A Theory of Multiple Classifier Systems And Its Application to Visual Word Recognition

Tin Kam Ho

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

May, 1992

© Copyright by Tin Kam Ho, 1992
All Rights Reserved

To Hing Kai, my husband, for his love and support.
To my parents, and Siu Lui, my sister, for their encouragement.
To Tin Hung and Yu Hai, my sister and brother, for their friendship.

In memory of Wai Man Middle School, Hong Kong, 1973-1979.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Professor Sargur N. Srihari, my advisor and chairman of my dissertation committee. He introduced me to the area of pattern recognition and led me to its challenging frontiers. It is his guidance and support that has made this dissertation possible.

I am also grateful to Dr. Jonathan J. Hull, co-chairman of my dissertation committee. His generous sharing with me his professional experience, and his enthusiastic involvement along the course of this research, helped shape much of its development.

I would like to thank Dr. David Sher, member of my dissertation committee, for his challenging questions which made the theoretical part of this dissertation more complete.

I am also indebted to Dr. Henry Baird of AT&T Bell Laboratories. His keen interest in the results of my research has strengthened my motivation to attain the best. As an external reader of this thesis, he has made valuable suggestions to the manuscript.

I enjoyed discussions with Dr. Saskia le Cessie and Dr. Tai-Shing Lau. Their comments and references have brought statistical highlights to this dissertation.

During an early stage of this research, Professor Jean-Claude Simon visited Buffalo. Our discussion reinforced my belief in the potentials of the multiple classifier methodology. Professor Ching Y. Suen's interest in this topic and our meetings at several conferences have also been very encouraging.

This research has been funded by the Office of Advanced Technology (OAT) of the United States Postal Service.* I am indebted to the people who have had the vision to support the fundamental research needed for a future mission on postal automation in the United States. The project provides a real-life problem of such a large scale that it challenges many theoretical and experimental researchers. I am very fortunate to have had the opportunity to be involved in the effort.

Special thanks to Postmaster General Anthony Frank, Senior Associate Postmaster General Kenneth Hunter, and Assistant Postmaster General Karen Uemoto. Their visits gave me strong support.

Special thanks also to Gary Herring, Director of OAT, who has provided strong support to this project from its very beginning.

Thanks also to Carl O'Connor of the USPS, Dr. Gerardo Garcia and Gilles Houle of Arthur D. Little, Inc., who listened to many of my presentations and offered useful suggestions.

My career as a graduate student was enriched intellectually by the advanced courses offered by Professor Stuart Shapiro, Dr. Xin He, Dr. Deborah Walters, Professor Patricia Eberlein, and Dr. Russ Miller. Their lectures have widened my vision in knowledge representation, analysis of algorithms, computational vision as well as parallel computing. Knowledge in these areas has been an essential support to this research and will play a role in its future development.

I appreciate very much the kindness of Dr. Jeffery Zucker, who taught me theoretical computer science and guided my husband's dissertation. His advice and support to both of us were extremely valuable.

Thanks to the members of the CEDAR research group (Center of Excellence for Document Analysis and Recognition) who contributed to a friendly and open research environment. Special thanks to Paul Palumbo, Edward Cohen, Stephen Lam, Peter Cullen, Scott Cohen, Steve Tylock, Dar-Shyang Lee, Liang Li, Yan Li, Mikhail Prussak, Piotr Prussak, Ralph Ames and John Giattino. Special thanks to Evelyn Kleinberg, who kindly helped proofread the manuscript.

Thanks also to Deborah Squire, Heike Berlind, and Eugenia Smith, secretaries of CEDAR, and Eloise Benzel, Gloria Koontz, and Sally Elder of the Department of Computer Science, for every help they have kindly provided.

* under Task Order No. 104230-88D-2576 and Contract No. 104230-84D-0962.

Abstract

Despite the success of many pattern recognition systems in constrained domains, problems that involve noisy input and many classes remain difficult. A promising direction is to use several classifiers simultaneously, such that they can complement each other in correctness. This thesis is concerned with decision combination in a multiple classifier system that is critical to its success.

A multiple classifier system consists of a set of classifiers and a decision combination function. It is a preferred solution to a complex recognition problem because it allows simultaneous use of feature descriptors of many types, corresponding measures of similarity, and many classification procedures. It also allows dynamic selection, so that classifiers adapted to inputs of a particular type may be applied only when those inputs are encountered.

Decisions by the classifiers are represented as rankings of the class set that are derivable from the results of feature matching. Rank scores contain more information than unique class choices, and can be given on a common and consistent scale by each classifier for different instances of the problem.

The rankings are combined by methods that either reduce or rerank a given set of classes. An intersection method and a union method are proposed for class set reduction. Three methods based on the highest rank, the Borda count, and logistic regression are proposed for class set reordering. The advantages and disadvantages of each method are discussed.

The theory is tested in an application to visual word recognition, where many classes need to be discriminated. A computational model is proposed that integrates three approaches to word recognition. These include a character recognition approach, a segmentation-based approach, and a word shape analysis approach. A control strategy selectively activates the suitable classifiers and candidate words in a given lexicon. The class decisions are combined by the methods given in our theory.

The success of the model is demonstrated in an experiment with noisy word images of multiple font types. The results show that the integrated model outperforms each individual approach. Similar results are observed in two other experiments in character recognition.

Contents

1 Introduction **1**

1.1 Definitions 1

1.2 Motivations 3

1.3 Research Goals 3

1.3.1 Representation of Class Decisions 4

1.3.2 Decision Combination Methods 4

1.3.3 Elimination of Redundant Classifiers 5

1.3.4 Dynamic Classifier Selection and Multi-Stage Organization . . 5

1.4 The Visual Word Recognition Problem 6

1.5 Outline of the Dissertation 8

2 Literature Review **9**

2.1 Multiple Classifier Systems 9

2.1.1 Multiple Approaches to Pattern Recognition 9

2.1.2 Similar Ideas in Related Areas 13

2.1.3 Decision Combination in Other Contexts 14

2.2 Word Recognition Methodologies 15

2.2.1 Character-Based Methods 15

2.2.2 Segmentation-Based Methods 25

2.2.3 Word Shape Analysis Approaches 26

2.3 Conclusion: Rationale for a New Approach 27

3 A Theory of Multiple Classifier Systems **31**

3.1 Advantages of Using Rankings 32

3.1.1 Using Rankings Instead of Unique Class Choices 32

3.1.2 Using Rankings Instead of Numerical Similarity Scores 33

3.1.3 Summary of Advantages of Using Rankings 38

3.2 Objectives of Decision Combination 39

3.3 Methods for Class Set Reduction 41

3.3.1 The Intersection Approach 41

3.3.2 The Union Approach 44

3.3.3 Remarks 48

3.3.4 Approximation Methods 49

3.4 Methods for Class Set Reordering 51

3.4.1 The Highest Rank Method 51

3.4.2 The Borda Count Method 53

3.4.3 The Logistic Regression Method 56

3.4.4 Summary of Class Set Reordering Methods 68

3.5 Dynamic Selection of Classifiers 71

3.6 Multiple Stage Organization 73

3.7 Alternative Approaches 74

4 A Computational Model for Visual Word Recognition **75**

4.1 The Word Recognition Problem 75

4.2 Constraints on the Problem Domain 76

4.3 Limitations of Traditional Approaches 79

4.4 Integration of Three Approaches 80

4.5 A Computational Model for Word Recognition 81

4.6 Character Recognition and Contextual Postprocessing 83

4.6.1 Character Segmentation 86

4.6.2 A Modified K-Nearest-Neighbor Classifier 90
4.6.3 Character Recognition by Multiple Algorithms 91
4.6.4 Character Decision Postprocessing 92
4.7 Segmentation-Based Word Recognition 96
4.8 Word Shape Analysis 99
4.8.1 Word Image Area Partitioning 100
4.8.2 Word Shape Feature Extraction 102
4.8.3 Prototype Matching and Classification 103
4.8.4 Alternative Feature Sets 104
4.9 The Control Strategy 110
4.9.1 Lexical Activation by Global Features 111
4.9.2 Dynamic Selection of Classifiers 111
4.10 Decision Combination 112
4.11 Remarks 114

5 An Experimental Word Recognition Algorithm 115
5.1 Algorithm Descriptions 115
5.2 Implementation Considerations 116
5.2.1 Word Image Preprocessing 116
5.2.2 The Five Parallel Classifiers 118
5.2.3 Selection of Font Samples for Word Prototype Synthesis . . . 119
5.2.4 Prototype Clustering in Word Shape Analysis 122
5.2.5 Speed Considerations for the Segmentation-Based Method . . 123
5.3 Experimental Results 124
5.3.1 Background of the Application 124
5.3.2 The Image Database 125
5.3.3 System Training 125
5.3.4 Test Results 126

5.3.5 Observations 127
5.4 Dynamic Selection of Classifier Combinations 127

6 Applications to Other Domains **131**
6.1 Example Application to Handwritten Digit Recognition 132
6.2 Example Application to Machine-Printed Character Recognition . . . 133

7 Conclusions **140**
7.1 Summary 140
7.2 Future Directions 141
7.2.1 Decision Combination 141
7.2.2 Word Recognition 142
7.2.3 Creating A Multiple Classifier System 144

References **146**

Appendix **154**

List of Figures

1.1 The visual word recognition problem 6
1.2 The fragile dividing line between legibility and illegibility. 7
1.3 Examples of image degradation and font style variations. 8

2.1 Summary of word recognition methods. 16
2.2 A combination of pixel values as a feature. 19
2.3 Different descriptors of an image. 29

3.1 Distances and rank scores for 50 inputs and 5 classes. 35
3.2 Distances and rank scores for enlarged class sets. 36
3.3 The intersection approach to class set reduction. 42
3.4 The union approach to class set reduction. 45
3.5 Example regions determined by the intersection and the union methods. 49
3.6 Class set reordering by the highest rank method. 52
3.7 Class set reordering by the Borda count method. 54
3.8 Logistic functions. 58
3.9 Class set reordering by the logistic regression method. 60
3.10 Plot of empirical logits versus ranks by two classifiers. 63
3.11 Plot of fitted logits versus ranks by two classifiers. 63
3.12 Residual plots. 64
3.13 Plot of empirical logits at all ranks decided by two classifiers. 66
3.14 Plot of empirical logits at top 10 ranks by two classifiers. 68
3.15 Plot of linearly fitted logits at top 10 ranks by two classifiers. 69

3.16 Plot of piecewise fitted logits at top 10 ranks by two classifiers. 69

4.1 Example characters in the American Bankers Association E-13B font. 77
4.2 Example characters in decorative fonts. 78
4.3 Examples of problems in character segmentation. 80
4.4 A computational model for word recognition using multiple approaches. 82
4.5 The character-based approach. 84
4.6 Word images better recognized by character-based methods. 85
4.7 Irrecoverable errors in character recognition. 86
4.8 A problem in using aspect ratios. 87
4.9 Two dot matrix images before and after smearing. 88
4.10 Vertical projection profiles of characters in serif and sans serif fonts. . 89
4.11 Vertical projection profiles of an ideal image and a faint image. 89
4.12 Similar shapes appearing in different contexts. 90
4.13 A modified k-nearest-neighbor classifier. 91
4.14 An example image and top choices by 6 classifiers. 95
4.15 The segmentation-based word recognition approach. 97
4.16 A word image and the extracted characters. 98
4.17 Word images better recognized by segmentation-based methods . . . 99
4.18 The word shape analysis approach. 100
4.19 Word images better recognized by word shape analysis. 100
4.20 An image before and after white gap packing. 102
4.21 Smearing and reference line location. 103
4.22 An image and its 40 area partitions. 104
4.23 An example of stroke direction distribution. 105
4.24 Example of edge detection and selection. 106
4.25 A skeleton image and the end points detected by convolution. 107
4.26 Examples of structural features and their descriptors. 109

5.1 Example images of different density levels and results after enhancement.117
5.2 Results of font matching using 1,055 training images. 121

List of Tables

3.1 Comparison of effects of rank thresholds and distance thresholds. . . 37

3.2 An example of determining neighborhood sizes for an intersection. . . 43

3.3 An example of determining neighborhood sizes for a union. 46

3.4 Example distribution of rank scores and the empirical logits. 62

3.5 Analysis of maximum likelihood estimates in logistic regression. . . . 65

5.1 Definition of density levels and corresponding enhancement procedures. 117

5.2 Summary of methods used in the parallel classifiers 118

5.3 Summary of performance on 1,671 test images using a 33,850 word lexicon 126

5.4 Definitions of agreement classes and estimated model parameters . . . 129

5.5 Comparison of results by static and dynamic combination 129

5.6 Breakdown of results by agreement class 130

6.1 Analysis of maximum likelihood estimates in logistic regression. . . . 133

6.2 Performance of digit classifiers and their combination on test set. . . 133

6.3 Six classifiers for character recognition. 135

6.4 Analysis of maximum likelihood estimates in logistic regression. . . . 137

6.5 Performance of character classifiers and their combinations on test set. 138

Chapter 1

Introduction

Despite the fact that many satisfactory results have been reported for constrained domains, recognition problems involving noisy patterns and many classes remain difficult. Recently, a promising direction was suggested that a number of classifiers can be used in parallel to tackle the difficulties. This has the advantage that features and classification procedures of different types can be used simultaneously to complement one another, and improve the overall correctness of recognition.

A critical need in this approach is a decision combination method that can take advantages of the strengths of the individual classifiers and avoid their weaknesses. This thesis proposes and analyzes several combination methods, and discusses reasons that justify the use of a multiple classifier system.

In the second half of the thesis, the theory of multiple classifier systems is applied to integrate several word recognition approaches. The objective is to obtain a robust algorithm that recognizes noisy word images of multiple font types.

1.1 Definitions

A *classifier* is a function that takes in a descriptor of a pattern, and decides on the pattern's membership in a given set of classes. In recognition problems, the decision is considered to be correct if it is the same as one that would be made by a human being.

A descriptor is derived from a pattern by a *feature extraction* procedure. *Features* are symbolic or numeric entities that describe a pattern. An example feature in a visual pattern is an edge of a particular location and orientation. An example feature in an aural pattern is the number of phonemes. Features are defined by the detection operators [126], and represented by descriptors of many kinds. Example descriptors include multi-dimensional vectors and symbol strings.

A class is often exemplified by a set of prototypes, called the *training patterns*. Feature descriptors are derived from the training patterns to represent a class. Feature descriptors representing a class may be used to derive a model for that class. The process of deriving a model or its parameters from the class prototypes is referred to as *training* a classifier.

Feature matching is a process wherein the descriptor of the input pattern is matched to the descriptors of the classes. This is done either by matching the input descriptor directly to the prototype descriptors, or by matching it to the class models derived from the prototype descriptors. An example of a class model is a linear discriminant function. Components of a feature vector are multiplied by a set of coefficients defined in the discriminant for a particular class. The discriminant maps the feature values to a score. Such a score is a typical result of feature matching. It measures the similarity between a pattern and a class. Classification decisions are made according to these scores.

A *multiple classifier system* consists of a set of classifiers and a decision combination function. Each classifier uses a particular descriptor of an input pattern, and decides on the membership of the pattern to a given set of classes. Decisions by the individual classifiers are combined to derive a final decision, which is the output of the system.

1.2 Motivations

The use of multiple classifier systems is motivated by the existence of many alternative solutions to a pattern recognition problem, and the observation that these solutions often complement one another in correctness. The diversity of solutions is well-known for all stages in recognition, including feature extraction, feature matching and classification. For instance, in character recognition, numerous feature extraction and matching procedures have been proposed, and high performance is claimed for each of them on specific data sets [87] [93].

A fair evaluation of these alternative solutions is interesting. A more interesting question is whether it is possible to integrate these alternative solutions, in such a way that the integration excels the individuals in performance. In several preliminary studies [44] [68] [86] [135], there are hints that robust solutions to certain recognition problems may involve a number of independent methods. These studies suggest the idea of a multiple classifier system.

Conflicts arise when two or more classifiers make different decisions. How to resolve these conflicts is a central problem for the construction of a multiple classifier system. A theory that outlines methods for decision combination is critical to further development of the multiple classifier methodology. This thesis attempts to provide such a theory.

1.3 Research Goals

Our primary goal in this research is to obtain an effective combination method for a multiple classifier system that takes advantages of the individual strengths of the classifiers.

Several issues need to be addressed to solve this problem: (i) to find a representation of the class decisions that is most effective for combination; (ii) to look for

possible decision combination methods and study their advantages and disadvantages; (iii) to obtain a method for determining which classifiers make significant contribution to the correctness of the combined decisions; and (iv) to explore other possible modes of cooperation by multiple classifiers besides a parallel activation. Our results on these topics are given in Chapter 3, and outlined as follows.

1.3.1 Representation of Class Decisions

Decisions given by the classifiers need to be represented at an effective level of abstraction. Possible representations of a classifier's decision include a unique class choice, a ranking of all the given classes in an order of similarity to the given pattern, and a numerical score assigned to each given class that reflects the strength of the classifier's belief in its being the true class.

We suggest that the class decisions be represented as rankings of a given class set. A ranking of a class set is better than the numerical scores in that it can be easily derived from the results of feature matching, and requires no additional knowledge or assumptions on the classifier's past performance. It is better than the unique class choices in that it contains information about the classifier's subsequent preferences from which some near misses could be recovered. We therefore focus our study on rank-based combination methods.

1.3.2 Decision Combination Methods

The objective of decision combination can be to reduce or reorder the class set. A pattern is recognized if the class set is reduced to a unique member, or if the class set is ordered such that the correct class appears at the top. These two objectives can be achieved by different methods.

Two methods are proposed for class set reduction: the intersection method and the union method. Three methods are proposed for class set reordering: by orders

of the *highest rank*, the *Borda count*, and the *estimated logit*. Chapter 3 presents the details of each method, as well as their advantages and disadvantages.

1.3.3 Elimination of Redundant Classifiers

The effectiveness of the classifiers in a combination is often not uniform. Some classifiers may be redundant in a combination and can be eliminated for efficiency. A simple case is that, if two classifiers always produce identical class decision for every pattern, then one of them is redundant. In other cases, the significance of each classifier depends on how its decisions are to be combined with others. We will discuss the conditions under which a classifier is redundant in the context of each combination method.

1.3.4 Dynamic Classifier Selection and Multi-Stage Organization

Two or more classifiers can be applied in parallel. They can also cooperate in other ways. In particular, we are interested in the possibilities of *dynamic classifier selection* and a *multi-stage organization*.

Each classifier in a collection may work better for certain types of input than for others. If it can be detected which type of input is encountered, only the classifier or classifiers that are the best for that input need to be activated. This is the idea of dynamic classifier selection. We will discuss how a dynamic selector can be constructed and give an example.

A collection of classifiers can be organized in serial or in parallel. Multiple stage organizations are also possible. It is a challenging open problem to determine the best organization. The possibility of dynamic selection introduces even more complexity. Instead of attempting an analytic method for evaluating all possible classifier organizations, which is too complicated to be included in the scope of this research, we suggest some intuitive guidelines and resort to an experimental approach.

1.4 The Visual Word Recognition Problem

Our theory of multiple classifier systems is tested in an application to the visual word recognition problem. In this context, a pattern is a word image. The set of classes is a lexicon containing the word in the image. Ideally, a ranked list of words from the lexicon is derived such that the word in the image is ranked as close to the top as possible (Figure 1.1). The ranking can then be used in further contextual analysis.

Word recognition has been intensively studied for several decades. The problem is attractive because it is well-defined, data for experiments are easy to obtain, and it has high practical value in real-life applications. Areas of practical appli-

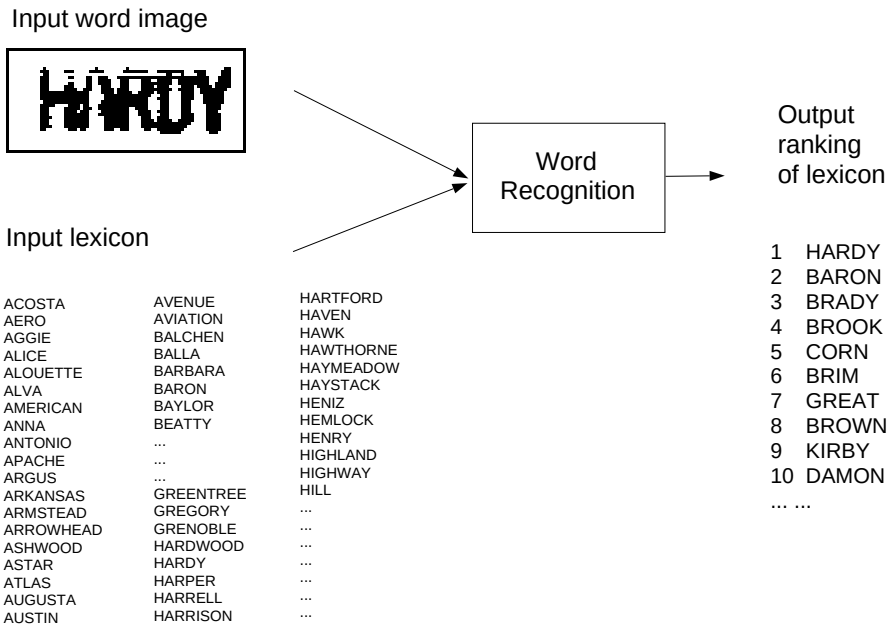


Figure 1.1: The visual word recognition problem. Given a digitized image of a word and a lexicon containing the word, the objective is to order the lexicon such that the word in the image is ranked as close to the top as possible.

cations include providing reading aids to the blind [77], and quick reading of large volumes such as in mail sorting. In addition, solutions to the problems encountered in studying some marginally legible images (Figure 1.2) may provide insights to our understanding of the pattern recognition abilities of human beings.

Many algorithmic solutions have been proposed for visual word recognition. However, none is robust enough for application in a domain involving word images of multiple font types and highly variable quality, like those shown in Figure 1.3.

A multiple classifier system is considered a good solution to this problem. In this research, a computational model for word recognition is proposed based on our theory. We demonstrate its success by experiments with real-world data. Our results show that the model outperforms each of the individual classifiers.

The utility of a multiple classifier system is by no means limited to visual text recognition. It is applicable to any pattern recognition problem that involves a large set of classes and many alternative classifiers.

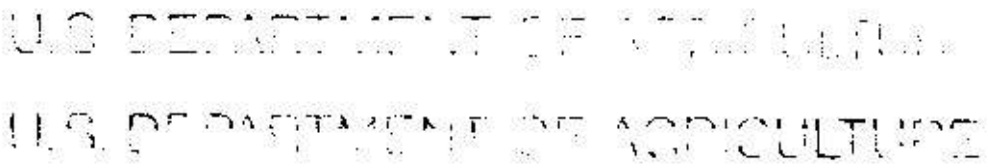


Figure 1.2: The fragile dividing line between legibility and illegibility. Words in the lower row, though appearing faint, can be recognized with a little effort. Recognition of the words in the upper row is much harder, if not impossible. A good guess for words in the upper row can be made by comparing corresponding pieces in two rows. Why is it so? How much degradation in an image can our recognition mechanism tolerate? Solutions to problems encountered in constructing a word recognition algorithm help in our understanding of human recognition abilities.

1.5 Outline of the Dissertation

In Chapter 2, we review some prior attempts at constructing multiple classifier systems, suggested methods for decision combination, and previously proposed solutions to visual word recognition. A theory of multiple classifier systems is presented in Chapter 3. In Chapter 4, discussions are focused on the problems in visual word recognition and their solutions. A computational model is proposed. In Chapter 5, we describe an implementation of the model. Experimental results using real-world data are reported. In Chapter 6, two applications of the theory to other domains and experimental results are discussed. The dissertation is concluded with a summary and discussions of future research directions in Chapter 7.

