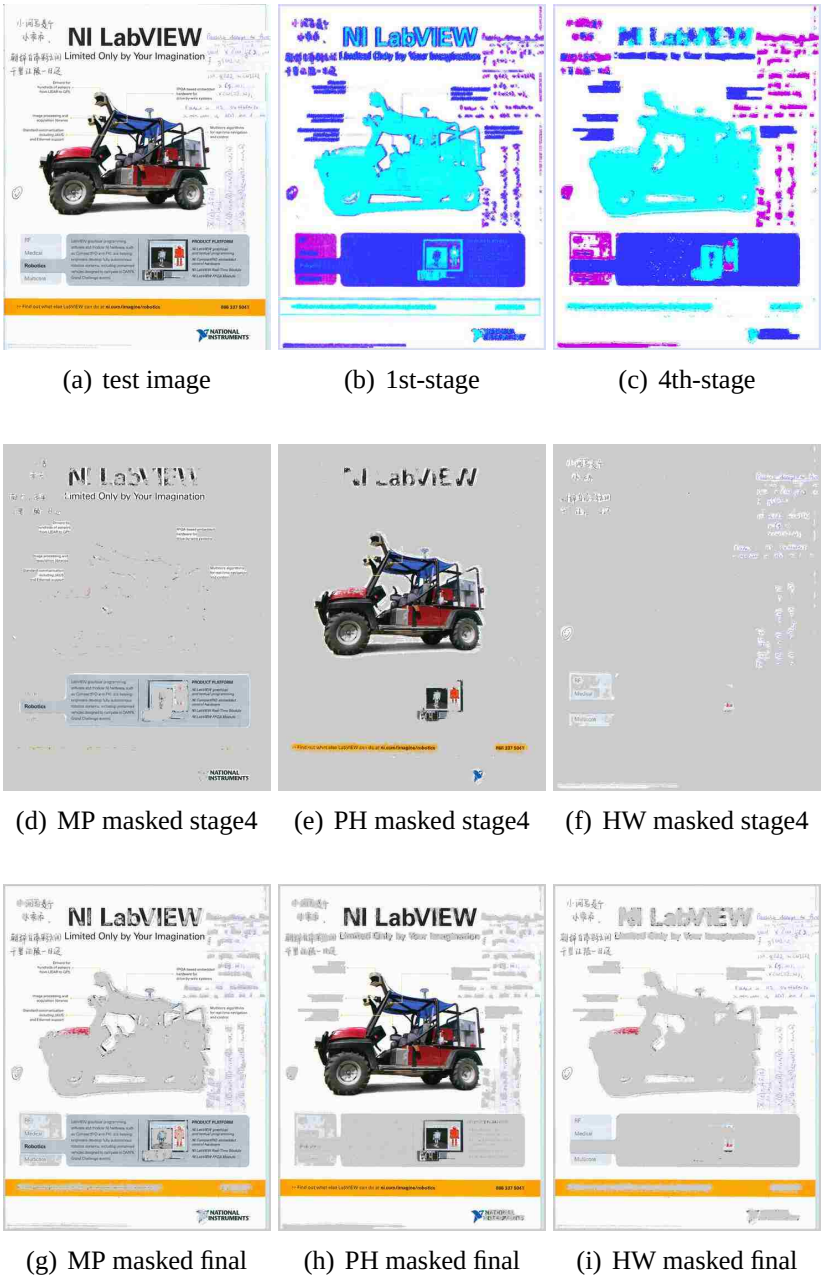
We show that iterated classifiers increase both recall and precision on all three principal content types, stage by stage. Two policy changes, the multi-stage voting rule and blank truthing policy, improve the results of DICE on both recall and precision. This new policy yields results intuitively pleasing to the eye, and should not cause ambiguity or confusion for downstream processing. This is justified by the notion that iterated classification provides high recall and precision on BL pixels.

(a) test image (b) 1st-stage (c) 4th-stage

(d) MP masked stage4 (e) PH masked stage4 (f) HW masked stage4

(g) MP masked final (h) PH masked final (i) HW masked final

Figure 8.4: Illustration of improved recall of each content types. A skewed document image with a complex non-rectilinear page layout contains content of MP, HW (in English and Chinese, horizontal and vertical), PH and BL. The final MP, PH and HW masks, extracted from the results of combining two policy changes, are shown in (g)-(i), which yields higher recall than 4th-stage without causing confusion.