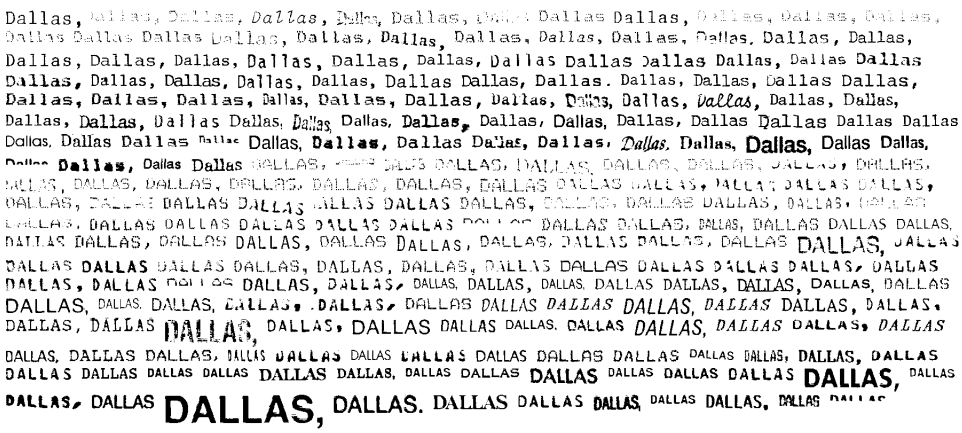


Figure 1.3: Examples of image degradation and font style variations. These images are extracted from images of envelopes captured on a live mail processing system. How can an algorithm be constructed to recognize images of such a diversity of font types and quality? The answer is discussed in this dissertation as an application of the theory of multiple classifier systems.

Chapter 2

Literature Review

The idea of using multiple approaches to a given pattern recognition problem has been suggested in the past. Some preliminary methods for combining classifiers have been proposed. In Section 2.1 we review these prior approaches and their difficulties, which we intend to overcome with our theory. In Section 2.1.3, discussions are focused on methods proposed in other contexts for combining decisions by multiple experts, which are also useful in classifier combination.

In Section 2.2 we review prior approaches to the word recognition problem. Three major methodologies are identified, among which the character recognition approach is the most popular. Section 2.2.1 is devoted to a review of character-based methods for word recognition. Alternative methods for word recognition include the segmentation-based methods reviewed in Section 2.2.2, and the word shape based methods discussed in Section 2.2.3. These three methodologies are integrated in our computational model proposed in Chapter 4.

2.1 Multiple Classifier Systems

2.1.1 Multiple Approaches to Pattern Recognition

The advantages of using multiple methods to solve a pattern recognition problem were realized early, as noted in several previous attempts.

Pattern recognition by a committee of decision makers is discussed in an early

text book by Nilsson [98].

Haralick [53] gives error bounds for combined decisions by multiple Bayesian classifiers. An example is given where the use of multiple classifiers is motivated by the need to use a different resolution in each feature dimension.

Srihari [129] gives a theoretical analysis of the performance of a system that takes a majority vote of n independent decision makers on a two-class decision problem, where n is an odd integer. The correct rate of decisions by each classifier, p, is assumed to be identical for all n classifiers. It is shown that the correct rate of the decisions by this n-classifier system increases monotonically with n for $p > 0.5$, and decreases monotonically for $p < 0.5$.

Fu [44] attempts to use both syntactic and statistical methods to solve a recognition problem. In his approach, a set of pattern primitives are first extracted from an image and represented by symbol strings. In addition, the attributes of a primitive are represented by a parameter vector associated with the primitive. A (dis)similarity score between two descriptors is defined to be the sum of the syntactic distance between the symbol strings, and the semantic distance between the associated parameter vectors. While the idea of using the attributes together with the primitives is appealing, this early coupling of two types of descriptors creates a problem in measuring similarity. The two distances are measured in incompatible scales, and have to be properly normalized before combination. Moreover, the parameter vectors are always associated with the primitives represented by the symbols, therefore there is a lack of independence between information contained in the two types of descriptors, which limits the advantage gained by using both methods.

Mazurov et. al. [88] discuss the application of committee methods to two-class discrimination problems, where the final decision is made by majority vote. They give theorems of existence of a minimal committee, and an algorithm for its construction. This method cannot be easily generalized to a many-class problem.

Mandler and Schuermann [86] address the advantages of using multiple classification algorithms. They describe a method for combining independent classifier decisions based on Bayes decision theory and Dempster-Shafer theory of evidence. Statistical models and the Bayes rule are used to transform the distance measures computed by the classifiers into confidence values associated with each class, which are then combined by the rules of evidence theory. The main difficulty in this method lies in the transformation from the distance measures to confidences, since distances are not often good predictors of classification correctness. This problem will be further discussed with results from an experiment in Section 3.1.2 of Chapter 3.

Kleinberg [73] studies the combination of a large number of automatically generated classifiers for a two-class problem. The combined power of many unintelligent classifiers which are different from one another is analyzed. It is proved that given a recognition problem, the number of classifiers needed to achieve essentially perfect performance is far less than the number of all possible classifiers, making this method computationally feasible. There is no interaction among the individual classifiers prior to their combination, therefore they can be generated and applied independently in parallel. A theory is developed which details how a recognition algorithm can be automatically constructed using a set of training samples [74]. A multi-class discrimination problem is solved by combining the results of pairwise discriminations.

Hull et al. [68] show that multiple algorithms are useful for character recognition, where the results are combined by a decision tree method. The decision tree needs to be manually designed for a specific set of algorithms and hence is limited in generality. The application of the method described in this thesis to the same problem is suggested later in [69].

Suen et al. [135] propose using several experts for handwritten character recognition, where the decisions are combined by majority vote. The voting mechanism takes in the top class choice by each classifier. If a classifier does not discriminate

among a set of candidate classes, its vote is split evenly among them. Nadal et al. in [94] describe several heuristic rules for combining decisions by a similar set of classifiers. In a many-class problem such as character recognition, more information from the classifiers than merely the top choices is useful.

Kurzweil [77] discusses the advantages of using multiple experts to solve a recognition problem. Seeing the difficulty in recognizing noisy characters, he suggests, "one way to deal with the vagaries of real-world patterns is to have redundant experts and multiple ways of describing the same type of pattern." ([77], page 240) An expert manager is proposed that selectively believes decisions from different experts based on confidence values on a continuous scale. This suggests the idea of *dynamic classifier selection*. Its success depends on the effectiveness of the confidence values.

Hansen and Salamon [52] describe a use of ensembles of independently trained neural networks to overcome the problem of local optimality. Several methods are proposed to achieve a collective classification that is cross-validated. Decisions are combined by the majority voting rule and the plurality voting rule. It is unclear as to how this method can be applied to a many-class recognition problem.

Bradford and Nartker [17] studied the correlation of errors made by several optical character readers. A plurality voting method is used to combine their decisions. They report that there is significant decorrelation between those errors, and conclude that a combination of several independent recognition devices should lead to substantial improvement in correctness.

Kimura and Shridhar [72] describe a statistical algorithm and a structural algorithm for handwritten numeral recognition. Several simple schemes are used to combine their decisions. The focus is on reducing the error rate by rejecting the cases when the two algorithms disagree.

The advantage of combination of independent methods has been realized early

in our study. As discussed in [56] [57], a rank-based method is considered useful. The idea is further developed into a series of combination methods which are applied to word recognition in [58]. A regression method to combine decisions by several character recognition algorithms is proposed in [59]. Details of our theory are presented in Chapter 3.

2.1.2 Similar Ideas in Related Areas

There is an interesting parallel of the multiple approach methodology in discussions of commonsense problem solving by artificial intelligence. Multiple agent problem solving is proposed to overcome the difficulty in building a comprehensive expert system to solve generic commonsense problems. According to Minsky, "the versatility that we need can be found only in large-scale architectures that can exploit and manage the advantages of several types of representations at the same time. Then, each can be used to overcome the deficiencies of the others." ([91], page 36)

The ideas that redundancy and diversity in representations and procedures are needed, and that they should be connected at an effective level of abstraction, are of the same philosophy as that of a multiple classifier system.

Motivations and decision combination methods for multiple classification systems are also developed in studies of *sensor fusion* [45] [85]. Sensor fusion is a process to integrate information from various sensors for object detection and identification. Fusion is performed at three levels: the signal level, the feature level, and the decision level.

A multi-sensor system using decision level fusion is the same as a multiple classifier system. In most of the known sensor fusion systems, the decisions are typically given as probabilities or confidence ratings associated with each class. The most popular approaches for multi-sensor decision combination include majority voting, Bayes decision theory, Dempster-Shafer theory of evidence, fuzzy logic and production rules.

One major difficulty in this approach is in deriving an effective confidence measure for decisions given by the sensors.

Sher [118][119] describes a method to construct robust image operators from a set of specialized ones. The likelihood that the observed data fit in the assumed model of each specialized operator is computed. These likelihoods can be used to dynamically select the best fit operator for each input. A robust operator for boundary pixel detection is thus constructed as an example.

2.1.3 Decision Combination in Other Contexts

Many methods for committee decision making are proposed in other contexts. These include voting and consensus ranking that are useful in a multiple classifier system. There are also other decision combination models including Bayes decision theory, Dempster-Shafer theory of evidence, fuzzy logic, and production rules.

Committee Decision Making. *Group consensus functions* or *social welfare functions* are developed to combine rankings given by a committee of decision makers. The majority vote is a particular group consensus function. Arrow [6] and Goodman [46] discuss the conditions on such functions and possibility theorems based on several assumptions. In [13] Black presents the logic of committee decisions and elections, as well as a history of the mathematical theories of committees and elections. A formal analysis of social choice functions, with emphasis on the study of majority vote is given by Fishburn [40]. A theory of distances between partial orders is given by Bogart [14][15]. In a many-class recognition problem, it is often useful to consider more than the top choice by each classifier. Therefore the ranking combination methods are more useful in our context than the majority vote.

Combination of Belief and Confidence Values. Most other theories on decision combination, including Bayes decision theory [104] and Dempster-Shafer theory [117], require that the decisions be given in the form of probability or belief

values associated with each class. To convert numerical similarity scores computed by the individual classifiers into belief values on a normalized scale, heuristic models are often used which suffer from a lack of generality. On the other hand, the method of ranks does not rely on assumptions about the statistical distributions of the similarity scores, and is therefore generally applicable.

2.2 Word Recognition Methodologies

Research in text recognition has long been focused on character recognition. Word recognition is traditionally done by adding preprocessing and postprocessing steps to character recognition. A three-step process is common: character segmentation, character recognition, and contextual postprocessing. A practical system designed for mail sorting [113] is based on this approach. Contextual constraints in a special domain are particularly useful in this approach. An interesting example is given by Baird and Thompson [9], where a system is described that applies this approach to read chess moves and uses the rules of the game as contextual constraints.

Many contextual constraints are present in a given lexicon that specifies the set of allowable words. For instance, if there are no words where the character 'p' is followed by the character 'q', the 'pq' combination can be eliminated from the character decisions. Three major classes of methods have been proposed to use a given lexicon in word recognition. These are referred to as the character-based method, the segmentation-based method, and word shape analysis. Each of these approaches uses the contextual knowledge contained in a lexicon in a different way. Figure 2.1 summarizes some important methods in these three approaches.

2.2.1 Character-Based Methods

Character-based methods consist of three steps: character segmentation, character recognition, and contextual postprocessing. Each of these steps has been intensively

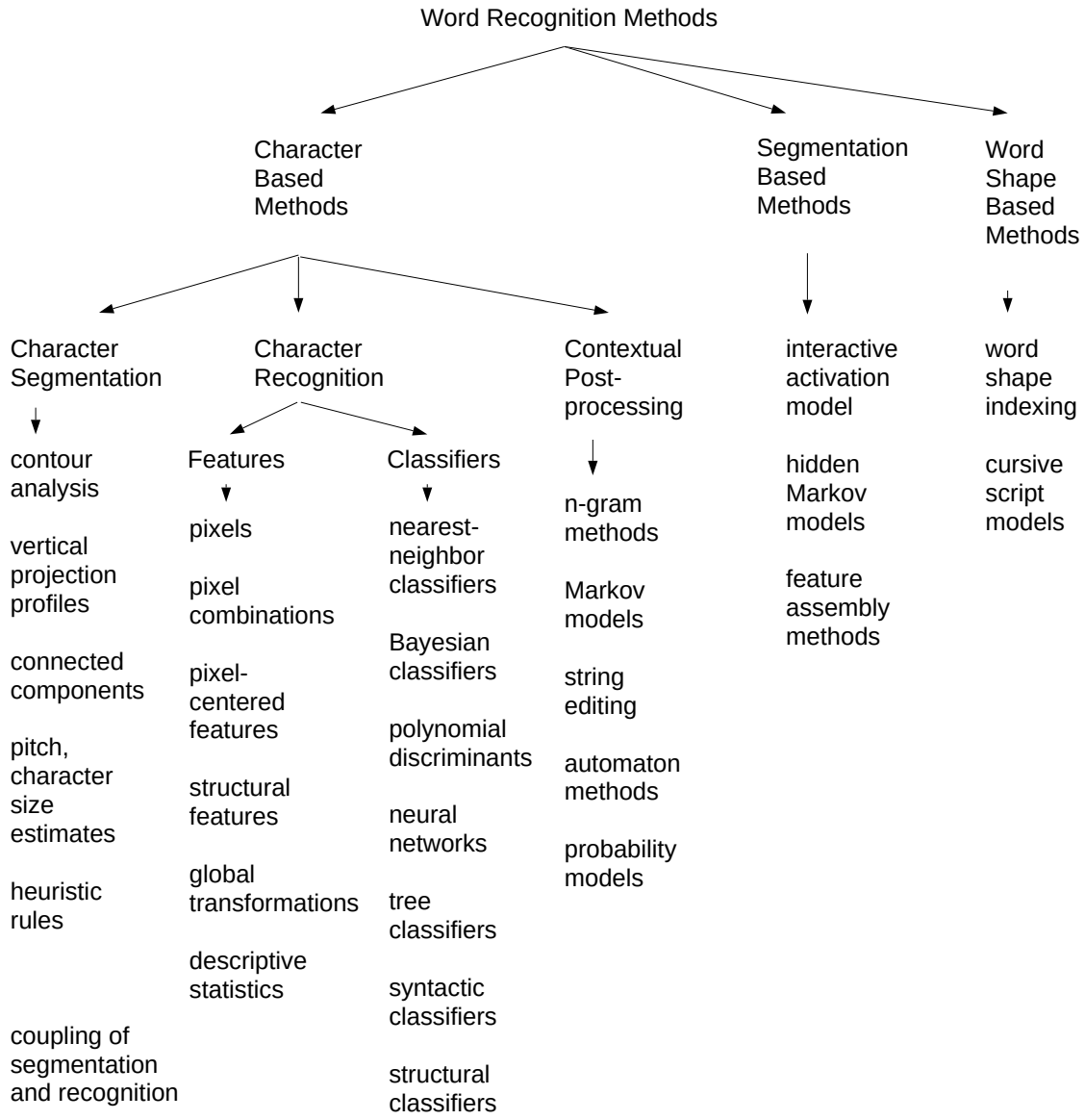


Figure 2.1: Summary of word recognition methods.

studied.

Character Segmentation

Character segmentation is a procedure that divides a word image into individual character images. Commonly used methods [37] include analyses of contours [60], vertical projection profiles [70], and connected components [105] [124] [125] in a text image. Segmentation can also be assisted by pitch [138] and character size estimation. Heuristic rules are often used in locating the best segmentation points.

While satisfactory performance is claimed for many proposed algorithms, none of them works perfectly for noisy text images of a large variety of font types. It is likely that a robust segmentation algorithm needs to use a combination of several techniques selectively for different types of inputs. An interesting approach is the coupling of segmentation and recognition suggested by Bayer [12] and Casey and Nagy [20].

Character Recognition

Character recognition algorithms take in a character image and return decisions on the character's identity. Characters are extracted from word images and are sometimes normalized to a fixed size for recognition. Surveys by Mantas [87] and Mori et al. [93] review the more important approaches. Schuermann [114] reviews commercial systems for character recognition. Character recognition methods differ from one another in the features and feature descriptors used to represent a character image, and in the definitions of the similarity scores and classification procedures.

Features. While in psychological studies, a feature is often defined to be a symbolic entity such as an edge, a loop, an endpoint etc., that a human would identify in an image, in pattern recognition, features are defined by the detection algorithms. According to Simon [126], "A feature is an entity defined by the recognition oper-

ator that detects it in a representation; so the feature and the operator are in fact inseparable." The best features are those that are efficiently computable, versatile to patterns of the same class, and sensitive to patterns of different classes.

The simplest features are the colors or gray-level values of all the pixels in an image [51], usually represented by an N-dimensional vector, where N is the number of pixels in the image. The difficulty in using these features is the complexity involved. For instance, 2^{256} possible patterns may appear in a binary image of size $16 \times 16 = 256$, and 8^{256} in an 8-level gray scale image of the same size. Since no abstraction is applied, all the variances among the patterns are to be handled by a classification procedure. While a direct comparison, called *template matching*, is sometimes used, an efficient classifier design is often desirable. Casey and Nagy [21] describe a decision tree procedure that compares only selected pixels in a sequential test, so that a pixel-by-pixel comparison to all prototypes is avoided.

Features of the second type are predefined combinations of pixel values (Figure 2.2). A practical system using these features is described in [32] and [113]. The configurations of the pixel combinations are flexible. They may appear like a random pattern that has no resemblance to any familiar perceptual entity. The difficulty in using these features is in designing a set of combinations that will be sufficient to distinguish among all classes. Typically, the selection is guided by a mathematical procedure that evaluates the orthogonality among the features [132].

Features of the third type are values that are defined on a local scope centered at one pixel. Baird et al. [8] give a set of templates for detecting such features. More heuristics are built in the definitions of these features than in the random combinations of pixel values. The templates are often purposely designed to detect some perceptual entities, such as an edge oriented to the right boundary of the image. Features of this type are typically detected by a convolution technique [11].

Features of the fourth type are the results of detection of certain perceptual

entities such as a hole, a vertical stroke, a T-joint, or a contour, by an algorithm that is often designed by intuition. They are referred to as structural features. The scope of definition over the image area is usually flexible. Most of the complexity in recognition is handled by the detection algorithms. Examples of these features include the line adjacency graphs [103], and the parameterized structural features including strokes and holes as described in [7] [70].

Features of the fifth type are the results of some global transformations of an image. These include moments [22] [137], Fourier descriptors [48] [123], and the Walsh transform [63]. The difficulty in using these features for recognition is usually in defining an effective similarity score, say, between two sets of Fourier descriptors.

Other features used for character recognition include some global descriptive statistics such as the height-to-width ratio, number of connected components, density of pixels of a particular color, and projection profiles. These features are mostly used for a pre-classification, or together with other shape features in classification.

Classification Procedures. The most commonly used classifiers include nearest-neighbor classifiers, Bayesian classifiers, and polynomial discriminant classifiers. These are referred to as statistical classifiers. They take in feature descriptors

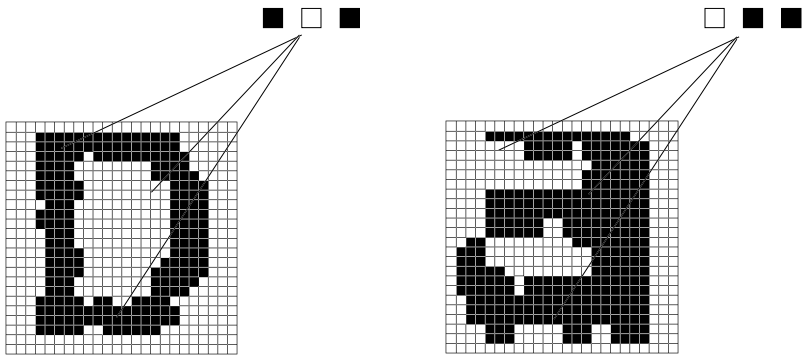


Figure 2.2: A combination of pixel values as a feature.

often in the form of vectors. Classical pattern recognition literature, including books by Duda and Hart [33], and Devijver and Kittler [31], is mostly devoted to discussions of statistical classifiers. Nagy in [96] surveys the statistical classifiers useful for character recognition and compares their complexity.

The nearest-neighbor classifiers [27] perform direct prototype matching using a predefined distance measure to score the similarity between a pattern and those in a class. When pixel values are used as features and a nearest-neighbor classifier is used, the method is referred to as template matching. The prototypes for each class are explicitly stored and used in comparison. Simple distance functions such as the Hamming distance and the Euclidean distance are often used to match features described by vectors. More complicated distance functions, such as a string edit distance, may also be used to match other descriptors, such as symbol strings [83]. The classifier outputs the class with a prototype that is closest to the input according to the computed distances. A variant of the nearest-neighbor classifier is a k-nearest-neighbor classifier which finds k closest matches and uses a voting mechanism to decide on the class [100]. There are also other variants that attempt to reduce the complexity [29].

A Bayesian classifier assigns a pattern to a class with the maximum *a posterior* probability. The class prototypes are used in a training stage to estimate the class-conditional probability density function for a feature vector [33]. To avoid combinatorial complexity, independence between components of the feature vector is often assumed, as in [82] for instance.

A polynomial discriminant classifier [32] [113] assigns a pattern to a class with the maximum discriminant value which is computed by a polynomial in the components of a feature vector. The class models are implicitly represented by the coefficients in the polynomial.

In neural network classifiers, multiple layers of interconnected nodes are used

to imitate the organization of neurons in a biological system. Values of an input feature vector are propagated along the net, modified by stored coefficients in each node and thresholded. The class represented by a node receiving the maximum value is chosen. Recent studies have found that multilayer perceptrons are minimum mean squared-error approximations to the Bayes optimal discriminant function [50] [110].

Tree classifiers [92] are motivated by the need to reduce the complexity in prototype matching. They are intuitively appealing but difficult to design. Possible methods include those by mutual information [116] and by probability models [78] [115].

Grammars at all levels in the Chomsky hierarchy are used to describe class models in syntactic classifiers [2] [4] [35] [43] [101] [102] [133]. These classifiers take in a high level descriptor such as a symbol string instead of feature vectors. The class models are abstracted as grammatical rules that can be used to generate the prototypes.

Other classifiers often compute a heuristic similarity score. These methods are sometimes referred to as a structural approach. Examples include an elastic matching method designed to recognize handwritten characters by Burr [19], the random graphs proposed by Wong [142], as well as heuristic rule-based systems [75].

An Observation. All classifiers use a similarity score to evaluate the match between an input feature descriptor and a class model. A ranking of the candidate classes can always be derived with respect to a pattern according to a linear order of the computed similarity scores, no matter whether the scores are *a posteriori* probabilities, discriminant values, distances to prototypes, or any other heuristic measures. In some instances, the ranking may be incomplete or certain classes may share the same rank. For instance, many syntactic classifiers accept only one class and reject all other classes, thus providing a two-level ranking. A ranking of the classes is very useful in a many-class recognition problem. Our theory is on how to improve

classification correctness using these rankings.

Contextual Postprocessing

Because of difficulties in segmentation and recognition, character decisions often contain errors that need to be corrected by a postprocessing algorithm. Some postprocessing methods assume that the set of allowable words is confined in a given lexicon, and others do not. If a lexicon is not specified, there is often an implicit assumption that only the words permitted by ordinary spelling rules are used, so that we can simply consider the lexicon to be the set of all those words.

Surveys by Hall and Dowling [49], Elliman and Lancaster [37], as well as others included by Srihari in [131] summarize many useful techniques for postprocessing character recognition decisions.

All postprocessing methods need models to link two essential elements: what is in the lexicon, and what is in the input. The methods differ in their emphasis on either of these two aspects, and in the mathematical tools used in modeling. Some methods focus on modeling the probabilities of occurrences of single characters and character groups in the lexicon, others focus on modeling the error patterns in the input. Models with emphasis on letter occurrences include the n-gram methods and the Markov chain methods. Models of error patterns include the string editing methods, joint event probabilities, and other approximate string matching techniques.

N-gram Methods. Digrams are arrays of probabilities of letter pairs that co-occur in words in a lexicon. Similarly, trigrams, quadgrams, and n-grams represent the co-occurrence probabilities of letter triples, quadruples and n-tuples. Trigrams and quadgrams are rarely implemented because of their huge sizes: whereas a digram has $26^2 = 676$ cells, a trigram of an alphabet of 26 characters has $26^3 = 17,576$ cells, and a quadgram has $26^4 = 456,976$ cells. Suen [134] studies the n-gram statistics including the densities and how they vary with word lengths and letter positions.

Riseman [107] describes a binary digram method where the probabilities of co-occurrences of letters are simplified to either 0 or 1. The method uses C_n^2 position-specific digrams for words that are of a particular length n. The method is intended to eliminate invalid words when a character recognizer is unable to discriminate among candidate characters in a small substitution set. The method is generalized to binary n-grams in [108]. N-gram methods are good for efficient elimination of illegal strings. However, simple n-grams are not too useful when the set of valid strings is large. Position-specific n-grams cannot handle missegmentations which are common. Moreover, priorities among candidate character classes in a substitution set are not used.

Markov Model Methods. Words are considered to be generated from a Markov source in the Viterbi algorithm [41] [97] [139] and its variants [64] [65] [121] [122] [128] [130], where dynamic programming is used to maximize the *a posteriori* probability in a search through a *trie* [42]. The *a posteriori* probability is a function of letter occurrence probabilities, letter-to-letter transition probabilities estimated from bigram frequencies, and conditional probabilities that characterize confusions in recognition.

Letter transition probabilities are collective statistics to represent constraints in a lexicon. They are insufficient to forbid production of illegal strings. Several attempts have been made to explicitly use the lexical constraints in addition to the basic Viterbi algorithm. The maximization is confined within the lexicon in [130] and is implemented by the A* algorithm [67]. Strings produced by the Viterbi algorithm may also be checked against the lexicon to ensure its validity [120] [122].

In the Markov models the stages are defined to be positions of the characters in a word. Problems arise when an expected character is absent or in a wrong position due to errors in character segmentation. Another problem is caused by the sequential nature of the Viterbi algorithm. In the modified Viterbi algorithm, a limited num-

ber of alternatives are selected at each stage to avoid a combinatorial search. This restriction has the undesirable effect that recognition errors near the beginning of a word are more likely to cause problems than those near the end.

String Editing Methods. The string editing methods measure the similarity between two strings by the minimum number of operations needed to transform one string into another. This measure is referred to as the string edit distance or the Damerau-Levenshtein metric [28] [99]. The operations can be weighted by a constant cost. The edit operations are models of typographical or spelling errors as well as recognition errors, including substitution of one character for another, insertion and omission of single characters, and reversal of adjacent characters which does not occur in recognition. A dynamic programming approach is described in [140] that computes the minimum edit distance when insertion, omission and substitution errors are allowed. It is extended to handle reversals in [84].

Models of Character Segmentation Results. Bozinovic and Srihari [16] observed that errors in recognition of a missegmented word could be none of insertion, omission and substitution. A split character is a substitution of one character with two, but not an insertion and a substitution; and a merged character is a substitution of two characters with one, but not an omission and a substitution. A stack decoding algorithm is proposed which uses channel probabilities in addition to transition probabilities. The model is theoretically appealing, however, manual training is needed to derive the channel probabilities, because there is no automatic procedure to judge whether a segmentation is correct.

Other Probability Models. Hall and Dowling [49] as well as Kashyap and Oommen [71] propose a method that models patterns of errors with joint event probabilities. It allows a null character in the middle of a word, and is hence able to handle insertions and omissions. The likelihood function for a word is computed using a recurrence relation, similar to the dynamic programming method for computing edit

distances. The relaxation method is used by Goshtasby and Ehrich [47] that allows propagation of compatibility information within a word. These methods also have a shortcoming like the Markov models in that the probabilities as collective descriptions are insufficient to forbid production of invalid strings, so that they need to be aided by a correction procedure.

2.2.2 Segmentation-Based Methods

A second class of word recognition methods uses the contextual information embedded in a given lexicon directly in feature matching. Similar to the character recognition methods, an input word is first segmented into characters. However, the characters are not recognized in isolation. Instead, features extracted from the characters are collected and matched to descriptions of the words in the lexicon. In other words, the features are matched in the context of a word.

The Interactive Activation Model. McClelland and Rumelhart's interactive activation model [89] [111] is a well-known example of this approach. Feature descriptions, characters and words are connected with excitatory and inhibitory links. Features extracted from the character images reinforce or suppress activation of specific characters and words through those links. It is a psychological model and was not intended to be a working recognition algorithm. While the model is interesting, it is unclear how the parameters can be related to statistical knowledge about the input, such as reliability in feature detection, so that performance can be optimized.

Hidden Markov Models. English text is modeled as a hidden Markov chain for handwriting recognition by Kundu et al. [76]. The model assumes that the underlying Markov source is not directly observable, but is observable through another stochastic process. The observable process is represented by a set of symbols, which could be cluster centers of feature vectors that do not necessarily correspond to character classes. Word recognition is achieved by maximizing a probability function.

Feature Assembly Methods. In a less complicated method, descriptors of character features are assembled to construct a feature descriptor for the word. Prototype matching is then performed on the word descriptors. Duderstadt et al. [34] describe a system that uses contour curvatures as features and vectors as descriptors.

Since these methods depend on character segmentation, they are vulnerable to segmentation errors just like the character-based methods. Nevertheless, these methods constitute a unique approach to the problem.

2.2.3 Word Shape Analysis Approaches

Word shape analysis is different from other methods in that a word is treated as a single symbol and recognized as a whole. No character segmentation is applied, so that errors in the segmentation process are avoided. The contextual knowledge contained in a lexicon is used directly in recognition. Word shape analysis is motivated by discoveries in psychological studies of the human reading process [24] [38].

Word Shape Indexing. As part of a computational theory of fluent reading based on psychological studies, a word shape analysis method is proposed by Hull [67], where word shapes are described by structural features such as holes, dots, and vertical bars. The detected features are represented by a code string, which is used to index into a lexicon to retrieve a neighborhood of words sharing the same features. Words in the neighborhood are further discriminated by a hypothesis testing procedure, which uses another set of structural features. The advantage of this approach is that errors in character segmentation and premature decisions on character identities can be avoided. The features and the matching procedure need to be refined for the method to be applied to noisy images of multiple font types.

Cursive Script Recognition. Word shape analysis is also used for cursive script recognition [18], where character segmentation is very difficult. Hayes [54] proposes a hierarchical relaxation method that describes word shape with several

graphs: the feature graph, the stroke graph, the possible segmentation point graph, and the letter graph. Several relaxation processes are used to enforce consistency at various levels. An interesting idea proposed is that multiple segmentation guesses are selected in a relaxation process. It is shown that the performance depends on the reliability of stroke and letter identification, which was unsatisfactory.

A method is proposed by Simon [127] that uses regular and singular features to describe a cursive script. The emphasis is in obtaining invariant features to describe images of the same word. Features are described by a chain of symbols and matched by a heuristic method. The study is ongoing and reported results are preliminary.

Farag [39] suggests representing a cursive script by a sequence of stroke directions, in the order in which the strokes are entered. A nonstationary Markov model is used to match the stroke sequences. Farag's method, as well as other on-line methods surveyed in [136], uses timing information in recognition and is not solely based on visual signals. However, if a reliable link can be developed that recovers on-line information from off-line data, these methods may be useful in off-line applications as well.

2.3 Conclusion: Rationale for a New Approach

As shown in our review of methods for word recognition, many alternative solutions may exist for a particular recognition problem. Preliminary attempts to apply multiple approaches simultaneously to a given problem have suggested some advantages, such as a reduction in error rate. With examples from methods proposed for text recognition, we can now summarize the advantages of using multiple classifiers as follows:

Multiple Descriptors for A Mixture of Features. Since each classifier takes in one feature descriptor, many types of descriptors can be used in a multiple classifier system to highlight different information needed to characterize a pattern.

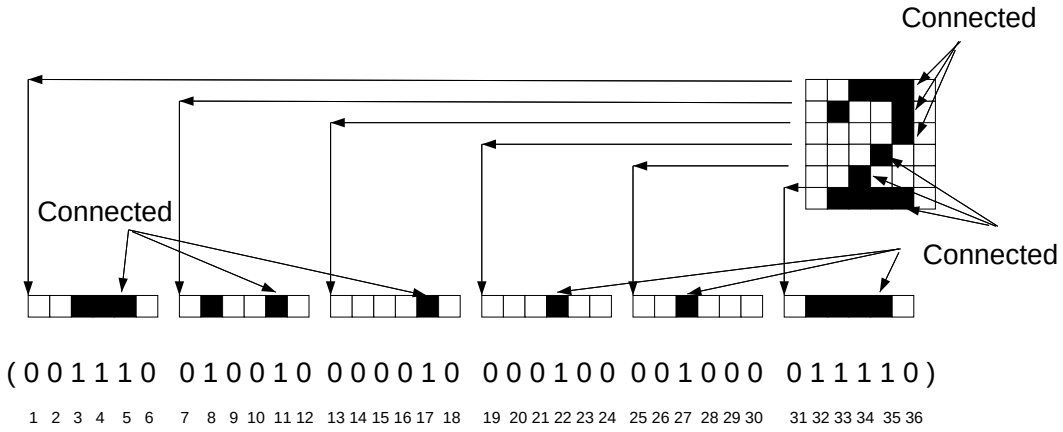
Figure 2.3 gives an example.

Cooperation of Multiple Classification Procedures. In a multiple classifier system, each feature descriptor can be matched separately by a classifier. This eliminates the need to define a single similarity score for descriptors of different types. A classifier that is most effective for a certain set of features can be used. For instance, a syntactic classifier using grammars may be used to match a chain code, and a nearest-neighbor classifier using Euclidean distance may be used to match a vector of pixel values.

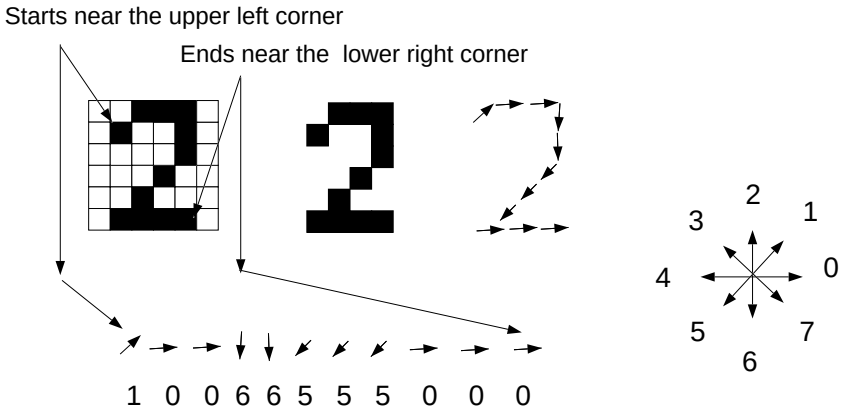
Dynamic Adaptation to Input Conditions. Features of different types are sensitive to different input conditions, such as various types of image degradation. It is difficult to optimize the performance of a single classifier that uses a large collection of features under all circumstances. An alternative approach is to construct classifiers that are specialized for different types of inputs, and apply only those classifiers that are the best for each particular input.

Several methodologies have been attempted to combine decisions by multiple classifiers. These include majority votes of top choices and combination of confidence values. In a many-class problem, secondary choices by the classifiers are often useful for recovering from near misses. It would be ideal if each class is given a confidence score that represents a classifier's belief in its being the true class. However, obtaining confidence scores from the results of feature matching is a difficult problem with no known general solution.

We have observed that all classifiers use a similarity score to evaluate the match between a pattern and a class model, and a ranking of the given classes can always be derived from such scores. Therefore, decision combination based on the rankings is feasible. In the next chapter, we will present a decision combination theory based on the rankings.



(a) A feature vector obtained by a row-by-row scan across the image. Each component of the vector represents the color of a pixel. Though the vector can be used to reconstruct the image, it does not make explicit some information that is useful to characterize the appearance of the pattern. For instance, it is obvious to our eyes that in the image, some black pixels are connected to each other. Yet it is difficult to tell such relationships directly from the vector.



(b) A chain code description of an image. This descriptor captures the relationship between adjacent black pixels in a partial contour trace. However, information about the positions of the black pixels is not explicitly represented. For instance, it is difficult to tell from the descriptor that, the chain starts near the upper left corner and ends near the lower right corner of the image.

Figure 2.3: Different descriptors of an image.

An example of a complicated many-class recognition problem is visual word recognition. Previous studies have long been focused on character-based methods. Some recent investigations suggest that treating a word as a whole object is a promising alternative. Seeing the advantages of each approach, we will attempt to construct a robust solution that integrates both character-based and word shape based techniques together with an intermediate method based on character segmentation. We will show how the theory of multiple classifier systems can be applied in this solution, and demonstrate its success by experiments with real-world data.

Chapter 3

A Theory of Multiple Classifier Systems

The traditional view of a pattern recognition system is a single-stream process that uses a fixed set of features, a single descriptor, and a unique classification procedure. This view is too restrictive because it allows no flexibility to use features and classifiers of different types simultaneously. We propose a pattern recognition system that consists of a set of independent classifiers which use different feature descriptors and classification procedures in parallel. A decision combination function is then necessary to integrate the contributions of these classifiers. This chapter presents a theory for decision combination in such a system.

Organization of Discussion. We assume that each classifier outputs the classification decisions in the form of a ranking of a given set of classes with respect to an input pattern. In Section 3.1 we discuss the reasons for using the rankings. Our theory focuses on decision combination. Two objectives of decision combination, class set reduction and reordering, are discussed in Section 3.2. Methods for achieving each objective are proposed in Sections 3.3 and 3.4. The use of multiple classifiers implies the possibilities of a dynamic selection and a multiple stage organization. Issues involved in dynamic classifier selection and multiple stage organizations are discussed in Sections 3.5 and 3.6.

3.1 Advantages of Using Rankings

3.1.1 Using Rankings Instead of Unique Class Choices

In statistical pattern recognition, a classifier is defined to be a function that maps an input feature vector to an assignment of the vector into one of the given classes [33]. Typically a measure called a discriminant, which is a function of values of the input vector, is computed. A class label is assigned based on the value of the discriminant according to a decision rule. An example decision rule is to assign an input to one of two classes depending on the algebraic sign of the discriminant value ([33], page 20).

According to this definition, the output of a classifier is always a unique class choice. Each of the given classes is decided to be either the class of the input or not. A classification decision is either correct or wrong. This is the reality for a two-class problem, on which most of the statistical classification methods are focused. Generalization of the methods to many-class problems is often considered trivial.

However, there exists a fundamental difference between many-class problems and two-class problems. The decisions for a two-class problem are always binary, one class being identified as the class of the input pattern and the other not. For a many-class problem, the classification decisions can be much more complex. It is often possible to assign a ranking on the set of classes, instead of identifying only the most likely class in the set. For example, instead of choosing the class with the maximum *a posteriori* probability, a Bayesian classifier can order all the classes by their *a posteriori* probabilities.

A ranking is particularly useful for problems that involve a large class set, where it is highly probable that the classifiers are unable to uniquely identify the correct class, but are able to include it in several guesses. A ranking of the classes carries much more information than a unique class choice. Analyses and evaluation of the classifier performance will be more fruitful if the rankings are taken into consideration.

Therefore, we consider it preferable to represent a classifier's decision as a ranking of the class set with respect to an input pattern, instead of as a unique class choice.

3.1.2 Using Rankings Instead of Numerical Similarity Scores

As we have pointed out in Chapter 2, all classifiers use a similarity score to evaluate the match between a pattern and a class model. In this discussion, we are concerned with whether decision combination should be based on those scores, or the rankings derived from those scores. There are two reasons in favor of using the rankings: i) not all the similarity scores are good predictors of correct classification; and ii) a common model is needed to compare scores of different types. These two reasons are explained as follows.

Correlation Between Similarity Scores and Classification Correctness

One difficulty with using the numerical scores is that not all such scores are good predictors of the true class identity, if they are taken out of the context of the scores computed for other classes. This is especially true for scores computed by nonparametric classifiers, such as the distances computed by a nearest-neighbor classifier.

Outputs from a classifier must contain sufficient hints to the identity of the true class to be useful in decision combination. That is, the scores associated with the correct classes must be sufficiently distint from those associated with the wrong classes. The effectiveness of the combination function is dependent on how well the scores associated to the right and wrong classes separate from each other. Their separation is influenced by the existence of confusing classes, which are more common when the class set is large. As we will see, as the size of the class set grows, the numerical similarity scores may become inferior to the rank scores in predicting the true classes. This is illustrated in the following example.

Example

A nearest-neighbor classifier is used to recognize a word image as one of a given set of words. Each image is represented by a feature vector (for the details of this classifier, see discussions of the *stroke direction features* in Section 4.8 in Chapter 4). Each class is represented by one prototype vector. Classification is performed by computing a distance from an input vector to each of the prototype vectors. These distances are the concerned numerical similarity scores. For each input image, the given set of words are sorted by these distances and a rank score is then assigned to each word.

A set of 50 word images were used to test this classifier. They are instances of five distinct words in many fonts. The class set consists of those five words. A word appearing in an image is called the *true word* for that image. All the other four words are called the *wrong words*. The distances and rank scores computed for the true word and the wrong words for each of the 50 test images are displayed in Figure 3.1. It can be seen that, the values of the true words and those of the wrong words separate well from each other on both the distance scale and the rank scale.

The same test was repeated twice when the class set was enlarged to 50 and 500 words by adding in confusing words that look similar to the original five words. The same set of images was used in all the three tests. The distances and rank scores of the five most similar classes to each input are displayed in Figures 3.2 (a) and (b). As we can see from these figures, the separation between the distances of the true and wrong words is less obvious than before. Nevertheless, the separation between the rank scores of the true and wrong words are well maintained.

To compare the separations, we consider the probability of obtaining a correct decision if a threshold is applied to each scale. Suppose that we always take the class ranked at the top as the output decision, that is, to threshold the rank scores at 1. In test 1 with 5 classes, all the classes above the threshold are correct, and therefore the correct rate is 100%. To use the distances and include the same number of true

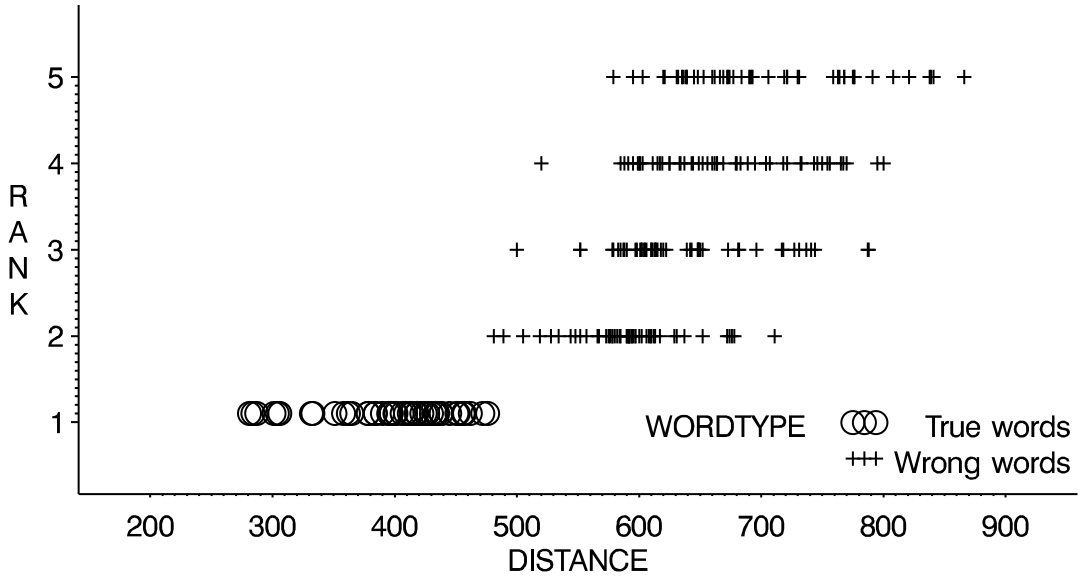
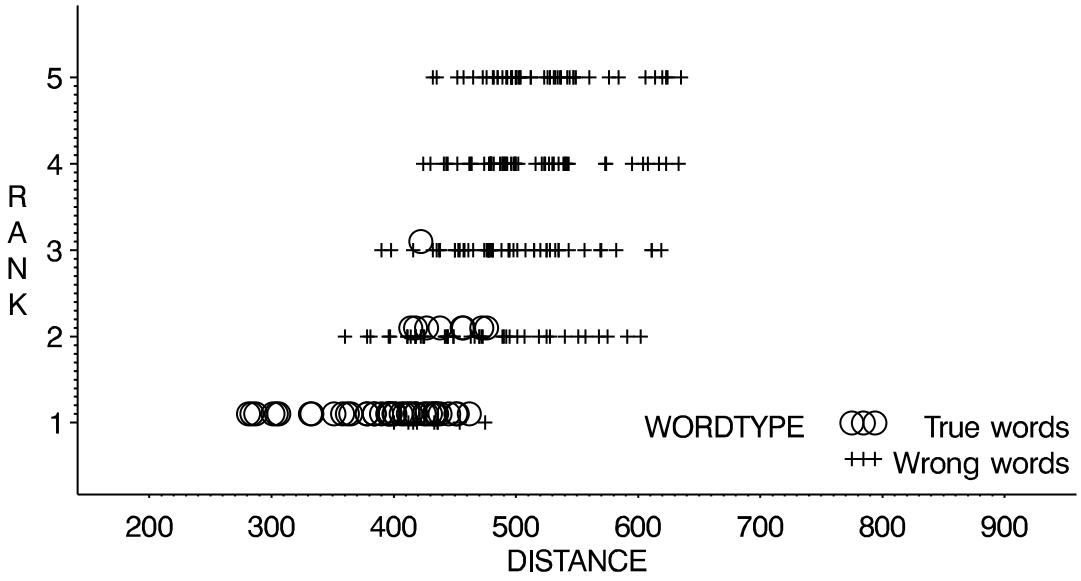


Figure 3.1: Distances and rank scores for 50 inputs and 5 classes. The true classes are well separated from the wrong classes on both distance and rank scales.

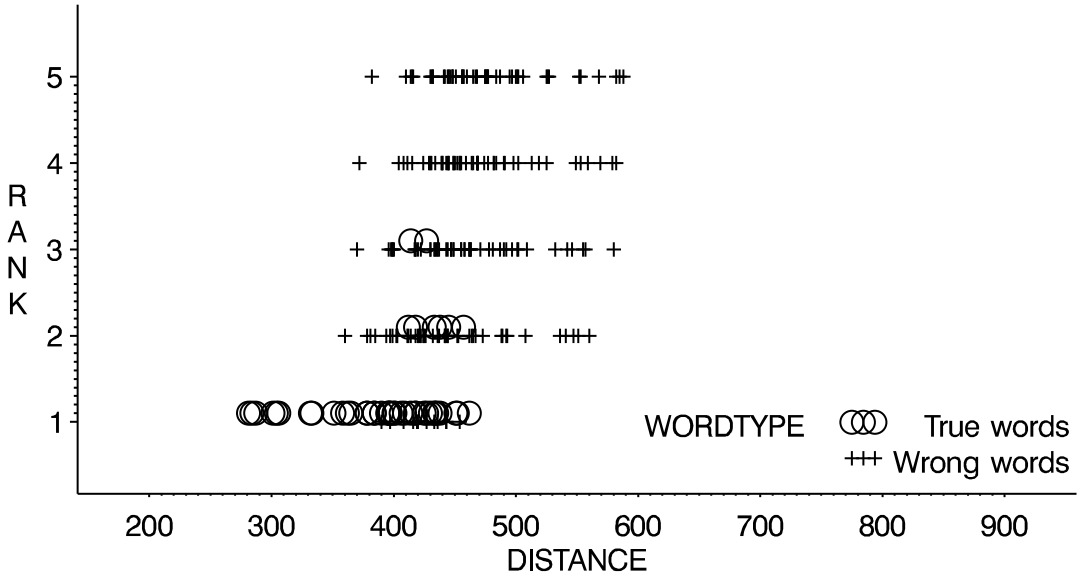
classes, we can threshold the distance at 476, and all the classes included are also correct. Therefore in test 1 we can threshold the outputs at either scale to achieve the same correct rate.