(a) test image (b) 1st-stage (c) 4th-stage

(d) MP masked stage4 (e) PH masked stage4 (f) HW masked stage4

(g) MP masked final (h) PH masked final (i) HW masked final

Figure 8.5: A magazine page with a complex non-rectilinear page layout contains content of MP, HW (in English and Chinese by different writers, horizontal and vertical), PH and BL. Note that some HW is very close to MP. In this image, only the English HW is represented in the training set, but the content of all types is well extracted. Although a box of gray background is incorrectly labeled HW, the foreground MP is correctly preserved.

8.4. DISCUSSION

Combination of both policy changes greatly increases recall, especially on difficult handwriting pixels, with little loss of precision. The masks obtained after policy changes are no longer disjoint; that is, they can share pixels.

Chapter 9

Conclusion

This research began by seeking improvements to Document Image Content Extraction (DICE), that is the location and segmentation of handwriting, machine-print text, photographs, and blank space. In evaluating candidate approaches, we wished to minimize the role of arbitrary manual decisions, and to avoid arbitrary restrictions on region shapes. After a long sequence of experiments, we decided on pixel-accurate post-classification: *i.e.* each training and testing sample would be an individual pixel in a document image. Our post-classifiers take "false color" (the color represents content class of a pixel) images as input and so yields "false color" images as output. This invited us to repeat the post-classification process in series of stages, and led to the design of "repeated classification." However, repeated classification failed after the first stage, due to an implicit, and erroneous, assumption that errors made by classifiers will occur again and again at every stage of post-classification. This failure of repeated classification motivated us to retrain classifiers at each stage: iterated classification, using a sequence of post-classifiers,

each different because each is trained separately on the training-data results of the previous classifier guided, as always, by ground truth.

We have experimented with iterated classifiers on large test set (482M pixels of 157 images). Experimental results show that iterated classifiers can drop the error rate by 24%. Also, iterated classifiers increase *both* recall and precision on all three principal content types, stage by stage. Two policy changes, a "multi-stage voting" and a "blank truthing" policy, improve the performance of document image content extraction on both recall and precision. This new policy yields realistic and useful results, intuitively correct, causing no ambiguity or confusion to downstream processing stages. The combination of both policy changes inevitably increases recall, especially on difficult handwriting cases, but at the same time with little loss of precision. The masks obtained from the policy changes are no longer disjoint: they share pixels. We have carried out a formal analysis of special (and somewhat artificial) cases suggesting why iterated-classification boundaries tend to converge to the ground truth.

In one of our early experiments, we found that iterated classifiers failed catastrophically at the ninth stage. Analysis of this instability suggested that a small cluster of failures, amplified by ground truth, had resulted in the failure. Thus we looked carefully into a variety of methodological issues, such as the best choice among competing ground-truth policies. Our original ground-truth policy was designed to drive the development of the classifier and features, and was chosen to minimize manual effort: since then, we have investigated potentially higher-effort policies which are "tight" and more accurate, and we have seen that they reduce classification errors.

Pixel-accurate segmentation has, very recently, attracted more attention from other researchers. PARC's Saund et al developed a tool for pixel-accurate ground-truthing; we have compared performance among three ground-truth policies: loose, tight and PARC's pixel-accurate, along with morphological expansions on pixel-accurate truth. Our experiments suggest that pixel-accurate ground truth can be captured for high-contrast document images as easily as loose ground truth, and can improve overall accuracy. We have also compared accuracies when classifiers for this problem are trained on one ground-truth policy and then evaluated using a different policy. This indicates that for each test policy, the highest accuracy is achieved by using the *same* policy for training. Our experience also suggests that for each test policy, the more similar of the training policy to the test, the higher the resulting accuracy.

Our iterated classification is designed to be trainable and data-driven. Ground truth and feature set decide the behavior of iterated classification. It requires little change to the scheme except perhaps for some more features. Therefore, it may be well adapted to other applications that could have different specification on ground truth. One of the limitations of iterated classification is its sensitivity to ground truth, that is, a feature set that works for one ground truth policy might not work for another. This sensitivity can be tracked through computing the overall error rate of training set. We also suggested a systematic method for selecting the right size of feature extraction window to fit the ground truth. In terms of complexity, iterated classification may be more time consuming than other methods. Collaborators in our lab have proposed approximations that speed up the classification process. This run time can be further reduced by classifying nxn "blocks" instead of individual

pixels. The input and output of iterated classification are in the form of images. This suggests that iterated classification can be applied as an image boosting technique.