However, in test 2, if we always take the top class choice (rank 1), we can get 41 true classes in the 50 top choices, and the rest 9 are wrong. Therefore we have an 82% (41/50) correct rate for the choices at or above the rank threshold. To include the same number of true classes, we need to threshold the distances at 438. This gives 41 true classes but also 32 wrong classes. Therefore the correct rate is only 56% (41/73).

In test 3, we can get 38 true classes by taking only the classes ranked at the top. The correct rate is then 76% (38/50). To include the same number of true classes, we need to threshold the distances at 433. This gives 38 true classes but also 94 wrong



(a) Distances and rank scores for 50 inputs and 50 classes.



(b) Distances and rank scores for 50 inputs and 500 classes.

Figure 3.2: Distances and rank scores for enlarged class sets. The true classes are no longer well separated from the wrong classes on the distance scale. Nevertheless, much of the separation is maintained on the rank scale.

Table 3.1: Comparison of effects of rank thresholds and distance thresholds.

number of given classes	rank threshold	number of true classes	number of wrong classes	correct rate	distance threshold	number of true classes	number of wrong classes	correct rate
5	1	50	0	100%	476	50	0	100%
50	1	41	9	82%	438	41	32	56%
500	1	38	12	76%	433	38	94	29%

classes. The correct rate is therefore only 29% (38/132). Table 3.1 summarizes the correct rates when these thresholds are applied.

It can be observed from Table 3.1 that the correct rates on the distance scale drop faster than those on the rank scale. That means the rank scores are more stable predictors of correct classification than the distance scores. The same phenomenon is observed with other classifiers in our experiments. This makes rankings a preferable basis for classifier combination.

A Common Model for Normalizing the Numerical Scores

Another difficulty in using the numerical scores is that we need a common scale to compare the scores computed by different types of classifiers. Depending on the type of a classifier, the numerical scores can be defined on many possible scales: a Minkowski metric, the Bayesian posterior probabilities, values of a discriminant function, string edit distances, probabilities associated with stochastic production rules, or any heuristic measures.

To combine the decisions by different types of classifiers. these scores need to be normalized to a common scale. As their definitions vary, these scores may differ from one another to a large extent in their ranges and distributions, so that a consistent normalization scheme is difficult to obtain. Moreover, it is unclear as to whether the scores for all possible inputs should be normalized to the same range, or a distinct range for each particular input.

A ranking of the class set is an existing scale to which all kinds of similarity scores can be converted, without additional assumptions on their distributions. It is a linear scale with equal intervals.

The similarity of an input pattern to each class, as determined by a particular classifier, is then represented by the rank score the classifier gives to that class. The combined similarity is computed by merging all the marginal contributions by each classifier. This is analogous to the computation of a similarity score from the feature descriptors, where the score is derived from a combination of marginal contributions by matches of individual feature components.

3.1.3 Summary of Advantages of Using Rankings

The advantages of using rankings to represent class decisions are now summarized:

1. Rankings contain more information than unique class choices.
2. Rank scores may better predict correct classifications than numerical scores.
3. Rankings are common scales for normalizing incomparable numerical scores.
4. No assumptions about the statistical distributions of the scores are needed.
5. Derivation of rankings from numerical scores is simple.
6. The method is generally applicable to all kinds of classifiers.

Therefore we choose a ranking of a given class set to represent a classifier's decision with respect to an input pattern. Recognition is then considered as a committee decision process. Functions are defined to determine a consensus class ranking by combining the individual rankings. Certain classifiers can only decide on accepting or rejecting a class. Their decisions are treated as a special type of ranking, where each class receives either one of two distinct rank scores, such as 0 and 1.

A Limitation. For simplification, we assume that in each combination, the same number of distinct ranks are used by each classifier for a given set of classes. In

practice, it is possible that ties exist in the rankings produced by some classifiers, such that a different number of distinct ranks are used by each classifier. If this happens, we suggest that the rank scores on the finer scales be converted to ranks on the coarsest scale. For example, suppose that four classifiers are to be combined, one of which can only accept a single class as the correct one and rejects all other classes. In this case, even though the other classifiers can produce a complete ranking of the class set, these rankings should be converted to binary scores, with one score for the top choice and another for the rest. Alternatively, the classifiers using the same number of distinct ranks may be combined first, and the combined rankings can then be converted to the same scale as those used by others, so that they can be combined at a second stage.

3.2 Objectives of Decision Combination

A multiple classifier system is justified if the combined decisions are better than those given by the component classifiers applied in isolation. An effective decision combination function should take advantage of the strengths of the individual classifiers, and avoid their weaknesses. The effectiveness of the decision combination function is critical to the success of a multiple classifier system.

There are two approaches to classifier decision combination, each with a different objective. They are referred to as *class set reduction* and *class set reordering*. In class set reduction, the objective is to extract a subset from a given set of classes, such that the subset contains the true class of the input pattern. In class set reordering, the objective is to derive a consensus ranking of the given classes, such that the true class is ranked as close to the top as possible.

It is useful to differentiate between the two objectives, reduction and reordering, each of which can be achieved by different means. The two objectives also lead to

different criteria for performance evaluation. In reduction, the emphasis is on reducing the size of the class set while the true class is not omitted. In reordering, the focus is on the position of the true class in the final ranking. Some methods for decision combination produce reduced class sets that hardly ever miss the true class, but the classes within the set are not ordered. Other methods may produce good rankings where the true class is often ranked close to the top, but, occasionally for a bad pattern, the true class may not be contained in a small neighborhood near the top.

For example, in a discrimination problem involving 100 classes, a method may always produce a reduced set of 5 classes that always includes the true class, but cannot tell which one it is among those 5 choices. Another method may be able to rank the true class at the top for 99% of the test patterns, but for one particularly bad input, it ranks the true class at the 50th position. If one miss is affordable, only the top choices need to be used. Otherwise, up to 50 choices need to be considered for every input. Therefore, which of the two methods is preferable depends on the needs of a particular application.

These two objectives are equivalent when special requirements are added. If it is required that the result set derived by a reduction approach always contains only one class, then it is the same as requiring a reordering approach to rank the true class always at the top. If the rankings derived by a reordering approach are so good that the true class is always ranked above a certain position, then it is always possible to include the true class in a neighborhood extracted from the top of the ranking up to that position. The two approaches can be applied to the same problem, so that the set of classes may first be reduced and then reranked, or first reranked and then reduced to a small neighborhood from the top of the ranking.

The rest of this chapter details methods that are useful for achieving these two objectives. In Section 3.3, two approaches to class set reduction as well as some

approximation methods are presented. In Section 3.4, three methods for class set reordering are presented. The simpler methods are first presented which are followed by the more sophisticated ones.

3.3 Methods for Class Set Reduction

Class set reduction is aimed at reducing the number of classes in a given set without losing the true class. The criterion for success is therefore two-fold: the size of the result set should be minimized, and the probability of inclusion of the true class in the result set should be maximized.

Assuming that the decisions by a number of classifiers for a set of training patterns are available, an intersection approach and a union approach are proposed to reduce the class set. Several approximation methods for overcoming certain practical limitations are described afterwards.

3.3.1 The Intersection Approach

In the intersection approach, a large neighborhood is obtained from the top of the class ranking by each classifier with respect to each pattern, and their intersection is the result set. Hence a class is in the result set if and only if it is included in *all* the neighborhoods. In other words, a class is chosen if all the classifiers agree that it is possibly a true class. Figure 3.3 illustrates the intersection approach to class set reduction.

The sizes of the neighborhoods are determined by the ranks of the true classes in the worst cases. Assume that a set of training patterns is available, for which all the true classes are known. To derive thresholds on the neighborhood sizes, each classifier ranks all the given classes for each pattern. After rankings are obtained for

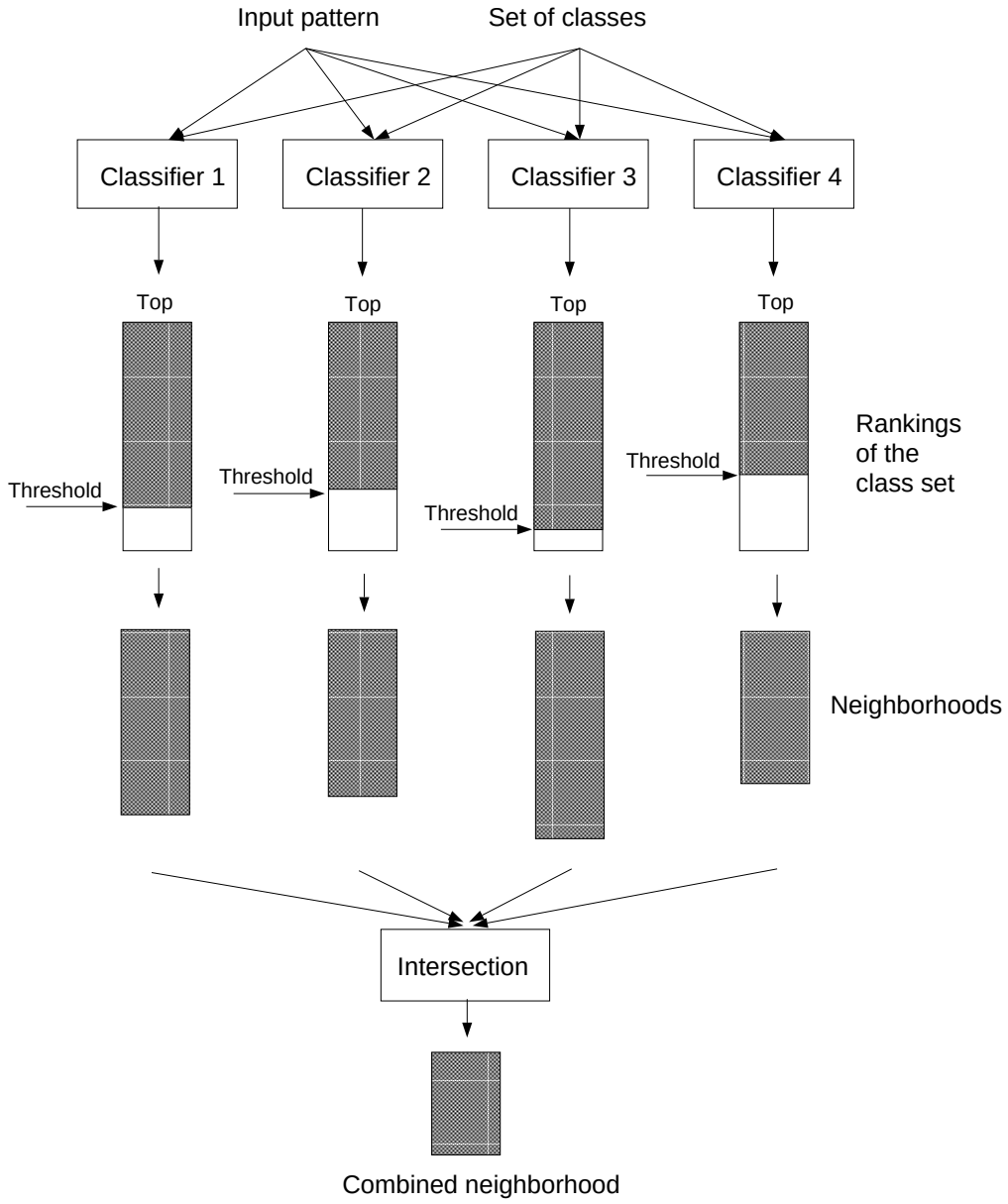


Figure 3.3: The intersection approach to class set reduction. A large neighborhood is obtained from each ranking. The intersection of these neighborhoods is the result set.

all training patterns, the lowest rank ever given by each classifier to any true class in the set is determined. These lowest ranks are taken as thresholds on the neighborhood sizes. For each test pattern, the same set of classifiers are applied to rank the classes and a neighborhood of size thus determined is taken from the top of each ranking. The set of classes is then reduced by taking the intersection of these neighborhoods. A classifier is redundant if its threshold is equal to the size of the class set.

Example

The intersection method is illustrated by an example shown in Table 3.2. The table shows the ranks assigned to the true class for each of 6 training patterns $I_1, I_2, ..., I_6$ by four classifiers C_1, C_2, C_3, C_4. For each classifier, the lowest rank ever assigned to a true class in the training set is determined and shown at the bottom of the table. These can be taken as thresholds on neighborhood sizes in test runs. According to the table, in order not to miss any true classes, 34 decisions should be taken from C_1, 36 decisions from C_2, 29 decisions from C_3, and 24 decisions from C_4. The intersection of these decisions is computed. The intersection is smaller than the class set if at least one of the thresholds is less than the size of the class set.

Table 3.2: An example of determining neighborhood sizes for an intersection.

	$rank_j^i$			
$I_i \backslash C_j$	C_1	C_2	C_3	C_4
I_1	3	12	1	24
I_2	1	5	29	12
I_3	34	3	4	6
I_4	9	7	6	7
I_5	4	36	5	5
I_6	16	2	3	4
$colmax_j$	34	36	29	24

If the probability of including the true class is to be traded off for a minimum size result set, all the rank thresholds can be set to one. That is, only the top class choice from each classifier is taken and their intersection is taken as a result set. The

result set is empty if there is a disagreement among the classifiers on the top choice, in which case the system refuses to commit to a decision for that particular pattern.

3.3.2 The Union Approach

In the union approach, a small neighborhood is obtained from each classifier. The *union* of these neighborhoods is output as a result set. A class is in a result set if and only if it is in *at least* one of the neighborhoods. Figure 3.4 illustrates the union approach to class set reduction. A max-min procedure has been developed to select the thresholds useful for the union approach, which is illustrated in the next example and formally presented afterwards.

Example

The procedure for threshold selection in the union approach uses a performance table for a training set like the one shown previously in Table 3.2. The procedure is illustrated in Table 3.3. The left half of Table 3.3 shows the ranks of the true class of each training pattern. The best (minimum) rank in each row is determined, and entered under the classifier that produces it in the right half of the table. The maximum of all these minima is computed for each column. It can be shown that, if a neighborhood is obtained from each classifier by thresholding the rank scores using these maxima, a union of the neighborhoods for each training pattern always contains its true class.

A Max-min Procedure for Threshold Selection

The max-min procedure is described as follows.

Let $rank^i{}_j$ be the rank score of the *true class* of a training pattern I_i by classifier C_j, where $i = 1, 2, ..., n$ and $j = 1, 2, ..., m$.

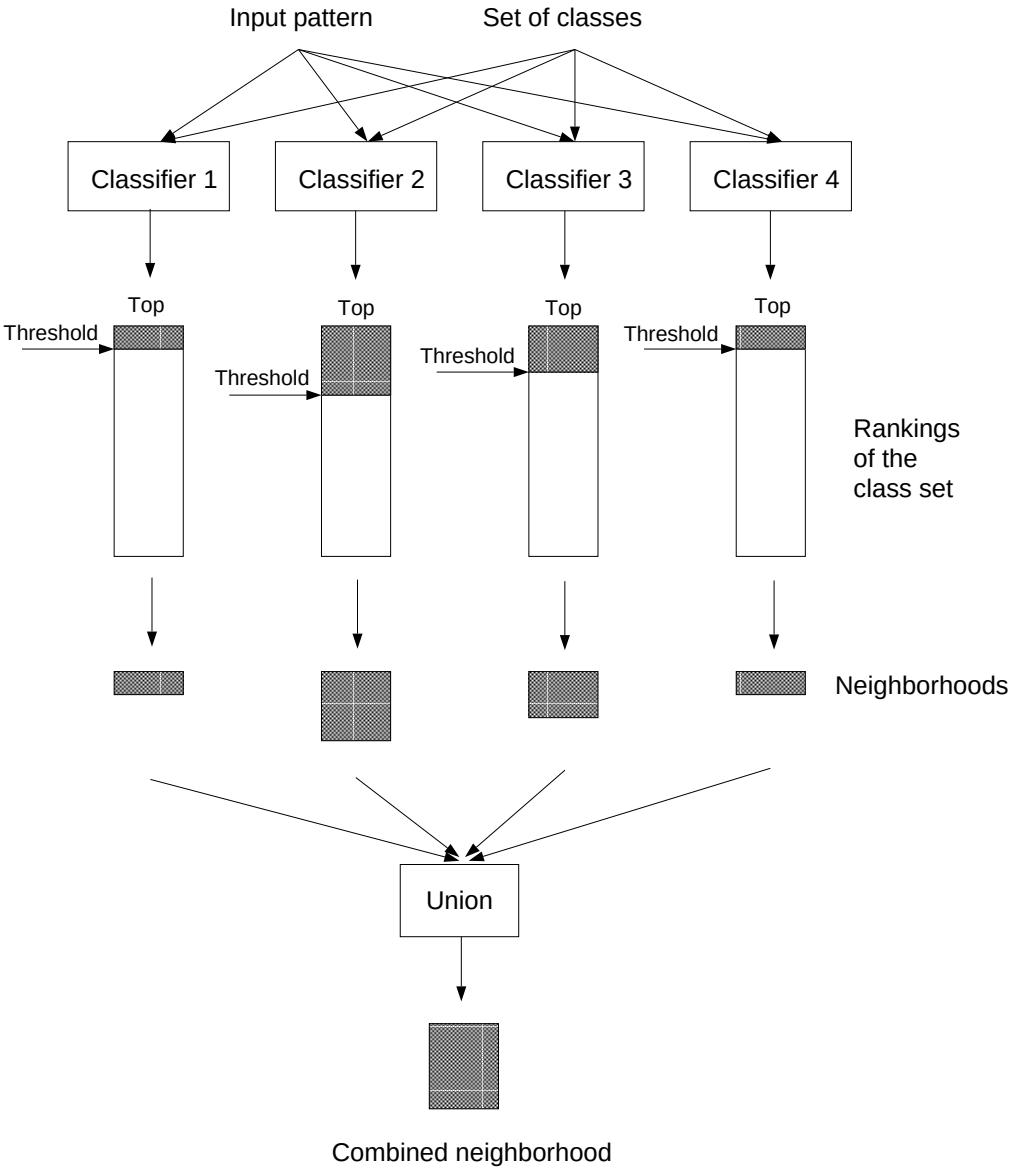


Figure 3.4: The union approach to class set reduction. A small neighborhood is obtained from each ranking. The union of these neighborhoods is output.

Table 3.3: An example of determining neighborhood sizes for a union.

	$rank^i_j$				$rowmin^i_j$			
$I_i \backslash C_j$	C_1	C_2	C_3	C_4	C_1	C_2	C_3	C_4
I_1	3	12	1	24	0	0	1	0
I_2	1	5	29	12	1	0	0	0
I_3	34	3	4	6	0	3	0	0
I_4	9	7	6	7	0	0	6	0
I_5	4	36	5	5	4	0	0	0
I_6	16	2	3	4	0	2	0	0
$colmax_j$					4	3	6	0

Define $$rowmin^i{}_j = \begin{cases} rank^i{}_j & \text{if } \forall p, 1 \le p \le m, rank^i{}_j \le rank^i{}_p \\ 0 & \text{otherwise,} \end{cases}$$
$$colmax_j = Max_i\ (rowmin^i{}_j).$$

We can then claim that for all i, there exists j such that $rank^i{}_j \le colmax_j$. Otherwise, there exists k, such that for all j, $rank^k{}_j > colmax_j$. Then there exists l such that $rowmin^k{}_l = Min_j\ (rank^k{}_j) > colmax_j$ $^{(*)}$, and $colmax_j \overset{(\text{def})}{\ge} rowmin^k{}_l \overset{(*)}{>} colmax_j$, which is a contradiction.

This claim says that for any pattern I_i in the training set, if we take the top $colmax_j$ choices from classifier C_j for each j, and take the *union* of all these choices, then the true class for pattern I_i must be contained in this union. The $colmax_j$ values can be used as thresholds on the rank scores to obtain the neighborhoods.

Intuitively, $colmax_j$ for classifier j represents the worst case decision by classifier j, when no other classifiers can do better than it. Consider an ideal system such that we know, by observing the performance on a training set, that for any input pattern, there always exists a classifier which can rank its true class at the top. In this case all the $colmax_j$'s equal to one, that is, we only need to consider the top choice from each classifier. Consider another system with only one classifier. In this case $colmax_1$ is the worst case rank assigned to a true class by that classifier.

Redundant Classifiers

For any j, if $colmax_j = 0$, then classifier j is redundant, meaning that its decision is always inferior to some other classifier's and should not be included in the union. Classifier C_4 in Table 3.3 is redundant in this sense.

Let $\{C_{j_1}, C_{j_2}, ..., C_{j_k}\}$ be a subset of the original set of classifiers. Assume that each classifier assigns a distinct rank to each class with respect to an input pattern. Then the theoretical maximum union size is

$$MUS = \sum_{j=j_1}^{j_k} colmax_j$$

For the example given in Table 3.3, $MUS = 4 + 3 + 6 = 13$. That means all the true classes for the patterns in this training set are included in a union of size *at most* 13, which is smaller than the number of choices needed from any individual classifier for the same correct rate.

Yet one may notice that for the example in Table 3.3, if we use only the classifiers $\{C_1, C_2\}$, or $\{C_2, C_3\}$, and obtain the thresholds $colmax_j$ following the aforementioned procedure, we need a union of an even smaller size (11 in both cases) to include all true classes. In fact, the smallest union size (10) can be obtained if we use the subset $\{C_1, C_3\}$. We can therefore say that a subset $\{C_{j_1}, C_{j_2}, ..., C_{j_k}\}$ with minimum MUS is optimal for the given training set.

A direct implementation of this classifier selection technique requires evaluation of MUS over the all possible combinations of the classifiers, and is therefore impractical for a large number of classifiers. One simplification is to omit the subset selection procedure, and use all the classifiers except the redundant ones. Another simplification is to approximate the theoretical optimum by a greedy algorithm, which aims at stepwise reduction of MUS by adding classifiers one by one.

Whereas MUS is the theoretical maximum size of the union, the true maximum or average union size can only be determined experimentally by actually obtaining all the rankings and forming the unions. This is because some classes may exist in the neighborhoods extracted from several rankings, such that the size of the union is smaller than the sum of their individual sizes (MUS). Hence the minimum MUS does not necessarily guarantee an optimal experimental union size. The difference between the true union size and MUS is determined by the correlation between the decisions by the classifiers.

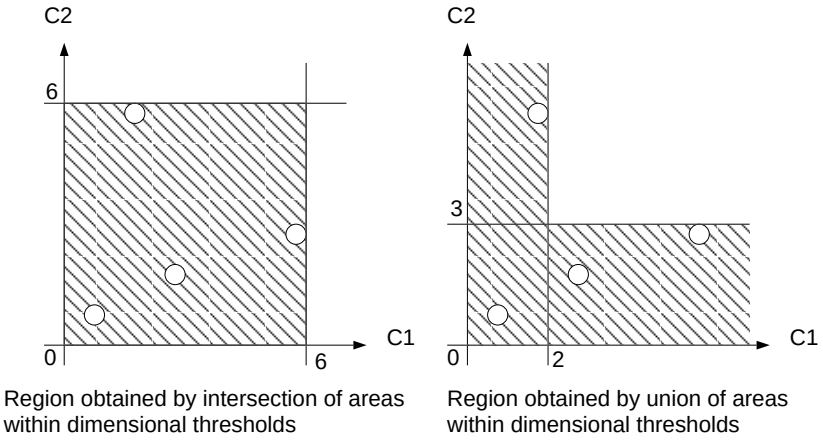
3.3.3 Remarks

Obviously, the intersection approach is useful only when all the classifiers have moderate worst case performance, so that small neighborhoods are obtained without missing the true class. However, this is usually not the case for a set of specialized classifiers working with small sets of features, so that each classifier may be excellent for certain types of inputs but poor for others. The neighborhoods may be undesirably large, since they are determined by the worst case behavior of the classifiers.

The union approach is preferred if the classifiers are specialized on different types of inputs. The ideal case is that the set of classifiers is sufficiently rich and all types of inputs are included in their specialties. That is, for each pattern, there is always one classifier that recognizes it well. Referring back to Table 3.3, this corresponds to the case that the row minima are small and so are the column maxima. Small neighborhoods are obtained in such cases, and therefore the unions are small. Essentially, the union approach focuses on the best case behavior of each classifier. The strength of each individual classifier is preserved, while its weakness is compensated for by others.

We therefore conclude that the union approach is preferred for combining a set

	C1	C2	Min	
I1	1	1	1	1
I2	3	2	0	2
I3	2	6	2	0
I4	6	3	0	3
Max	6	6	2	3



Region obtained by intersection of areas within dimensional thresholds

Region obtained by union of areas within dimensional thresholds

Figure 3.5: Example regions determined by the intersection and the union methods. All the true classes are included in each region.

of highly specialized classifiers, when the objective is to reduce the class set without missing the true class.

Note that both the intersection approach and the union approach can be applied to the space of numerical similarity scores as well. When applied to the numerical scores, the intersection method is equivalent to finding a rectangular convex hull which contains all the true classes and is most succinct (Figure 3.5). However, since each classifier may use a different range of numerical scores, the union method is not as useful as it is with the rankings. For instance, the scores given by one particular classifier may be constantly smaller than those by other classifiers, simply because a smaller numerical range is used.

3.3.4 Approximation Methods

The thresholds given by the max-min procedure are absolute, in the sense that they guarantee 100% success in including the true class for every training pattern. Because

of this, the effectiveness of the method is sensitive to outlying worst cases. For instance, if there is a pattern where no classifier ranks its true class above 50, then one of the thresholds is bound to be at least 50, and the resultant union must contain at least 50 classes. There is also a similar problem with the intersection method. For instance, if there happens to be one training pattern where a certain classifier C puts its true class at the bottom of its ranking, the threshold for C is set to be the largest rank, hence C has no contribution in reducing the class set.

In practical applications, if there are few outlying cases, and if the cost of a small number of errors is affordable, an approximation method is preferred over these methods.

One approximation method is to remove extremely bad patterns from the training set and proceed with the same procedure. The number of bad patterns to be removed is chosen according to the desired correct rate. For instance, if a correct rate of 95% is desired, then the worst 5% of all training patterns are removed, and a threshold selection procedure can be applied to the rest.

Another approximation method is a voting approach described as follows. A threshold is applied on the rankings produced by each classifier. A class receives one vote if it is ranked above that threshold. A class is included in the result set if it receives more than a predetermined number of votes from the classifiers. It can be seen that the intersection approach is equivalent to requiring full votes for each output class, and the union approach is to requiring at least one vote for each class.

The design of the voting scheme is more difficult than either the intersection method or the union method, since the thresholds and the number of votes need to be determined by considering the tradeoff between the error rate and the reduction rate. This is better done in the context of a particular application, where the true

reduction rate can be determined experimentally.

A third approximation method is to achieve class set reduction by first obtaining a consensus ranking of the class set, and then taking a small neighborhood near the top of the ranking. The size of the neighborhood is determined by observing, using a training set, the probabilities of inclusion of the true classes at and above each position in the consensus ranking. Methods to obtain such a consensus ranking are discussed in Sections 3.4.1 to 3.4.3 and summarized in Section 3.4.4.

3.4 Methods for Class Set Reordering

Class set reordering is aimed at promoting the rank of a true class. The criterion for success is the position of the true class in the resultant ranking, as compared to those in the individual rankings before combination. A method is considered successful if the probability of having the true class near the top of the combined ranking is higher than those in each of the original rankings.

3.4.1 The Highest Rank Method

One simple method to combine several rankings to reorder the class set is the highest rank method. Similar to the union approach, the highest rank method is good for combining a small number of classifiers, each of which specializes on inputs of a particular type.

Assume that, for each input pattern, m classifiers are applied to rank a given set of classes. Thus each class receives m ranks. The minimum (highest) of these m ranks is assigned to that class as its score. The classes are then sorted by these scores to derive a combined class ranking for that pattern. This procedure is illustrated in the example shown in Figure 3.6. Ties in the combined ranking may be broken

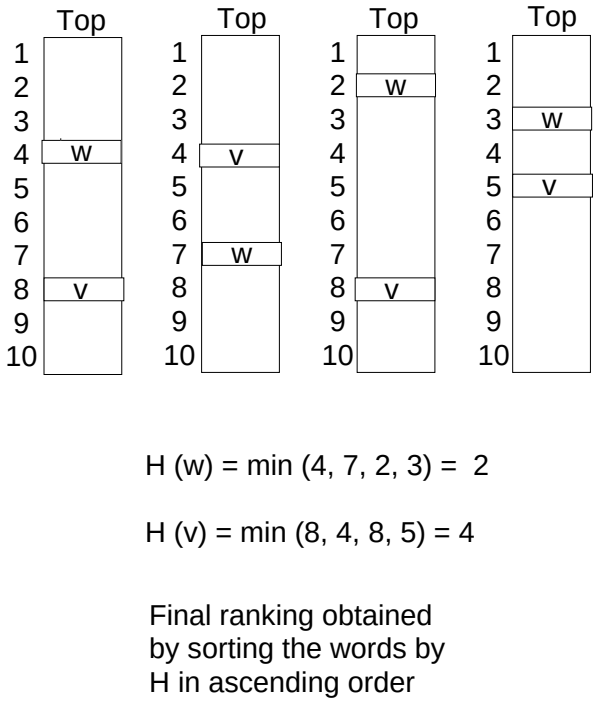


Figure 3.6: Class set reordering by the highest rank method.

arbitrarily to achieve a strict linear ordering.

This reordering method is particularly useful in a problem involving a large number of classes and few classifiers. The advantage is with its ability to focus on the strength of each classifier. For any input pattern, as long as there is one classifier that performs well and ranks the true class near the top, say, at rank k, no matter how the other classifiers perform, the true class will be at a position no farther then $k \times m$ from the top in the combined ranking, where m is the number of classifiers.

For example, in a problem with 10,000 classes and 5 classifiers, the true class of a pattern is ranked at positions 9823, 455, 2, 2008, and 7867 by the 5 classifiers respectively. The highest rank it receives is 2. Since there are 5 rankings, there are at

most 5 other classes that receive a highest rank of 1, and at most 4 other classes that receive a highest rank of 2. Therefore there are at most 9 (5+4) classes ranked above the true class, and the true class must be at or above the 10th position from the top of the combined ranking. This is despite the fact that four out of five classifiers do not perform well on that pattern.

Using this method, a classifier is redundant if the rank it assigns to a true class is always lower than those assigned by other classifiers. In other words, that classifier does not have a specialty which is different from others, and therefore can be ignored.

One disadvantage with this method is that the combined ranking may have many ties. The number of classes sharing the same ranks depends on the number of classifiers used. For example, if five classifiers are used, at least one and at most five distinct classes are ranked at the top. Those five classes are in a tie in the combined ranking, since they all receive the score 1. Therefore, this method is useful only if the number of classifiers is small relative to the number of classes. Otherwise, most of the classes are involved in ties and the final ranking is not interesting. In the next two sections, other methods for ranking combination are described that do not have this problem.

3.4.2 The Borda Count Method

Decision combination in a multiple classifier system can be viewed as a committee voting problem, where the decisions are the class rankings determined by the individual classifiers.

In the context of group decision theory, the mapping from a set of individual rankings to a combined ranking is referred to as a *group consensus function*. One useful group consensus function is referred to as the *Borda count* [13], which is a

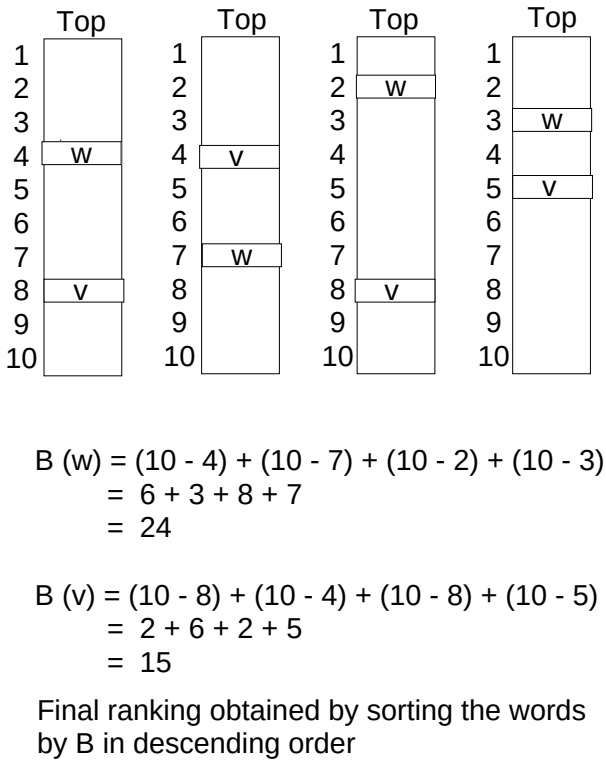


Figure 3.7: Class set reordering by the Borda count method.

generalization of the majority vote. The Borda count for a class is the sum of the number of classes ranked below it by each classifier. The definition is given as follows:

For any class c in a set S, let $B_j(c)$ be the number of classes in S which are ranked below c by classifier C_j $(j = 1, ..., m)$. The Borda count for class c is $B(c) = \sum_{j=1}^{m} B_j(c)$.

The consensus ranking is given by arranging the classes so that their Borda counts are in descending order. An example of ranking combination by the Borda count is given in Figure 3.7.

Intuitively, if a class is ranked near the top by more classifiers, its Borda count tends to be larger. The magnitude of the Borda count for each class measures the strength of agreement by the classifiers on the proposition that the input pattern belongs to that class. The resultant ranking is in the order of strength of such an agreement. For a two-class problem, the Borda count is equivalent to the simple majority vote.

The Borda count function assumes additive independence among the contributions of the individual classifiers. If the rank assigned to a particular class c by a classifier is considered as a marginal similarity between an input pattern and class c, then the global similarity is a linear function of the marginal similarities. This can be compared to the Minkowski metric, which is a function of the sum of the partial distances measured in each dimension of a space.

There are many variations of the Borda count function, including one that handles ties in the rankings [13]. The generalized Borda count is a linear function of the class ranks.

Using the Borda count method, a classifier is redundant if it always reinforces errors made by the others. That is, if all the classes it ranks above a true class are always contained in some other classifier's choices above the true class, then it is redundant and should be removed.

The Borda count method computes a consensus ranking by a measure of agreement among the classifiers. It is simple to implement, and requires no training. However, it does not take into account the differences in the individual classifier capabilities. All classifiers are treated equally, which may not be preferable when we know that certain classifiers are more likely to be correct than the others. A statistical approach that computes a weight for each classifier is described as follows.

3.4.3 The Logistic Regression Method

The Borda count method needs to be modified by assigning weights to the rank scores produced by each classifier, so that the method can be adapted to combine classifiers with imbalanced performances. The weights should reflect the relative significance of each classifier in the combination. Insignificant classifiers should be removed for efficiency. The significance of each classifier should be evaluated in the context of the combination, since certain classifiers may not perform well for all kinds of inputs, but can compensate for some errors made by others that are otherwise irrecoverable.

Moreover, it will be useful to measure the confidence on the combined decisions given by the Borda count method. For example, we wish to estimate the likelihood that the top decision in the combined ranking is correct. Possible measures include a statistic derived from the distribution of sums of a given range of ranks, and the intervals between the computed Borda counts for a given class set [55] [90].

However, we realize that the distribution of rank sums is affected by the correlation between the classifiers, and does not necessarily indicate classification correctness. For instance, if rankings by two identical classifiers are combined, the rank sum for the class that is their common top choice falls on an extreme of the distribution, no matter whether that decision is correct or not. The two effects, namely, classification correctness and classifier correlation, must be treated separately in computing a satisfactory confidence measure. The correct classes should be separated from the incorrect ones in deriving a combination scheme.

An explicit binary label can be used to distinguish the correct classes from the incorrect ones. A binary variable Y_c can be associated to each class c in a given set with respect to a given pattern. Y_c has the value 1 if c is the true class of that pattern, and 0 otherwise. The goal of recognition is therefore to predict the value of Y_c for

each class c with respect to a pattern.

Hence the decision combination problem can be reformulated in the context of regression analysis. The rank scores produced by each classifier are used to predict the value of Y_c for each class c. Consider these rank scores as a set of random variables. A multiple regression function can be used to model their effects on Y_c. Because Y_c is binary, a special form of regression analysis, which involves a logistic response function, is particularly useful in this context [1] [26].

For simplicity in notation, we denote the response variable Y_c by Y, which has a value for each class with respect to each input pattern. $Y = 1$ if that class is the true class of the pattern, and $Y = 0$ otherwise. For a training pattern, the true class is known and therefore each class has a known value of Y. For an unseen pattern, the value of Y for each class has two possible outcomes. Y therefore has a Bernoulli distribution with expected value

$$E(Y) \;=\; 1 \times P(Y = 1) + 0 \times P(Y = 0) = P(Y = 1).$$

Denote the probability $P(Y = 1)$ as $\pi(\mathbf{x})$, where $\mathbf{x} \;=\; (x_1, x_2, ..., x_m)$ represents the rank scores assigned to that class by classifiers $C_1, C_2, ..., C_m$. For convenience in discussion, we assume that x_i has the largest value if the class is ranked at the top by C_i.

Using

$$E(Y^2) \;=\; 1^2 \times \pi(\mathbf{x}) + 0^2 \times [1 - \pi(\mathbf{x})] \;=\; \pi(\mathbf{x}),$$

the variance of Y is then

$$V(Y) \;=\; E(Y^2) - [E(Y)]^2 \;=\; \pi(\mathbf{x})[1 - \pi(\mathbf{x})].$$

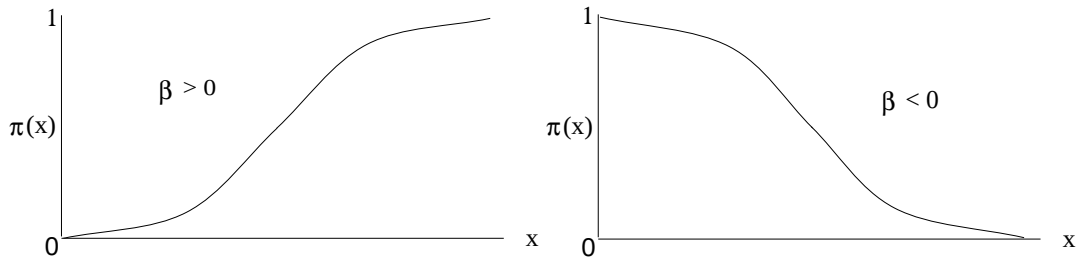


Figure 3.8: Logistic functions.

A linear regression model relating $\mathbf{x}$ and $E(Y)$ is

$$E(Y) \;=\; \pi(\mathbf{x}) \;=\; \alpha + \boldsymbol{\beta}\mathbf{x} \qquad\qquad (**),$$

where $\alpha,\ \boldsymbol{\beta} = (\beta_1, \beta_2, ..., \beta_m)$ are constant parameters.

However, since Y has a Bernoulli distribution, $E(Y) = \pi(\mathbf{x})$ which must fall between 0 and 1. Yet model (**) predicts $\pi < 0$ and $\pi > 1$ for sufficiently small and large $\mathbf{x}$ values. Also, since Y is not normally distributed, and $V(Y)$ depends on $\mathbf{x}$ through $\pi(\mathbf{x})$, the conditions assumed for ordinary least squares estimators are not satisfied.

Intuitively we expect that the likelihood of a class being the true class $P(Y = 1)$ is larger when it is ranked higher by the classifiers. The relationship between the ranks and the tendency towards being a true class is expected to be monotonic, but not necessarily linear. In fact, we expect $\pi(\mathbf{x}) \to 0$ when component values of $\mathbf{x}$ are small, and $\pi(\mathbf{x}) \to 1$ when component values of $\mathbf{x}$ are large.

These conditions on the response function suggest the use of a logistic response function [1]. Figure 3.8 shows the graphs of such a function.

For the case with one independent variable ($m = 1$), the function has the form

$$\pi(x) \;=\; \frac{\exp\ (\alpha \,+\, \beta x)}{1 \,+\, \exp\ (\alpha \,+\, \beta x)},$$

where α, β are constant coefficients.

Using this function as a model, the odds of having response 1 are

$$\frac{\pi(x)}{1 \,-\, \pi(x)} \;=\; exp\ (\alpha \,+\, \beta x).$$

The log odds has a linear relationship with the independent variable x

$$log\ \ \frac{\pi(x)}{1 \,-\, \pi(x)} \;=\; (\alpha \,+\, \beta x),$$

which can be generalized to the multivariate case with m independent variables

$$log\ \ \frac{\pi(\mathbf{x})}{1 \,-\, \pi(\mathbf{x})} \;=\; (\alpha \,+\, \beta_1 x_1 \,+\, \beta_2 x_2 \,+\, \ldots \,+\, \beta_m x_m).$$

A transformation $L(\mathbf{x}) \;=\; log\ \frac{\pi(\mathbf{x})}{1 \,-\, \pi(\mathbf{x})}$ is hence defined. L is referred to as the *log-odds*, or the *logit*, and is linearly related to $\mathbf{x}$.

The logit transformation links the problem to linear regression analysis. Methods based on maximum likelihood or weighted least-squares can be used to estimate the model parameters $\alpha, \beta_1, \beta_2, ..., \beta_m$ [1] [26]. The relative magnitudes of the parameters indicate the relative significances of the classifiers in their marginal contribution to the logit. Hence the parameters can be used as weights for the rank scores.

For each test pattern, the logit for each class is predicted by the estimated model. If a class is ranked at x_1,x_2,...,x_m by classifiers C_1,C_2,...,C_m respectively, its probability of being the true class is estimated by

$$\pi(\mathbf{x}) \;=\; \frac{\exp\ (\alpha \,+\, \beta_1 x_1 \,+\, \beta_2 x_2 \,+\, \ldots \,+\, \beta_m x_m)}{1 \,+\, \exp\ (\alpha \,+\, \beta_1 x_1 \,+\, \beta_2 x_2 \,+\, \ldots \,+\, \beta_m x_m)}$$

$$\;=\; (1 \,+\, \exp^{-1}(\alpha \,+\, \beta_1 x_1 \,+\, \beta_2 x_2 \,+\, \ldots \,+\, \beta_m x_m))^{-1}$$

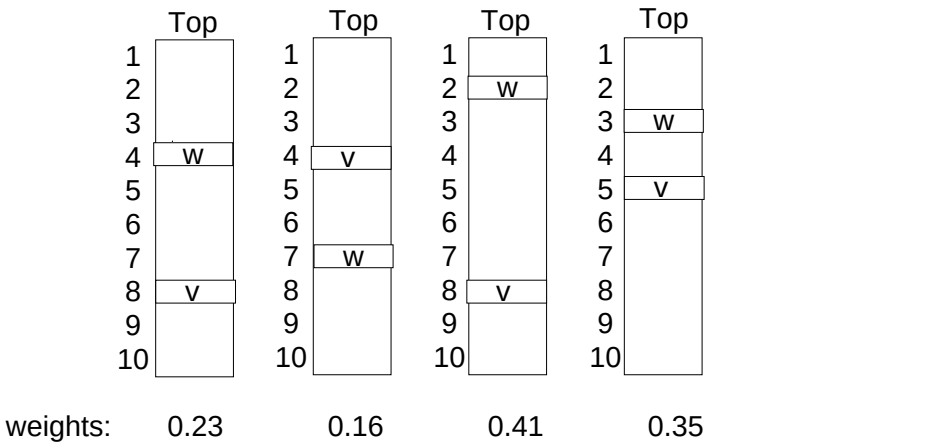


Figure 3.9: Class set reordering by the logistic regression method.

where α and β_i $(i = 1, ..., m)$ are the model parameters.

If only a ranking of the class set is needed, the classes can simply be sorted by the predicted logits in descending order. The class with the largest logit is then considered as most likely to be the true class. This process is illustrated in Figure 3.9.

The values of $\pi(\mathbf{x})$ or the logit can serve as a confidence measure. A threshold on these values can be determined experimentally, so that class choices with confidences lower than the threshold can be rejected.

Example

Weight estimation by logistic regression is illustrated with an example application in word recognition, where two classifiers were used to recognize an image as one of 67,305 classes.

For convenience in implementation, we consider only the top 10 choices from each classifier, and use the largest score to represent a top choice. That is, a class receives a 10 if it is ranked at the top. A class receives a 0 if it is ranked below the 10th position. Using a union method together with three other classifiers, a neighborhood of up to 50 observations were taken for each image. A total of 43,422 observations were obtained using 1,055 training images.