Bibliography

[AB07] A. Antonacopoulos and D. Bridson. Performance analysis framework for layout analysis methods. In *Proc., IAPR 9th Int'l Conf. on Document Analysis and Recognition (ICDAR07)*, Curitiba, Brazil, September 2007.

[AB08] Chang An and Henry S. Baird. Convergence of iterated classification. In *Proc., 8th International Workshop on Document Analysis Systems*, Nara, Japan, September 2008. International Association of Pattern Recognition.

[ACL10] D. Karatzas A. Clavelli and J. Llados. A framework for the assessment of text extraction algorithms on complex colour images. In *Proc., 9th International Workshop on Document Analysis Systems*, Boston, USA, June 2010. International Association of Pattern Recognition.

[AGB07] A. Antonacopoulos, B. Gatos, and D. Bridson. Icdar2007 page segmentation competition. In *Proc., IAPR 9th Int'l Conf. on Document Analysis and Recognition (ICDAR07)*, Curitiba, Brazil, September 2007.

[AK98] Ethem Alpaydin and Cenk Kaynak. Cascading classifiers. *Kybernetika*, 34:369–374, 1998.

[AKB06] A. Antonacopoulos, D. Karatzas, and D. Bridson. Ground truth for layout analysis performance evaluation. In *Proceedings., 7th IAPR Document Analysis Workshop (DAS'06)*, Nelson, New Zealand, February 2006.

[APBP09] A. Antonacopoulos, S. Pletschacher, D. Bridson, and C. Papadopoulos. ICDAR2009 page segmentation competition. In *Proc., IAPR 10th Int'l Conf. on Document Analysis and Recognition (ICDAR09)*, Barcelona, Spain, July 2009.

[BD77] Peter J. Bickel and Kjell A. Doksum. *Mathematical Statistics*. Holden-Day, San Francisco, 1977.

[BM] C. Blake and P. M. Murphy. UCI repository of machine learning databases. http://www.ics.uci.edu/ mlearn/ MLRepository.html.

[BMA07] H. S. Baird, M. A. Moll, and Chang An. Document image content inventories. In *Proc., SPIE/IS&T Document Recognition & Retrieval XIV Conf.*, San Jose, CA, January 2007.

[BMN+06] H. S. Baird, M. A. Moll, J. Nonnemaker, M. R. Casey, and D. L. Delorenzo. Versatile document image content extraction. In *Proc., SPIE/IS&T Document Recognition & Retrieval XIII Conf.*, San Jose, CA, January 2006.

[Bre02] T. M. Breuel. Two algorithms for geometric layout analysis. In *Proc., IAPR 5th International Workshop on Document Analysis Systems*, Princeton, USA, 2002.

[CAB10] Dawei Yin Chang An and Henry S. Baird. Document segmentation using pixel-accurate ground truth. In *Proc., IAPR Int'l Conf. on Pattern Recognition (ICPR'10)*, 2010.

[Cas06] Matthew R. Casey. *Fast Approximate Nearest Neighbors*. Computer Science & Engineering Dept, Lehigh University, Bethlehem, Pennsylvania, May 2006. M.S. Thesis; PDF available at `www.cse.lehigh.edu/~baird/students.html`.

[CB06] Matthew R. Casey and Henry S. Baird. Towards versatile document analysis systems. In *Proceedings., 7th IAPR Document Analysis Workshop (DAS'06)*, Nelson, New Zealand, February 2006.

[Dou92a] Edward Dougherty. *Mathematical Morphology in Image Processing*. CRC Press, New York, 1992.

[Dou92b] Edward R. Dougherty, editor. *Mathematical Morphology in Image Processing*. Marcel Dekker, New York, 1992.

[EDC97] K. Etemad, D. Doermann, and R. Chellappa. Multiscale segmentation of unstructured document pages using soft decision integration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(1):92 – 96, jan 1997.

[GDPP05] B. Gatos, D. Danatsas, I. Paratikakis, and S.J. Perantonis. Automatic table detection in document images. In *Proc., ICAPR2005*, Bath, UK, 2005.

[GNP09] B. Gatos, K. Ntirogiannis, and I. Pratikakis. ICDAR 2009 document image binarization contest (DIBCO 2009). In *Proc., IAPR 10th Int'l Conf. on Document Analysis and Recognition (ICDAR09)*, Barcelona, Spain, July 2009.