The distribution of the rank scores in these observations and the empirical logits derived from this set of data are shown in Table 3.4. A plot of the empirical logits versus x_1 and x_2 is given in Figure 3.10. A regression plane was fit to these logits by the SAS procedure LOGISTIC [112]. Figures 3.11 and 3.12 show the estimated regression plane and the residuals. The parameter estimates are given in Table 3.5. Both x_1 and x_2 are significant according to the parameter estimates. The estimated regression model is

$$L(\mathbf{x}) \;=\; -5.8557 + 0.1965\ x_1 + 0.4008\ x_2.$$

Remarks. In an ordinary logistic regression analysis, the Chi-square value computed for each model parameter can be used to evaluate the parameter significance. However, caution has to be taken in this application. Because there is only one true class for each pattern, $Y = 1$ for one class implies $Y = 0$ for all the others. In other words, values of the response variable Y are related for each pattern but independent across different patterns. This may lead to the problem of *overdispersion* or underestimates of the standard errors [26] [81]. Nevertheless, the weight estimates are

Table 3.4: Example distribution of rank scores and the empirical logits.

x_1: rank by classifier 1
x_2: rank by classifier 2
N: number of classes receiving (x_1, x_2)
$N_{(Y=1)}$: number of true classes at (x_1, x_2)
$\pi(\mathbf{x})$: $N_{(Y=1)}/N$

x_1	x_2	N	$N_{(Y=1)}$	$\pi(\mathbf{x})$	$log_e(\pi(\mathbf{x})/(1-\pi(\mathbf{x})))$	x_1	x_2	N	$N_{(Y=1)}$	$\pi(\mathbf{x})$	$log_e(\pi(\mathbf{x})/(1-\pi(\mathbf{x})))$
0	0	25145	179	0.007	-4.938	6	0	802	3	0.004	-5.585
0	1	918	6	0.007	-5.024	6	3	21	1	0.048	-2.996
0	2	912	1	0.001	-6.815	6	6	18	1	0.056	-2.833
0	3	889	5	0.006	-5.175	6	8	34	2	0.059	-2.773
0	4	861	5	0.006	-5.143	6	9	44	2	0.045	-3.045
0	5	852	8	0.009	-4.659	6	10	20	9	0.450	-0.201
0	6	832	14	0.017	-4.068	7	0	761	4	0.005	-5.243
0	7	748	16	0.021	-3.823	7	1	14	1	0.071	-2.565
0	8	704	19	0.027	-3.585	7	5	24	1	0.042	-3.135
0	9	601	39	0.065	-2.668	7	6	28	3	0.107	-2.120
0	10	410	94	0.229	-1.212	7	7	32	2	0.062	-2.708
1	2	18	1	0.056	-2.833	7	8	43	1	0.023	-3.738
1	4	12	1	0.083	-2.398	7	9	47	4	0.085	-2.375
1	5	13	2	0.154	-1.705	7	10	41	25	0.610	0.446
1	9	13	1	0.077	-2.485	8	0	707	2	0.003	-5.865
1	10	10	6	0.600	0.405	8	3	19	1	0.053	-2.890
2	0	896	4	0.004	-5.407	8	7	39	1	0.026	-3.638
2	3	13	1	0.077	-2.485	8	8	56	3	0.054	-2.872
2	4	16	1	0.062	-2.708	8	9	56	6	0.107	-2.120
2	7	15	1	0.067	-2.639	8	10	58	33	0.569	0.278
2	9	23	2	0.087	-2.351	9	0	625	7	0.011	-4.481
2	10	10	1	0.100	-2.197	9	2	13	1	0.077	-2.485
3	0	879	2	0.002	-6.083	9	5	30	1	0.033	-3.367
3	3	14	2	0.143	-1.792	9	6	29	4	0.138	-1.833
3	8	16	1	0.062	-2.708	9	8	58	6	0.103	-2.159
3	9	31	3	0.097	-2.234	9	9	132	16	0.121	-1.981
3	10	19	9	0.474	-0.105	9	10	78	48	0.615	0.470
4	0	849	1	0.001	-6.743	10	0	469	6	0.013	-4.346
4	1	21	1	0.048	-2.996	10	1	12	1	0.083	-2.398
4	8	32	3	0.094	-2.269	10	3	12	1	0.083	-2.398
4	9	24	1	0.042	-3.135	10	4	14	1	0.071	-2.565
4	10	15	6	0.400	-0.405	10	5	23	2	0.087	-2.351
5	0	821	1	0.001	-6.709	10	6	20	5	0.250	-1.099
5	2	12	1	0.083	-2.398	10	7	24	6	0.250	-1.099
5	7	40	1	0.025	-3.664	10	8	36	8	0.222	-1.253
5	8	31	4	0.129	-1.910	10	9	63	22	0.349	-0.623
5	9	21	2	0.095	-2.251	10	10	375	331	0.883	2.018
5	10	19	8	0.421	-0.318						

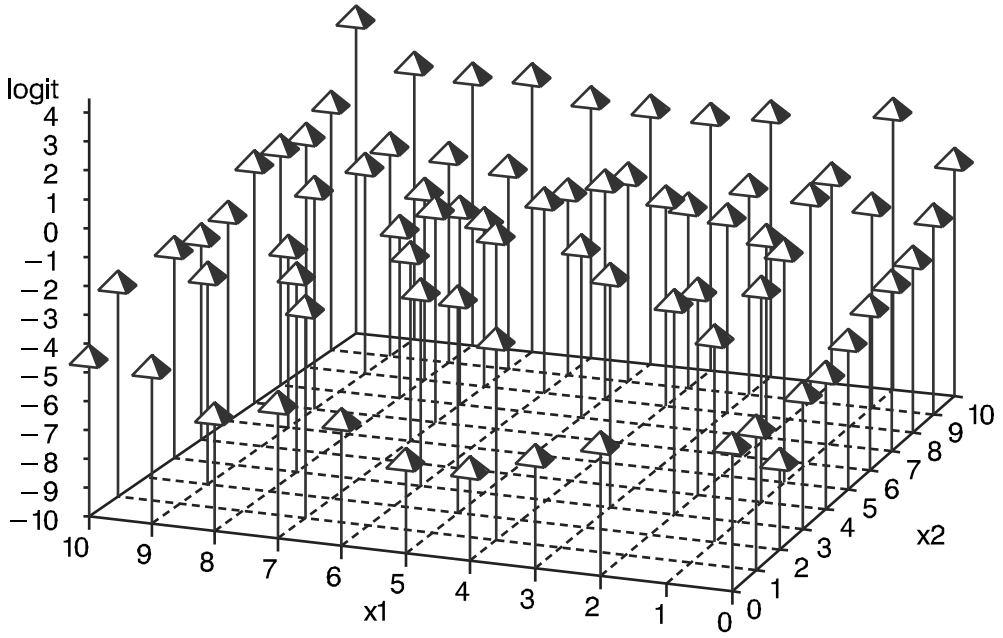


Figure 3.10: Plot of empirical logits versus ranks by two classifiers.

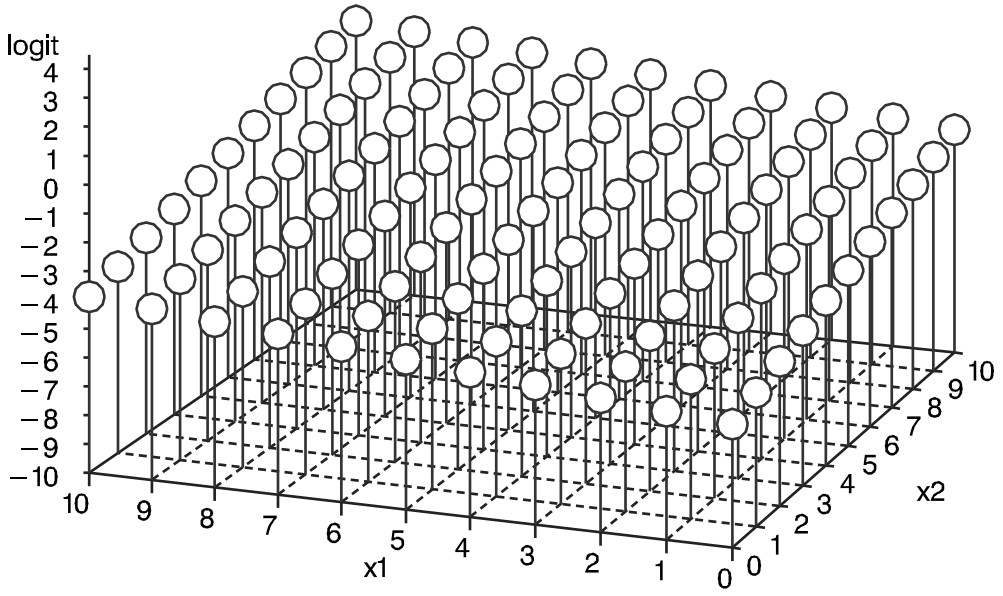
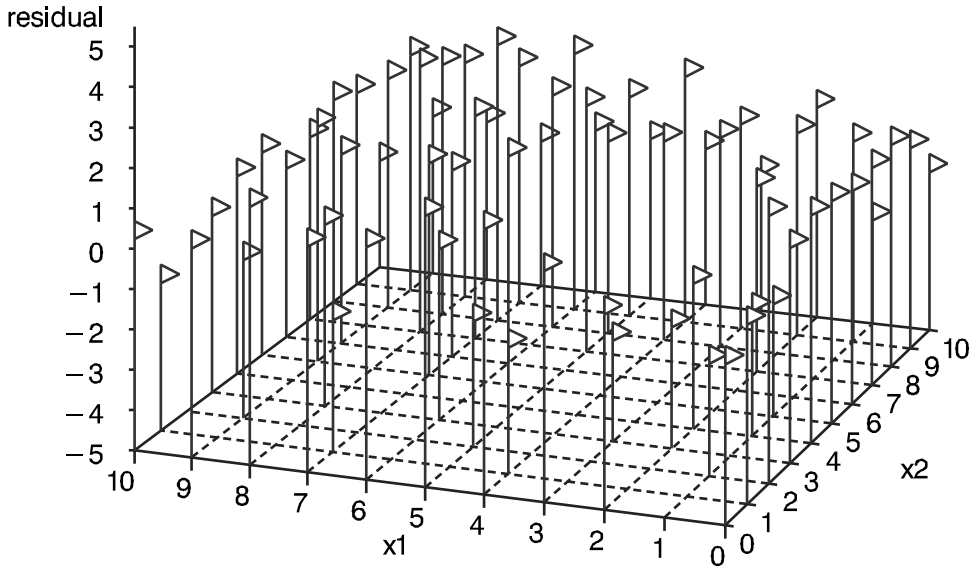
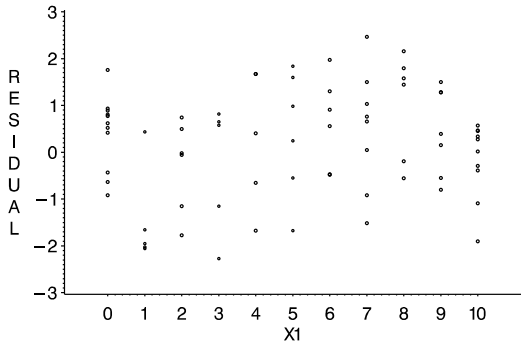


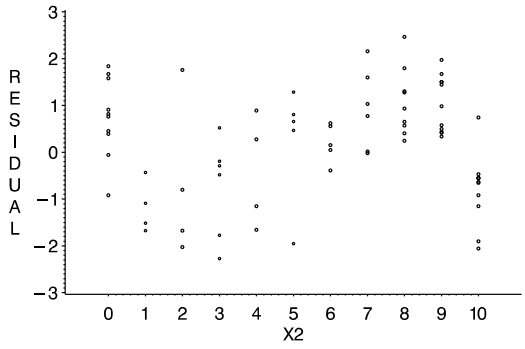
Figure 3.11: Plot of fitted logits versus ranks by two classifiers.



(a) Plot of residual versus x_1 and x_2.



(b) Partial residual for x_1.



(c) Partial residual for x_2.

Figure 3.12: Residual plots.

Table 3.5: Analysis of maximum likelihood estimates in logistic regression.

Variable	Parameter Estimate	Standard Error	Wald Chi-Square	Pr > Chi-Square	Standardized Estimate
INTERCEPT	-5.8557	0.0779	5656.2065	0.0001	.
X1	0.1965	0.00883	495.1244	0.0001	0.297987
X2	0.4008	0.0103	1516.6643	0.0001	0.607888

unaffected. As our concern is to obtain a useful combination scheme, we suggest that the significance of a model be evaluated experimentally by observing its performance on a test set. Nevertheless, relative significance of the classifiers in a combination can be told by the relative magnitude of the parameter estimates.

The residual plot can be used to examine if there is a systematic lack of fit between the estimated values and the actual values of the logits. Goodness of fit largely depends on whether the linearity assumption is satisfied by the empirical logits.

The linearity assumption may be invalid if more ranks instead of a small number of top decisions are used. This is because, for most classifiers, $\pi(\mathbf{x})$ increases much more rapidly near the top of the rankings than it does at lower positions. The surface formed by the empirical logits is likely to be curvilinear on a large rank scale.

This problem can be overcome in three ways. A better fit may be obtained by using a nonlinear regression model, though the model may then become too complicated. A second solution is to truncate the rankings at a certain threshold. The decisions at positions lower than the threshold may simply be grouped together and assigned a single rank, just as we did in the previous example (assigning 0 to all classes ranked below 10). A third solution is to attempt a piecewise linear fit. This is illustrated in the following example.

Example

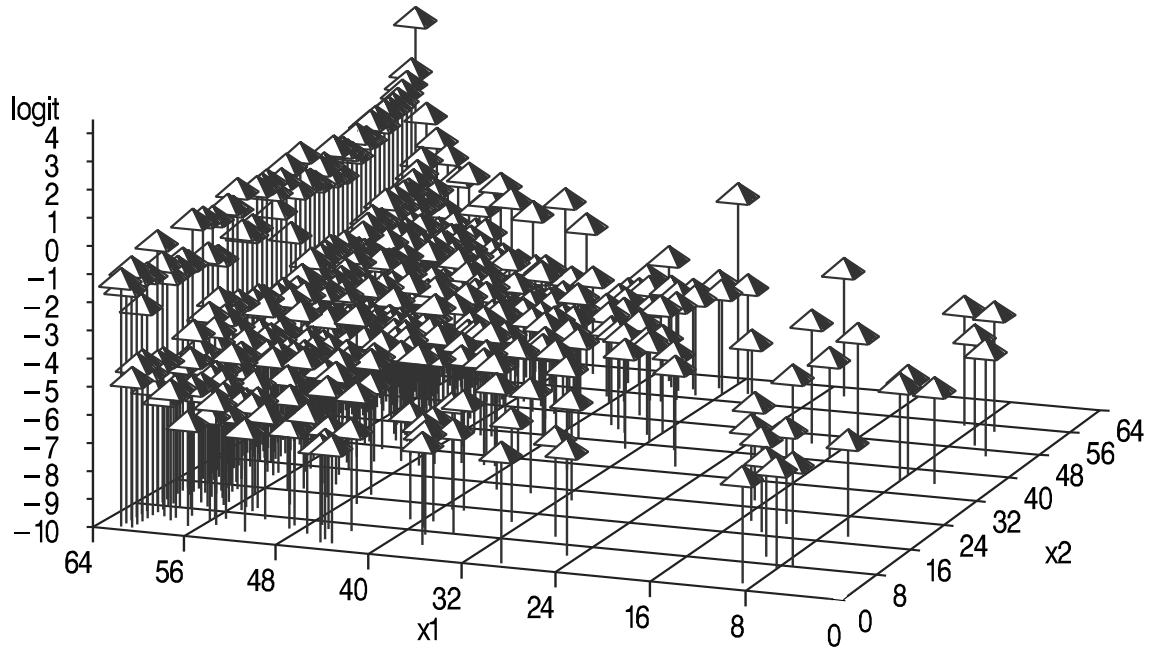


Figure 3.13: Plot of empirical logits at all ranks decided by two classifiers.

Two classifiers were used to classify a set of characters into 62 classes. The rank scores assigned by the two classifiers are denoted by x_1 and x_2 respectively. Both x_1 and x_2 range from 1 to 62. The class decided to be the top choice by either classifier receives a rank score of 62.

The empirical logits were computed for 30,280 training images. Figure 3.13 shows the empirical logits on the full rank scale, where the surface is obviously curvilinear.

The rankings were then truncated and only the top 10 choices were used. Figure 3.14 shows the empirical logits at the top 10 ranks only. A linear regression plane was estimated, which is shown in Figure 3.15. A model estimated by the SAS procedure

GLM is

$$L(\mathbf{x}) \;=\; -47.8197 + 0.5529\; x_1 + 0.2059\; x_2.$$

To attempt a piecewise linear fit, a breakpoint was chosen at the fifth position of each ranking (i.e., $x_1 = 57$ or $x_2 = 57$). Two new variables p_1 and p_2 were introduced, where

$$p_1 \;=\; \begin{cases} 1 & \text{if } x_1 > 57 \\ 0 & \text{otherwise,} \end{cases}$$

$$p_2 \;=\; \begin{cases} 1 & \text{if } x_2 > 57 \\ 0 & \text{otherwise.} \end{cases}$$

The new model estimated by procedure GLM is

$$L(\mathbf{x}, \mathbf{p}) \;=\; -8.95 + 0.03\; x_1 \;+\; 0.89\; (x_1 - 57)\; p_1 \;+\; 0.02\; x_2 \;+\; 0.32\; (x_2 - 57)\; p_2,$$

which is piecewise linear. The logits fitted with this model are shown in Figure 3.16.

Remarks. For quantitative criteria for goodness of fit, diagnostic procedures, and model building techniques, interested readers are referred to [1] [5] [62] [79] [81]. However, due to the dependency among the response values Y with respect to a given pattern, not all of the standard diagnostic procedures are applicable. Nevertheless, as our goal is to improve classification correctness, the effectiveness of a model can always be evaluated by experimental results in decision combination.

Using the same definition of Y, the problem can be reformulated as a two-class discrimination problem and solved by discriminant analysis. However, it has been observed that, if the normality assumption for $\mathbf{x}$ is invalid, which is true in our case, logistic regression is preferred over discriminant analysis [36] [106].

The logistic regression method can be applied to numerical similarity scores as well, since there are no range or unit assumptions for the independent variables.

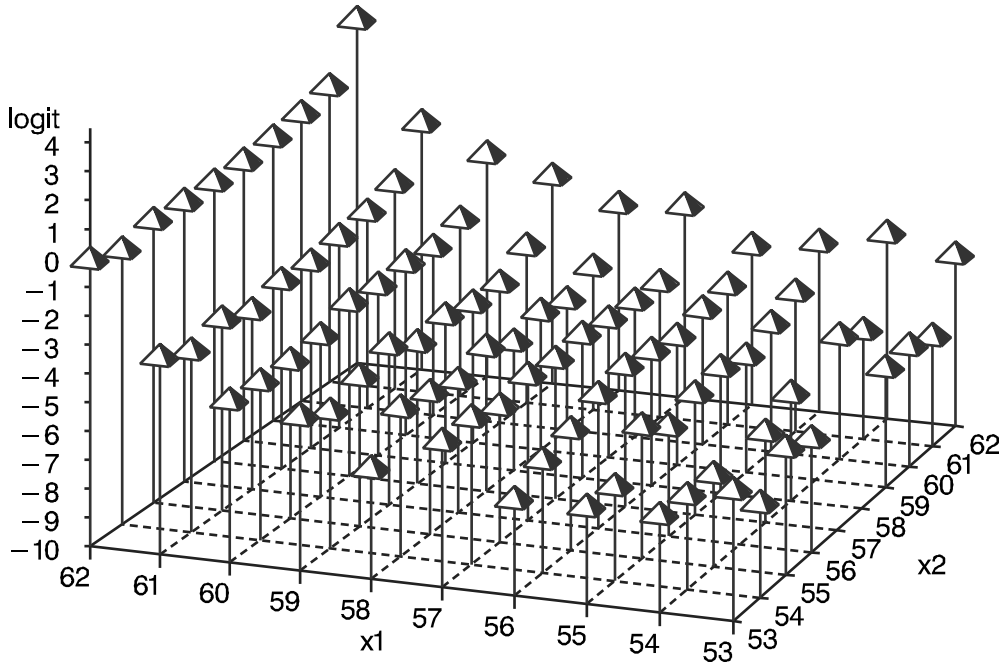


Figure 3.14: Plot of empirical logits at top 10 ranks by two classifiers.

However, rank scores are preferred unless a correlation can be established between the numerical scores and classification correctness (see discussions in Section 3.1.2).

3.4.4 Summary of Class Set Reordering Methods

All these reordering methods use a function defined on the rank scores given by the individual classifiers. Let the ranks assigned to a class by classifiers $C_1, C_2, ..., C_m$ with respect to a pattern be represented as a vector $\mathbf{x} = (x_1, x_2, ..., x_m)$, where x_i is smallest for the top choice. The consensus ranking is obtained by ordering the classes according to the values of a combination function

$$S = f(\mathbf{x}).$$

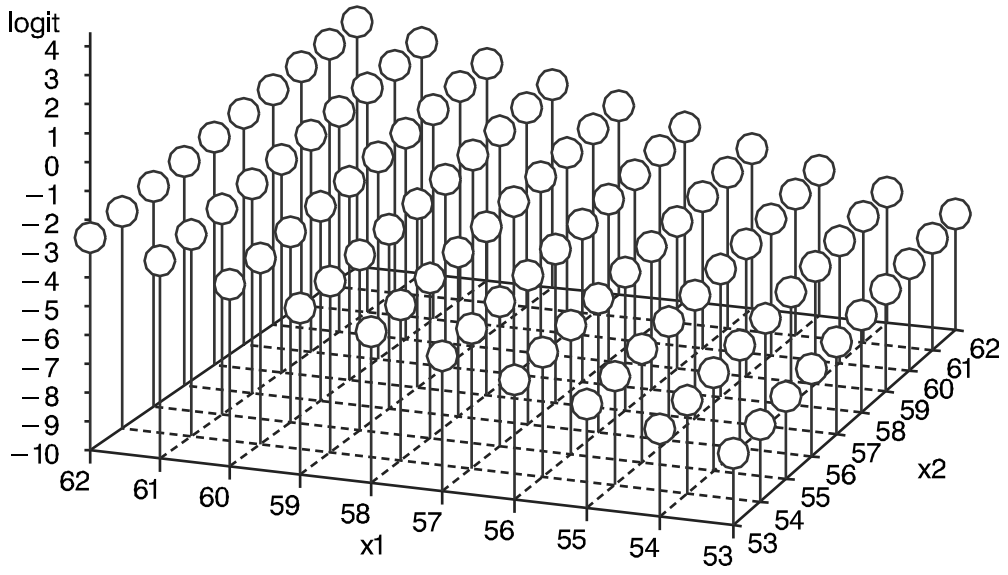


Figure 3.15: Plot of linearly fitted logits at top 10 ranks by two classifiers.

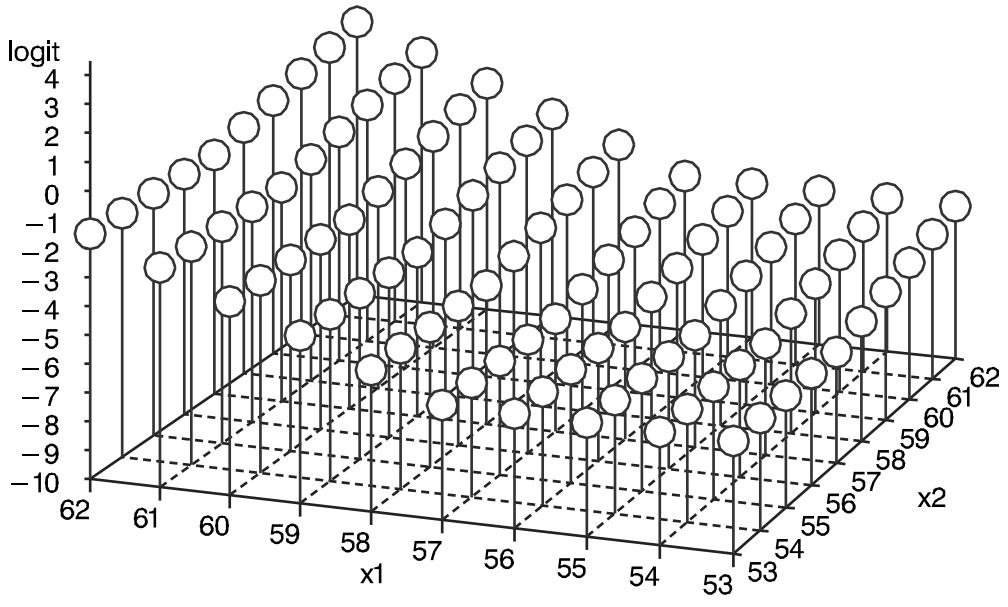


Figure 3.16: Plot of piecewise fitted logits at top 10 ranks by two classifiers.

Three forms of $f(\mathbf{x})$ are proposed:

$$
\begin{aligned}
f_1(\mathbf{x}) &= \min(x_1, x_2, ..., x_m), \\
f_2(\mathbf{x}) &= x_1 + x_2 + ... + x_m, \quad \text{and} \\
f_3(\mathbf{x}) &= w_1 x_1 + w_2 x_2 + ... + w_m x_m,
\end{aligned}
$$

where $w_1, w_2, ..., w_m$ are weight coefficients.

We have seen that f_1 is used in highest rank method, f_2 in the Borda count method, and f_3 in the logistic regression method.

Each of the three combination functions is useful under different circumstances. When it is needed to reduce a large class set to a small neighborhood, and the number of classifiers is small compared to the number of classes, f_1 is preferable. f_2 requires no training. It can be used to measure the agreement among the classifiers when many choices from each classifier are considered. f_3 is good for obtaining a consensus ranking when only a few top choices from each classifier are considered.

The logistic regression method is the most sophisticated among the three, and is the most powerful when all three methods are applicable. However, parameter estimation may be too expensive when more than a few top choices are considered. The weight estimates could also be ineffective if the linearity assumption is invalid.

Like most other statistical techniques, the weights for f_3 need to be estimated using a large training set, so that the distribution of rank scores is sufficiently dense at all levels. Therefore f_3 can be used only when many training samples are available.

It is possible to use all three combination functions in the same system that involves many classes. f_1 can be applied to reduce the given class set to a small neighborhood. f_2 can be applied to derive a combined ranking of classes in the neighborhood. f_3 can be used to derive a combined ranking for the classes ranked

close to the top. This is shown in the following example.

Suppose that m (e.g. $m = 10$) classifiers are applied to rank n classes with respect to a pattern, where n is a large number (e.g. $n = 5,000$). In the training stage, k (e.g., $k = 20$) decisions are taken from the top of each ranking to estimate a set of weights by logistic regression. In the test stage, the combination is performed in three steps. The highest rank method is first used to derive a combined ranking. p decisions are then taken from the top of the combined ranking as a neighborhood, where $p < n$ and $p > mk$ (e.g. $p = 500$). The rank scores assigned by the classifiers are retained for each class included in the neighborhood. In the second step, the classes in the neighborhood that are ranked above k by at least one classifier are combined by the logistic method. Since $p > mk$, all these decisions should have been included in the neighborhood. All other classes in the neighborhood that have not been combined in the second step may then be combined by Borda count.

In this way, the final ranking is a concatenation of three combined rankings: the combination by regression at the top, followed by the combination by Borda count, and then the combination by highest rank. This is useful if a complete ranking of all the classes is desired. However, in most practical applications, only a few top choices are needed from the final ranking, so that one method among the three may be sufficient.

3.5 Dynamic Selection of Classifiers

Consider an *oracle* that will always predict the best classifier for each pattern. If such an oracle is available, we can take the decisions only from the selected classifier, and ignore those by others. This is the ideal case of *dynamic classifier selection.*

The decision of such an oracle could be based on confidence of feature detection,

or the correlation of classifier performance with measurable characteristics of the pattern, or estimation of the type of degradation in a pattern. Dynamic selection can be applied on a set of classifiers that are statically determined to be useful when all possible cases are considered.

One way to approximate such an oracle is to specify a set of mutually exclusive conditions that divides a training set into several partitions. Classifier performance is measured separately on each partition. The best classifier for each partition is hence determined. For each test pattern, the best classifier for its type can be selected by evaluating the conditions.

Dynamic selection can also be applied at the decision combination level. After the training set has been partitioned by a computable condition, the significance of each classifier's contribution can be estimated using the logistic model. The estimated model gives the decision combination function for the type of inputs represented by that partition. After the model is estimated for each partition, a suitable decision combination function can be selected dynamically for each test case.

One method to do this is given as follows. Suppose that a set of classifiers that make uncorrelated errors are available. The quality of an input pattern can then be characterized by examining the class rankings produced by these classifiers. Intuitively speaking, the classifiers tend to agree on the top choice for patterns that are easy to recognize, and tend to disagree for difficult cases. Therefore, whether the top choices are the same is a hint to the difficulty in recognizing a particular input. A training set can be partitioned according to the state of agreement by the classifiers on the top choices. A regression model can then be estimated separately for each partition. In test runs, a combination function can be dynamically selected according to the same states of agreement on the top choices. In Section 5.4 of Chapter 5, we will describe one such experiment and show that this method is effective.

An important point to notice in the design of a dynamic selection procedure is that the conditions must be cheaply computable from the inputs. The importance can be seen if this selection is viewed as an intermediate method between two extremes, one using a static single classifier and the other using one classifier for each class that responds well for patterns of only that class. In the latter case, the partitioning condition is the true class identity of the input, and hence the selection of the best classifier is equivalent to the original recognition problem in difficulty.

3.6 Multiple Stage Organization

A number of classifiers and their combination functions may be organized in many different ways. One possibility is that all the classifiers are connected in parallel, and their decisions are combined by one or several methods applied serially. Alternatively, the classifiers may be organized in groups, with a combination function applied to each group; and the combined decisions from each group are recombined later to derive a final decision. Certain classifiers may be combined using the reduction methods, so that other classifiers can be applied later to the reduced set of classes. There are even more possibilities if dynamic selection is applied.

In general, the classifiers can be organized in a multiple stage structure. At each stage, a group of classifiers operate in parallel and their decisions are combined by any one of the methods proposed in our theory. A dynamic selector decides which classifiers are to be activated at each stage. The set of classes is then gradually reduced and reordered as it goes through each stage.

The optimal design of such a multi-stage organization is specific to a particular application. Factors to consider include the performance of each classifier, the correctness of the combined rankings given by each method, and costs of errors. An analytic model of all the factors involved is very difficult, if not impossible. Nevertheless,

the best criterion for the evaluation of alternative organizations is their experimental performance, since the recognition of an input pattern is our eventual goal.

3.7 Alternative Approaches

The methods for decision combination discussed in this chapter are based only on rankings. These methods are applicable regardless of what similarity scores are used by the individual classifiers. A unique advantage of using these methods for decision combination is that, in constructing a solution to a recognition algorithm, one has flexibility in selecting the best descriptors and similarity functions for each type of features, as long as a ranking of the class set can be generated.

There are uses of rankings in other contexts. For instance, the theory of multidimensional scaling and unfolding [109] attempts to recover distance information from a set of ranking data, and hence is useful to obtain a combined similarity score from a set of rankings. However, it is unclear how to obtain a consistent set of scaling parameters for different inputs.

Some other combination functions are useful in special types of multiple classifier systems. For systems where all the classifier decisions are binary, simple voting methods such as the majority vote may be satisfactory. In cases when reasonable and consistent confidence measures can be assigned to the decisions, heuristic functions or theories for confidence combination may be applicable. Other alternatives for decision combination include custom-designed decision trees, neural network approaches as well as symbolic inference systems.

Chapter 4

A Computational Model for Visual Word Recognition

4.1 The Word Recognition Problem

The visual word recognition problem is defined as follows: given an image of a word and a lexicon containing the word, produce a ranking of the lexicon such that the word in the image is ranked as close to the top as possible. A perfect solution is one that always assigns the top rank to the true word, and lower ranks to all other words in the lexicon.

Our objective is to construct a computational model that approximates human recognition performance. Though certain hints from psychological studies are used, the model is not intended to simulate in precise detail the human vision process.

The difficulty of word recognition is caused by the stylistic variations of the appearance of a word and the degradation of the image quality introduced in the image production and capturing processes. The size of the lexicon also has a major effect on recognition correctness. Therefore we constrain the problem in these aspects in order to obtain an initial solution. The constraints are discussed in Section 4.2. We will discuss the limitations of traditional approaches in Section 4.3, and our ideas to overcome them in Section 4.4. These ideas are organized in a computational model, which will be presented in Section 4.5 to 4.10.

4.2 Constraints on the Problem Domain

We restrict our study to domains with two constraints:

1. The images are machine-printed words in ordinary fonts of the English alphabet.
2. The word in the image is contained in a known lexicon of size limited to 100,000 words.

The implications of these two constraints are as follows.

Machine-printed words in ordinary fonts. Using machine-printed images reduces the complexity of the problem in the following aspects:

1. The variation in the styles of machine-printed words is relatively small as compared to handwritten words.
2. The alignment of the letters with respect to the reference lines is strict. This property makes it much easier to obtain a fixed coordinate frame that describes the location of the shape features.
3. The shape features of a word are a composition of the shape features of the characters in the word. This property guarantees that the prototypes for a word can be easily synthesized from the prototypes of characters in the same font. Hence a limited collection of font samples is sufficient to generate word images needed for training purpose.

Contrary to a popular perception, the problem of recognizing machine-printed words in nonspecific font types is far from being solved. Most of the successful systems to date work only with a small number of font types. For instance, if the input images are printed in a fixed font, such as the American Bankers Association

E-13B font (Figure 4.1), they can be recognized easily by simple algorithms. The complexity grows as more font styles are considered, as the variability in shape is much larger in a large collection of fonts. In fact, it is not easy even for human beings to extract essential shape features of a character in some decorative fonts (Figure 4.2).

We do not restrict the input images to be in special fonts used in some domains such as those shown in Figure 4.1. The algorithms are intended to recognize words printed in ordinary fonts used in daily communication. To keep the problem within a reasonable scope, we ignore decorative fonts such as those seen in Figure 4.2. Attempts to recognize characters in decorative fonts are discussed in [61].

For simplicity, we assume that an input image is binarized, and is represented by a two-dimensional array of 0 or 1 entries. Gray scale images can be binarized by a thresholding algorithm [23].

Words in a known lexicon of limited size. The assumption that the word is contained in a given lexicon confines the problem in the area of pattern recognition, so that the word can be identified based solely on its visual form. No other inferences such as abbreviation expansion are needed to recover the word identity.

A lexicon provides many contextual hints that facilitate recognition. For example, consider a word with six characters. Without a lexicon, such a word would be one



Figure 4.1: Example characters in the American Bankers Association E-13B font.

of $26^6 = 308,915,776$ possible strings. Even a large, 1,000,000 word lexicon contains only 0.3% of all the possible strings, and hence contains many constraints that can be used. For example, if there are no words where the character 'p' is followed by the character 'q', the 'pq' combination can be eliminated from character pair decisions.

The size of the lexicon is directly related to the complexity of the problem. A larger lexicon requires that the features have greater discriminating power, and a finer similarity score is need to evaluate the matches. This can be illustrated by an example. Suppose that the lexicon contains only two words, "Pennsylvania" and "Ohio", and the input image is "Ohio". In this case a rough estimation on the word length will be sufficient to determine which of the two words is in the input image. More features are needed if the lexicon contains words of similar lengths.

Small lexicons are obtainable in some restricted domains, such as in interpretation of postal addresses. Our target lexicon size is in the order of 10,000 to 100,000 words, which is comparable to the size of a general domain vocabulary.

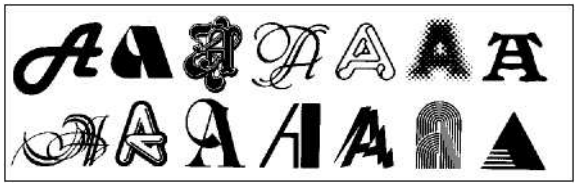


Figure 4.2: Example characters in decorative fonts.
[Source: D.R. Hofstadter, *Metamagical Themas: Questing for the Essence of Mind and Pattern*, Basic Books, Inc., 1985, page 243.]

4.3 Limitations of Traditional Approaches

As mentioned in Chapter 2, most existing word recognition algorithms perform their tasks in a three-step process: segment the word into characters, recognize the characters, and postprocess the character decisions. Dividing a word into characters simplifies the recognition problem for three reasons: characters are easily normalized to a fixed size for shape description; they are simple in shape when compared to the whole word; and the number of character classes is fixed and small, that usually equals to the size of the alphabet. However, there are several problems with this approach that keep it from being a perfect solution to recognizing noisy words in multiple fonts:

1. *Failures at the segmentation stage:* Character segmentation is difficult for degraded images, especially those in proportional pitch fonts (Figure 4.3(a)). Errors committed in segmentation affect all the following recognition decisions. To avoid this problem, some special purpose systems require the characters to be placed in preprinted boxes. Some systems simply reject any character-like segments larger than a predefined size. The effectiveness of such systems is therefore limited.

2. *Failures at the recognition stage:* Even if character segmentation is correct, because of noise present in the image, the appearance of some characters could be so distorted that the recognition decisions are premature without the word context coming into effect (Figure 4.3(b)). These errors are often irrecoverable at a later stage.

3. *Failures at the postprocessing stage:* Postprocessing algorithms are typically based on heuristic models of common recognition errors and text characteristics, or at best based on their probabilities. Accurate models are often difficult to

obtain, and the collective statistics might not be sufficient to guarantee correct decisions.

In Chapter 2 we have identified that, in addition to the character-based approach, there are other solutions to the word recognition problem, including the segmentation-based approach, and word shape analysis. Since each of these approaches excels for a limited type of images, their performance and generality have been restricted when applied in isolation. A particular method may effectively recognize text in a given, well-specified domain, but might fail poorly when presented with another type of text. This suggests that a robust solution to word recognition might need to use all the three approaches simultaneously, so that they can complement each other in correctness.

4.4 Integration of Three Approaches

The three approaches differ from each other to a large extent. In each approach, an input word image is transformed to a feature descriptor. Matches between these

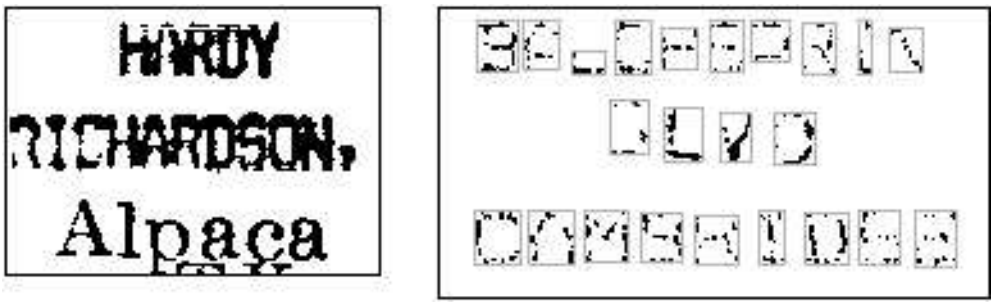


Figure 4.3: Examples of (a) word images difficult to segment, and (b) images with segmented characters that are difficult to recognize.

feature descriptors and the class prototypes are evaluated with various similarity scores. In some methods, the similarity score is defined as a simple function of the feature values, such as the sum of differences between corresponding components of two feature vectors. In other methods, the similarity score is defined by a sophisticated evaluation algorithm that cannot be written as a close form function.

These methods are therefore easier to be integrated using a multiple classifier system. The contributions of the component classifiers are combined after initial class decisions are made. The problem of mixing up feature measurements on various scales is thus avoided. The initial decisions can be represented as rankings, such that irrelevant details in the similarity scores are ignored.

Based on these ideas, a computational model is proposed as a robust solution to visual word recognition. The model integrates the three approaches to word recognition, such that the weakness of one method may be complemented by the strength of another. Cooperation of the individual methods is achieved by a decision combination function.

4.5 A Computational Model for Word Recognition

The proposed model for word recognition consists of an activation control mechanism, a set of parallel classifiers, and a decision combination mechanism. The control mechanism uses information from the input image and the given lexicon. It activates candidate words in the lexicon based on a set of global features computed from the image, and then selectively applies suitable classifiers to the image. The parallel classifiers take three different approaches (character-based, segmentation-based, and word shape based) to recognize the word. Each computes a ranking of the activated words according to their similarities to the image. The rankings are then combined by the decision combination mechanism and a consensus ranking is output (Figure

4.4).

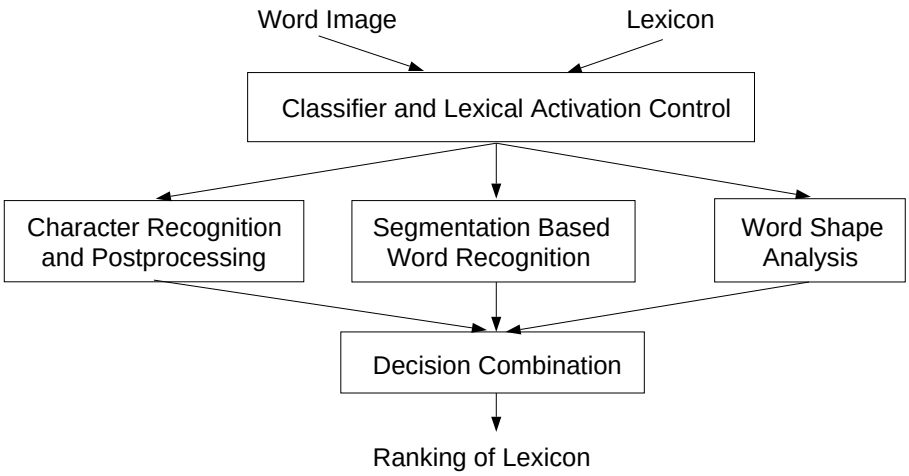


Figure 4.4: A computational model for word recognition using multiple approaches.

The classifiers use a continuum of scopes of shape and context to recognize a word. At one extreme, the character-based approach first recognizes individual characters as local units, and then modifies the decisions by lexical constraints. At the other extreme, the word shape analysis approach recognizes a word as a whole unit, and uses contextual hints directly in feature description and matching. Thus, these methods operate on multiple scales, ranging from isolated characters to whole words, and thereby introduce a redundancy that is needed to tolerate the possible variations in a word's appearance.

The control strategy uses both bottom-up and top-down information to selectively activate suitable classifiers and words in a given lexicon.

Bottom-up information is derived from a preliminary analysis of the word image. For example, if it is determined that the image is printed in a constant-pitch font

and can be reliably segmented, then high confidence can be placed in the character recognition decisions, and there is no need to activate word shape based classifiers. Bottom-up information also includes global shape descriptions computed from the image, for example, estimates of the word length and word case. The control strategy selectively activates the words in the lexicon that match these descriptions.

Top-down information includes any high level information from a global context in the text. An example of such high level information is the syntactic category to which the word belongs. If the word is considered to be a proper noun, then the word is assumed to be in upper or mixed case, and the words in the lexicon will be converted only to these cases.

When more than one classifiers are activated, multiple rankings of a lexicon are produced. These rankings are input to a combination function, which derives a consensus ranking.

Sections 4.6 to 4.8 discuss the problems and solutions in each of the three word recognition approaches. Emphasis is placed on character decision postprocessing, as well as on word shape analysis. Section 4.9 describes the control strategy. Section 4.10 discusses decision combination.

4.6 Character Recognition and Contextual Postprocessing

The character-based approach is illustrated in Figure 4.5. A segmentation algorithm is first used to separate a word image into a set of character images. A recognition algorithm is then applied to identify each individual character. The character decisions are then post-processed using a lexicon. Contextual knowledge is used only at this stage. A ranking of the lexicon is derived by the postprocessing algorithm.

Since recognition is focused on the shapes of the isolated characters, the in-

formation processed at the recognition stage is limited to a single character. The number of classes to be discriminated are limited to the size of the alphabet. These are advantages if the shape of a character is sufficient for classification, and the algorithm is efficient in such cases. Therefore these methods are suitable when a reliable segmentation can be obtained, and the extracted characters are not deformed by size normalization or other procedures that prepare a character for recognition. They are also useful to recognize well-segmented short words where little word level context can be used. Figure 4.6 shows some images better recognized by this method.

Problems arise when the local shape information within the scope of a character is insufficient for classifying that character. This happens when there are serious errors in segmentation, or when the image quality is so degraded that the characters are severely distorted in shape. These failures are independent of the effectiveness of the character recognition algorithms, since the problems are with the inputs. Even

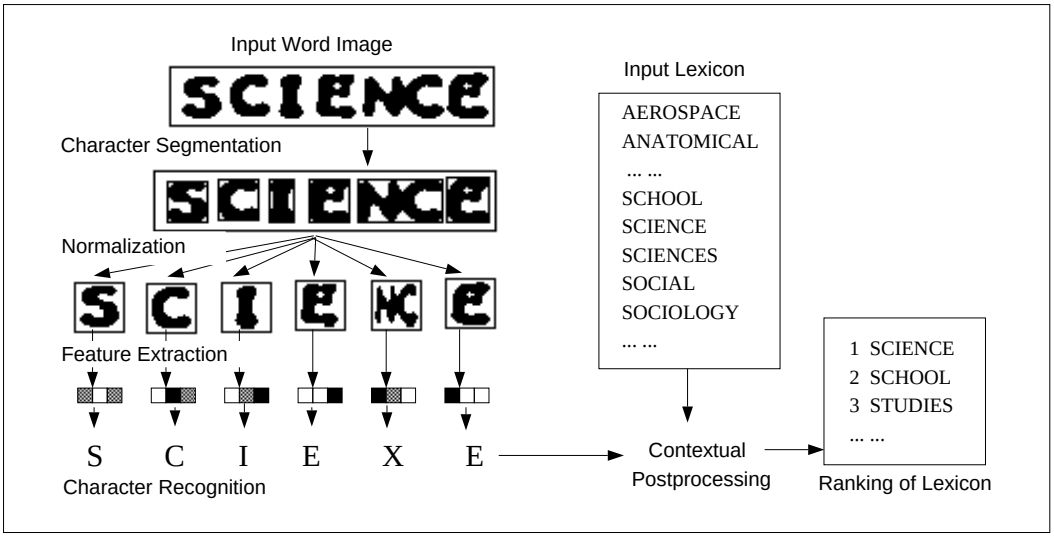


Figure 4.5: The character-based approach.

a classifier comparable to human performance might be confused by those inputs. When this occurs, the class identity of a character is better determined when the shapes of other characters in the word are taken into consideration.

Certain but not all ambiguities in the decisions on the character classes can be resolved by a contextual postprocessing algorithm. After the recognition stage, the shape information has all been converted to class labels. The class decision suggests only that the character is in a neighborhood of a standard shape of the decided class. Subtle information about how much the character deviates from that standard shape is forever lost. In some cases, this loss could be so severe that the true word cannot be recovered. An example of such a case is presented in Figure 4.7.

The difficulties at each stage in the character-based method, as well as their solutions, are discussed in Sections 4.6.1 to 4.6.4. Section 4.6.1 is devoted to problems in character segmentation. In Section 4.6.2, a modified k-nearest-neighbor classifier is proposed for character recognition. Section 4.6.3 describes a use of multiple character recognition algorithms. Section 4.6.4 discusses two types of methods for postprocessing character decisions, one for decisions from a single classifier, and the other for decisions from a multiplicity of classifiers.



Figure 4.6: Word images better recognized by character-based methods.

4.6.1 Character Segmentation

The primary source of errors in character-based methods is in character segmentation. A misplaced segmentation point may lead to substantial distortion in the character shapes. This problem was realized early. As commented by Nagy, "object isolation is all too often ignored in laboratory studies. Yet touching characters are responsible for the majority of errors in the automatic reading of both machine-printed and hand-printed text ... ." [95]

Most segmentation algorithms attempt to identify features that define the boundary of a character. The features are typically obtained from the vertical projection profiles, the outline contour of the characters, and the characteristics of the components that are characters or parts of them. Segmentation points are determined using heuristic rules that relate these features to the positions of the characters. Some methods include an estimation of the pitch and character sizes. If the word is deter-

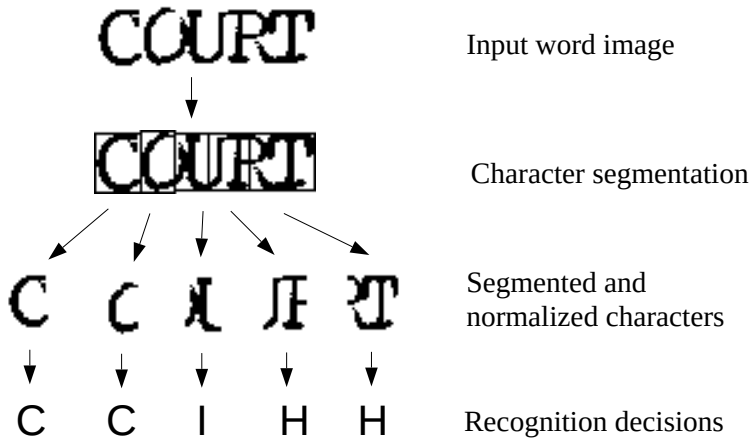


Figure 4.7: Errors in character recognition that are irrecoverable in contextual postprocessing.

mined to be in a constant pitch font, segmentation points are placed at the locations determined by the pitch.

The aspect ratio is often used to help in deciding whether a component contains an isolated character or several merged characters. This is not always successful, since in many fonts the widest characters such as 'w' and 'm' can be of twice or even thrice the width of the narrowest characters such as 'i' and 'l' (Figure 4.8). Therefore the number of characters estimated using the aspect ratios is unlikely to be reliable.

An analysis of the connected components provides good hints for segmenting well-printed words. Except for lower case 'i' and 'j', all characters in the English alphabet consist of a single component. It is also easy to handle the exceptions by a few heuristic rules in the algorithm. However, this simple approach fails when some characters in the document touch their neighbors and some other characters are broken into fragments. The fragmentation problem is particularly serious for word images printed in a dot-matrix font. Though, this problem can sometimes be solved by applying a smearing operation. (Figure 4.9).