[HP74] S. Horowitz and T. Pavlidis. Picture segmentation by a directed split-and-merge procedure. In *Proc., Second Int'l Joint Conf. Pattern Recognition*, pages 424–433, 1974.

[HP76] S. Horowitz and T. Pavlidis. Picture segmentation by a tree traversal algorithm. *J. ACM*, 23(2):368–388, April 1976.

[HPC10] High performance computing at lehigh, 2010. `http://www.lehigh.edu/computing/hpc/index.html`.

[Ish01] Yasuto Ishitani. Model-based information extraction method tolerant of ocr errors for document images. *icdar*, 00:0908, 2001.

[Jai89] Anil K. Jain. *Fundamentals of Digital Image Processing*. Prentice Hall, Englewood Cliffs, New Jersey, 1989.

[JLP01] A. McCallum J. Lafferty and F. Pereira. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *In International Conference on Machine Learning*, 2001.

[JY98] A.K. Jain and B. Yu. Document representation and its application to page decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI–20(3):294–308, 1998.

[KA00] C. Kaynak and E. Alpaydin. Multistage cascading of multiple classifiers: One man's noise is another man's data. In *Proc., 17th Int. Conf. Machine Learning*, pages 455–462, 2000.

[KA07] D. Karatzas and A. Antonacopoulos. Colour text segmentation in web images based on human perception. *Image and Vision Computing*, 25(5):564–577, 2007.

[KC94] G. Kopec and P. Chou. Document image decoding using Markov source models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI–16:602–617, June 1994.

[KGK+07] Sunil Kumar, Rajat Gupta, Nitin Khanna, Santanu Chaudhury, and Shiv Duff Joshi. Text extraction and document image segmentation using matched wavelets and MRF model. *IEEE Transaction on Image Processing*, IP-16:2117–2128, January 2007.

[KJK03] Kwang In Kim, Keechul Jung, and Jin Hyung Kim. Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI–25(12):1631–1639, June 2003.

[KZ04] V. Kolmogorov and R. Zabih. What energy function can be minimized via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI–26:147–159, Feb 2004.

[LL10] Bart Lamiroy and Daniel P. Lopresti. A platform for storing, visualizing, and interpreting collections of noisy documents. In *Proc., Fourth Workshop on Analytics for Noisy Unstructured Text Data (AND'10)*, 2010.

[LPH97] J. Liang, I. Phillips, and R. Haralick. Performance evaluation of document layout analysis algorithms on the uw data set. In *Proc., 4th SPIE Document Recognition*, 1997.

[LPS+05] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, R. Young, K. Ashida, H. Nagai, M. Okamoto, H. Yamamoto, H. Miyao, J. Zhu, W. Ou, C. Wolf, J. M Jolin, L. Todoran, M. Worring, and X. Lin. ICDAR 2003 robust reading competitions: entries, results, and future directions. *International Journal on Document Analysis and Recognition*, 7(2–3):105–122, 2005.

[LZ00] D. Lopresti and J. Zhou. Locating and reconizing text in www images. *Information Retrieval*, 2:177–206, 2000.

[MBA08] Michael A. Moll, Henry S. Baird, and Chang An. Truthing for pixel-accurate segmentation. In *Proc., 8th IAPR International Workshop on Document Analysis Systems*, Nara, Japan, September 2008.

[MC00] M. Mitra and B. B. Chaudhuri. Information retrieval from documents: A survey. *Information Retrieval*, 2(2–3):141–163, 2000.

[NGP08] K. Nitrogiannis, B. Gatos, and I. Pratikakis. An objective evaluation methodology for document image binarization techniques. In *Proceedings., 8th IAPR Document Analysis Workshop (DAS'08)*, Nara, Japan, September 2008.

[PCHH93] I.T. Philips, S. Chen, J. Ha, and R.M. Haralick. English document database design and implementation methodology. In *Proceeding of the 2nd Annual Symposium on Document Analysis and Retrieval*, pages 65–104, UNLV, USA, 1993.

[PGM+04] S. J. Perantonis, B. Gatos, V. Maragos, V. Karkaletsis, and G. Petasis. Text area indentification in web images. In *Proc., 3rd Hellenic Conference on AI, Methods and Application of Artificial Intelligence*, pages 82–92, Berlin, Germany, May 2004. Springer Berlin. Lecture Notes in Computer Science LNCS 3025.

[PL96] Kyeong-Ryeol Park and Chung-Nim Lee. Scale-space using mathematical morphology. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, PAMI-18(11):1121–1126, November 1996.

[PT94] James E. Peak and Paul M. Tag. Segmentation of satellite imagery using hierarchical thresholding and neural networks. *Journal of Applied Meteorology*, 33:605–616, May 1994.

[RM07] T. Retornaz and B. Marcotegui. Scene text localization based on the ultimate opening. In *Proc., 8th International Symposium on Mathematical Morphology*, Rio de Janeiro, Brazil, Octorber 2007.

[Ser82] J. Serra. *Images Analysis and Mathematical Morphology*. Academic Press, New York, 1982.

[SKB06] F. Shafait, D. Keysers, and T. M. Breuel. Pixel-accurate representation and evaluation of page segmentation in document images. In *Proc., IAPR 18th Int'l Conf. on Pattern Recognition (ICPR20006)*, Hong Kong, China, August 2006.

[SLS09] Eric Saund, Jing Lin, and Prateek Sarkar. Pixlabeler: User interface for pixel-level labeling of elements in document images. In *Proc., IAPR 10th Int'l Conf. on Document Analysis and Recognition (ICDAR09)*, Barcelona, Spain, July 2009.

[Smi09] Ray Smith. Hybrid page layout analysis via tab-stop detection. In *Proc., IAPR 10th Int'l Conf. on Document Analysis and Recognition (ICDAR09)*, Barcelona, Spain, July 2009.

[Smi10] Elisa H. Barney Smith. An analysis of binarization ground truthing. In *Proc., 9th IAPR International Workshop on Document Analysis Systems*, Boston, USA, June 2010.

[SNH07] T. Paquet S. Nicolas, J. Dardenne and L. Heutte. Document image segmentation using a 2d conditional random field model. In *Proc., Int'l*

Conf. on Document Analysis and Recognition (ICDAR2007), pages 407–411, September 2007.

[Soi08] Pierre Soille. Constrained connectivity for hierarchical image partitioning and simplification. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, PAMI-30(7):1132–1145, July 2008.

[SS94] J. Serra and J. Soille. *Mathematical Morphology and its Applications to Image Processing*. Kluwer Academic Publishers, Dordrecht, 1994.

[SS01] L. G. Shapiro and G. C. Stockman, editors. *Computer Vision*. Prentice-Hall, New Jersey, 2001. Lecture Notes in Computer Science LNCS 3517.

[SS03] S.J. Simske and M. Sturgill. A ground-truthing engine for proofsetting, publishing, re-purposing and quality assurance. In *Proceedings of the 2003 ACM Symposium on Document Engineering (Doc Eng'03)*, pages 150–152, Grenoble, France, 2003.

[Wal04] Hanna M. Wallach. Conditional random fields: An introduction. Technical report, University of Pennsylvania, 2004.

[WB08] Sui-Yu Wang and Henry S. Baird. Feature selection focused within error clusters. In *ICPR*, pages 1–4, Tampa, FL, Dec 8-11 2008.

[WJ06] C. Wolf and J. M. Jolion. Object count/area graphs for the evaluation of object detection and segmentation algorithms. *International Journal of Document Analysis*, 8(4):105–122, 2006.

[WMB99] Ian H. Witten, Alistair Moffat, and Timothy C. Bell. *Managing Gigagytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann, 1999.

[YAB10] Dawei Yin, Chang An, and Henry S. Baird. Imbalance and concentration in k-nn classification. In *ICPR*, pages 2170–2173, 2010.

[YBA10] Dawei Yin, Henry S. Baird, and Chang An. Time and space optimization of document content classifiers. In *DRR*, pages 1–10, 2010.

[YZ04] O. Veksler Y.Boykov and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI–23:1222–1239, November 2004.

[ZLDP01] Y. Zheng, C. Liu, X. Ding, and S. Pan. Form frame line detection with directional single-connected chain. In *Proc., IAPR 6th Int'l Conf. on Document Analysis and Recognition (ICDAR2001)*, Seattle, USA, 2001.

Vita

Chang An was born in Suining, China on June 13, 1981. He graduated from University of Electronic Science and Technology of China in June 2004, with B.S. in Computer Science. He attained his M.S. from Lehigh University in May 2007. Chang has authored and presented papers at IAPR ICDAR 2007, IAPR DAS 2008, IAPR ICPR 2010 and SPIE/IS&T DR&R 2011.