Peaks and valleys in the vertical projection profile have been identified as important cues for character segmentation. In a document where all the characters



Figure 4.8: A problem in using aspect ratios: the width of the widest character is greater than the combined widths of two or three smaller characters in the same font.

are perfectly printed, valleys in the projection profiles are good candidates for cut points (Figure 4.10). It is also suggested that spacings between the peaks be used as the sectioning criteria [60]. However, one problem with using peaks is to determine whether the character has one, two or three peaks. This is further complicated by false or missing peaks caused by image degradation. These problems are illustrated in Figures 4.11. Overlapping characters also create problems, but these are rarely seen in ordinary fonts.

As in the examples shown in Figure 4.12, proper segmentation of certain shapes is context sensitive in nature, so that any segmentation method that does not use the context is bound to fail. A better way to handle such ambiguities is to produce multiple guesses and pass them on to the recognition stage, or to invoke recognition algorithms and obtain feedback in the process of segmentation [20] [141].

Most practical segmentation algorithms use a combination of these methods together with some heuristic rules to handle exception cases. Our experiences suggest that, instead of a simple robust solution for all possible cases, a divide-and-conquer strategy seems preferable. Such a strategy involves characterizing the degradation patterns into several categories by some image quality measures, such as pixel density, connectedness, and variations in component sizes, and applying methods that are



Figure 4.9: Two images printed in dot matrix fonts before and after smearing. Many small components are connected after the operation.

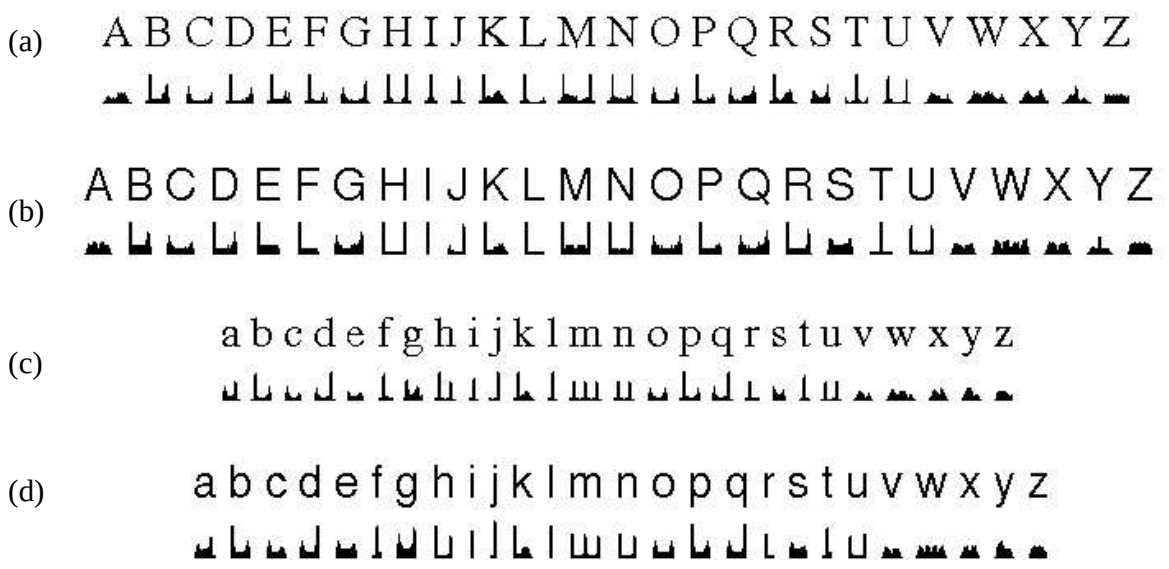


Figure 4.10: Vertical projection profiles of characters in serif and sans serif fonts : (a) Times Roman, upper case; (b) Helvetica, upper case; (c) Times Roman, lower case; (d) Helvetica, lower case.



Figure 4.11: Vertical projection profiles of an ideal image and a faint image. Many peaks are missing in the projection profile of the faint image.

suitable for each category. Context dependent ambiguities are better resolved by generating multiple guesses to be evaluated by feedbacks from recognition.

4.6.2 A Modified K-Nearest-Neighbor Classifier

In Section 2.2.1, features and classifiers commonly used for character recognition have been reviewed. Their effectiveness depends on the condition of a particular application, such as the number of font types, and the quality of the images.

The nearest-neighbor classifiers [33] are among those that are easiest to construct. It is applicable whenever a training set is available and a distance between the feature descriptors is defined. A variant called the k-nearest-neighbor classifier [33] is also useful under such conditions.

With a simple modification, the nearest-neighbor classifier can produce a ranking of the class set. This is done by defining a similarity score for each class to be the distance between the closest sample of that class to the input pattern. This method resembles the single linkage clustering method [33] in that for each class, only the sample that is closest to the input is considered.

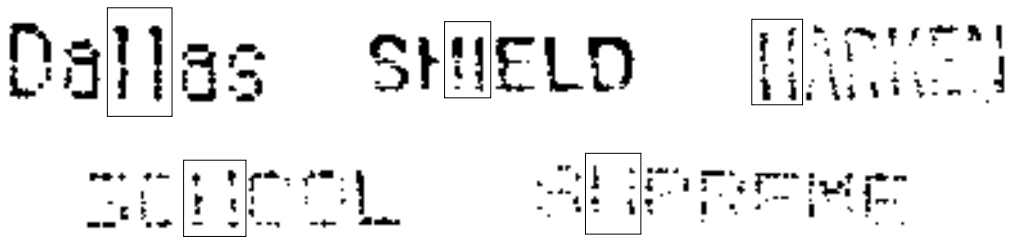


Figure 4.12: Similar shapes appearing in different contexts. The shapes shown in the boxes look alike, even though they are parts of different characters in each word.

However, classification by this method is sensitive to outlying samples. To overcome this problem, we redefine the similarity score for each class on k nearest neighbors instead of on a single nearest neighbor. An algorithm to implement this is described in Figure 4.13. The algorithm is called a *modified k-nearest-neighbor classifier*. The *average distance* from the input pattern to *k closest samples of the same class*, instead of the distance to only one closest sample, is used to evaluate the match between an input and a class. This procedure reduces the effect of single outlying samples. In our experiments, it is observed that the results with $k = 2$ are the most satisfactory.

4.6.3 Character Recognition by Multiple Algorithms

Most character recognition algorithms are optimized in the correct rate at the top choice or within a small substitution set. An alternative goal in constructing a char-

---

Let an input pattern be denoted by I, the number of classes be m, the number of samples in each class be n_j $(j = 1, 2, ..., m)$, and the samples be denoted by S_{ij} $(i = 1, 2, ..., n_j)$. Let $d(x, y)$ be the distance between two patterns x and y. Select a constant k $(1 \leq k \leq min(n_1, n_2, ..., n_m))$.

1. For $j = 1$ to m (each class)
   For $i = 1$ to n_j (each sample)
   compute $d_{ij} = d(I, S_{ij})$

2. For $j = 1$ to m (each class)
   sort $(d_{1j}, d_{2j}, d_{3j}, ..., d_{ij})$ in ascending order
   compute sum_j = the sum of first k elements of the sorted array
   compute $D_j = sum_j/k$

3. Rank class j in ascending order of D_j.

Figure 4.13: A modified k-nearest-neighbor classifier.

---

acter recognizer is to achieve maximum reliability in its decisions, even at the price of lowering the recognition rate. For classifiers that output confidence scores together with the class decisions, this is typically done by putting a high threshold on the confidence score. If the score is lower then the threshold, the decision is not used and the input is rejected. However, it is not always easy to define an effective confidence score that can be used for this purpose.

An external confidence measure defined on the agreement of several character classifiers is a useful alternative. This is most effective when errors made by the classifiers are uncorrelated. The confidence score for a class is the number of classifiers that output it as the top choice. In empirical studies, it is often observed that the decisions agreed on by several independent classifiers are highly reliable. A few exceptions are I's or l's which, even though agreed on by many classifiers, could be fragments of some split characters. A few heuristic rules can be written to ignore these decisions.

An advantage of this method is that no *ad hoc* definition of a confidence score is needed. However, one drawback with this method is in speed. The run time is a multiple of those needed for single classifier solutions.

4.6.4 Character Decision Postprocessing

The conventional approach uses a single character recognizer whose decisions are postprocessed using a lexicon. However, each of the prior postprocessing methods has some drawbacks. The n-gram techniques do not give a ranking of the valid words. The Markov models and the Viterbi algorithm are not easily adaptable to process decisions that may contain minor segmentation errors. The effectiveness of the string editing methods depends on the edit costs that need to be manually adjusted.

Based on observations of the results of character segmentation and recognition, we propose a heuristic postprocessing method. The method takes advantages of the success in character segmentation and recognition in most cases, and can tolerate minor errors produced by either of these procedures.

Postprocessing Decisions From A Single Character Recognizer

Many techniques can segment a majority of word images into a proper number of characters with at most one error (i.e. plus or minus one from the true word length). Therefore a matching algorithm that tolerates such an error level is useful. Similarly, most recognition algorithms are able to include the correct character class in the top two choices. A matching algorithm may place special emphasis on the top two choices, while not excluding the possibility that they are wrong. A heuristic string matching algorithm is developed based on these observations.

The algorithm takes top two class choices for each character in a word. It first constructs a string by using all the top choices, and then makes several more strings by replacing each top choice in turn with the corresponding second choice. The strings are matched to the words in the lexicon that are of the same length, and those of one character longer or shorter. The matches are graded by the number of common characters in the two strings. A penalty is added to the matches that involve a second choice. In matching words of unequal lengths, a character is removed from each position of the longer string in turn. A penalty is added to degrade the matches of words of unequal lengths. The overall score for a word is the highest of the scores it receives from matches with all these strings. The lexicon is then ranked by the scores.

Postprocessing Decisions From Multiple Character Recognizers

As discussed in Section 4.6.3, more than one recognizer may be applied to a character image. Reliability in recognition can be improved by using multiple recognizers and obtaining their agreement. Valuable information is contained in the reliable decisions, which may be used in contextual postprocessing. We new describe a method that uses only the reliable decisions and ignores the others.

An example technique is given as follows. Assume that a word image is segmented into isolated characters, which are then normalized to a fixed size. Six classifiers are applied to the normalized characters. Four of them are nearest-neighbor classifiers using different features. The other two are Bayesian classifiers with the assumption that the features are independent of each other.

The agreement of the six classifiers is measured to obtain a reliability score. For each character, the top choice given by each classifier is taken. If all six classifiers agree on a common top choice, that class is considered to be most reliable and assigned a score of 6. A score of 5 is assigned to a class if five classifiers agree, and 4 if four agree. Decisions agreed on by three or less classifiers are ignored. Figure 4.14 shows an example image together with the character decisions by the six classifiers.

As shown in Figure 4.14, even when all the six classifiers agree, there is still a chance that the decision is an error (see the character decision 'C' in the figure). Typically, a human looking at that character in isolation may make the same mistake. Therefore, minor allowances for errors are still necessary in contextual postprocessing.

A regular expression matcher is used to postprocess the reliable decisions. A set of character constraints are generated using the decisions agreed on by 6, 5, or 4 classifiers. The constraints are in two groups. One group is to match the strings formed by all reliable characters in the word, the other group is to match single

reliable characters.

In the first group, all the reliable decisions in a word are matched in a single expression. There are three expressions in this group, derived from the decisions with reliability scores 6, 5, and 4, respectively. In a position where a reliable decision of that level is obtained, the decision is represented by an equivalence set containing the decided class, such as '[w|W]', '[x|X]', '[I|l]', and '[C|D|O|c|o]'. The purpose of using the equivalence sets is to allow for common shape confusions. A free character is placed in a position where a reliable decision of that level is missing. For instance, the string 'T-NNYSCN' in Figure 4.14 is represented by the expression 'T[A-Z|a-z]NNY[S|s][C|D|O|c|o]N', where '[A-Z|a-z]' means a free character. Each expression is associated with a score according to the level of reliability it represents.

In the second group, each reliable decision is matched in an individual expression. A number of free characters are placed to the left and the right of the character, according to the position of that character in the word image. The numbers are not exact, optional characters are used to allow for some fuzziness in the position of the decided character. For instance, in the example given in Figure 4.14, the decision 'S' is represented by the expression '....(.?|..)[S|s].(.?|..)', where each '.' is equivalent to '([A-Z]|[a-z])', and '(.?)' means that character may or may not be there, and '(.?|..)'

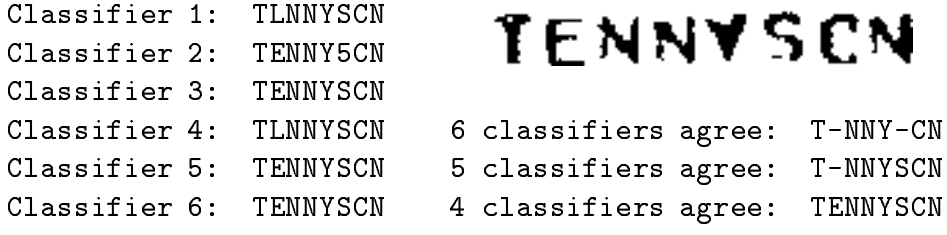


Figure 4.14: An example image (TENNYSON) and top choices by 6 classifiers.

means that there may be zero, one or two characters at that position. Again, a score is associated with each expression and the score varies with the reliability level of that decision.

The regular expressions thus derived are compared to the words in the lexicon [3]. A word in the lexicon that successfully matches an expression receives the score associated with that expression. A word may match more than one expressions and the scores can be accumulated. The words are then ranked by the accumulated scores.

4.7 Segmentation-Based Word Recognition

An alternative to the character recognition based techniques is to defer decisions about the character identity and to perform *segmentation-based word recognition.* This technique is suitable for word images where the characters can be reliably segmented, and better recognized together with other characters in the word.

In this approach, features are extracted from character images and matched at the word level (Figure 4.15). A word image is first segmented into individual character images, which are then normalized to a fixed size. Features are extracted from the normalized character images and represented by feature vectors. These character feature vectors are then concatenated to form a word feature vector, which is then matched with the prototypical feature vectors for the words in the lexicon. Hence the character features are compared in the context of a word.

This method is most useful for word images that are easy to segment but whose characters are difficult to recognize in isolation. An example image is given in Figure 4.16. This image is so broken that most of the shape features are lost, and the isolated characters are so degraded that even a human looking at them individually would have difficulty in guessing their identities. However, when the characters are

placed together as in the original word, they reinforce one another and the word identity can be determined. A few other examples are shown in Figure 4.17.

Binary pixel values are example features useful in this approach. Suppose that each character is normalized to 24×24 in size. The pixel values of each character can then be concatenated to form a word feature vector. Thus, a word segmented into 6 characters has a feature vector of 24×24×6 = 3,456 components. These vectors are then matched to the prototype vectors for words in the lexicon. The prototype vectors for each word are synthesized from the character features of some sample fonts.

The Hamming distance can be used to match these vectors. The distance between two vectors is the number of different components in corresponding positions. The distance is normalized by the number of characters compared.

Words in the lexicon that are of lengths equal to the number of extracted charac-

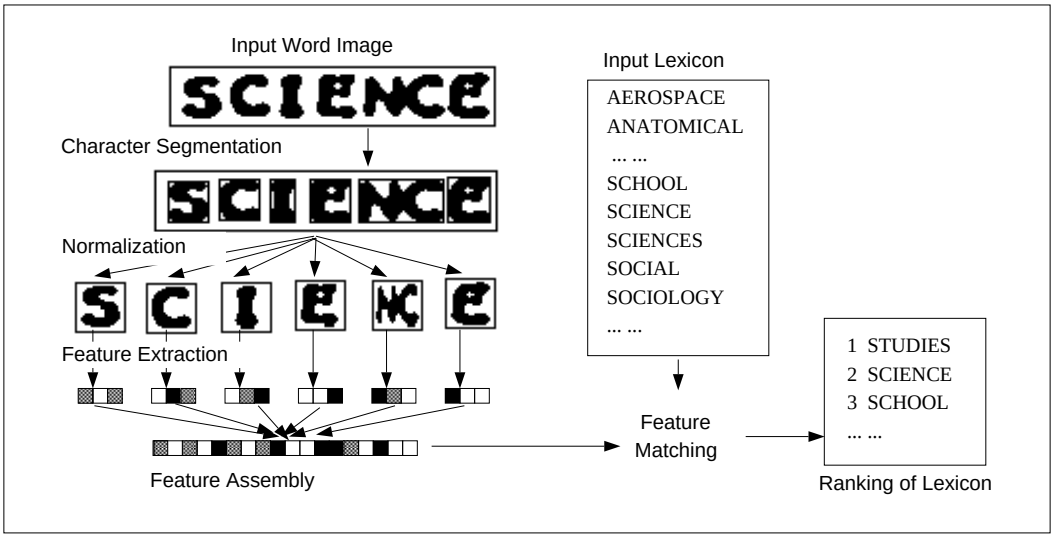


Figure 4.15: The segmentation-based word recognition approach.

ters are compared directly. The words of lengths that differ by one from the segmentation are compared with one character removed at a time from the longer vector. The minimum distance among all the comparisons to the same word with various characters removed is taken as the distance between the input to that word. The distances from the input to the words with unequal lengths are penalized by a weight, which is determined experimentally. The words are then ranked by the adjusted distances.

The uniqueness of this approach is better observed when it is put among the other methods in an implementation. Empirical studies reveal that there are certain types of images that are better recognized by this method than by other techniques. As in the character-based approach, the success of this approach is largely determined by the reliability of the segmentation algorithm.

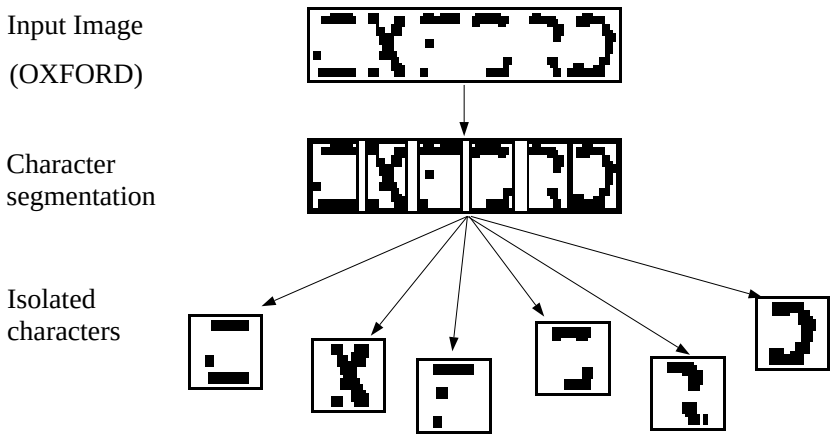


Figure 4.16: A word image and the extracted characters.

4.8 Word Shape Analysis

The word shape analysis approach attempts to describe and compare the shape of the word as a whole object. Features that describe details of a word shape are extracted and their locations are recorded in a feature descriptor. The feature descriptor is matched to the descriptors of the prototypes of the words in a given lexicon. A ranking of the lexicon is produced by evaluating these matches. The method is illustrated in Figure 4.18.

The advantage of this approach is that some errors in character segmentation and premature decisions on character identities are avoided. It is especially suitable for images that are difficult to segment into characters, or where the characters are distorted when they are extracted and normalized. Figure 4.19 shows some example images that are better recognized by word shape analysis.

The shape of a word is composed by the shapes of the characters in the word. Many variations are introduced in the composition process, including the vertical positioning of the characters, and the widths of the white gaps between neighboring characters. In most machine-printed text, characters are aligned along the top line and the base line of a word. Therefore these reference lines are useful to describe the locations of shape features. Certain normalization is necessary to reduce the variations in the horizontal spacing between neighboring characters. Section 4.8.1

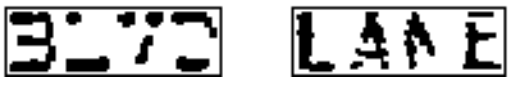


Figure 4.17: Word images better recognized by segmentation-based methods using word level context.

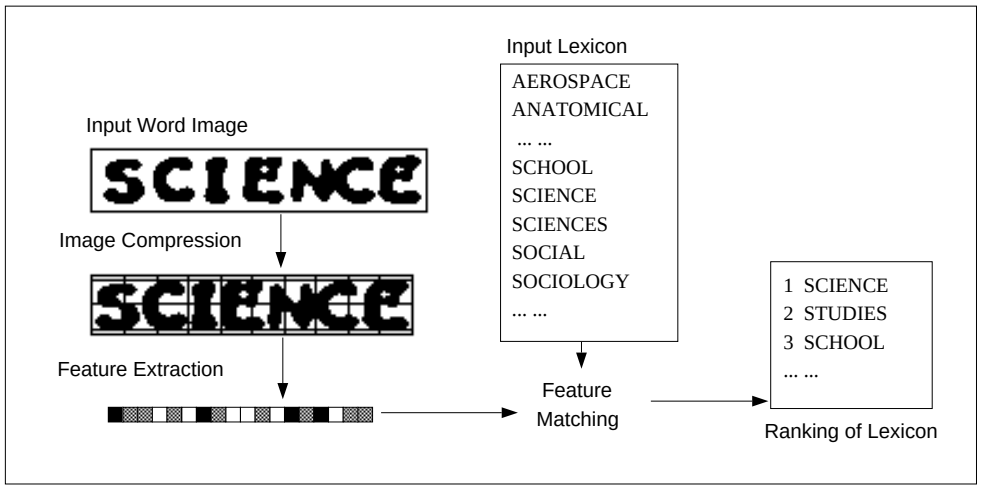


Figure 4.18: The word shape analysis approach.

describes these preprocessing procedures.

In Sections 4.8.2 and 4.8.3 a set of features and a classifier useful for word shape analysis are described. Section 4.8.4 discusses other features and descriptors that are potentially useful with a word shape analysis methodology.

4.8.1 Word Image Area Partitioning

Word shape analysis is different from other methods in that a global reference frame defined on the word image is used to represent the locations of shape features.

The global reference frame consists of four reference lines including the upper



Figure 4.19: Word images better recognized by word shape analysis.

boundary, the top line, the base line, and the lower boundary. The upper and lower boundaries are the upper and lower edges of the most succinct rectangle that contains all the black pixels in the image. The top line is a line formed by the top pixels of the lower case characters that do not have an ascender nor a dot ({a, c, e, g, m, n, o, p, q, r, s, u, v, w, x, y, z}). The base line is a line formed by the bottom pixels of all the upper case characters as well as the lower case characters that do not have a descender ({a, b, c, d, e, h, i, k, l, m, n, o, r, s, t, u, v, w, x}).

These four reference lines are detectable in most words consisting of mixed case characters. The four lines divide those images into three vertical regions: the ascender region, the middle region, and the descender region. Because many characters locate entirely in the middle region, that region is further divided into upper and lower parts to facilitate more accurate position description. The image is divided into ten equal-sized regions along the horizontal axis. As a result, the image area is partitioned into 4 vertical regions, and 10 horizontal regions, i.e., 40 cells.

The top line cannot be detected in words consisting of purely upper case characters, as well as in words that do not contain short characters, such as the word "Hill". In such cases the top line will be made identical to the upper boundary. The base line may also be identified with the lower boundary for images without descenders. Therefore some images may have no ascender or descender region. The middle region divisions and the ten horizontal divisions are unaffected. For convenience in implementation, the image is always considered to be in 40 cells, though 10 or 20 of them may be empty.

To avoid variations in gap widths between neighboring characters, the word image is preprocessed by reducing all detected white gaps to one pixel wide (Figure 4.20). This normalization procedure is useful when the technique is applied in a multi-font environment. Though, it has to be applied with caution to faint images, and

should be omitted if most of the images encountered in an application are vulnerable to the compression.

The top line and the base line are then estimated by the following method. A word image is first smeared horizontally. The upper and lower contours of the smeared image are then computed. A histogram is computed for the heights from the upper boundary to the upper contour. The mode of these heights is taken as the location of the top line. The base line is determined by finding the mode in the histogram of heights from the lower contour to the lower boundary. Figure 4.21 shows an image before and after smearing, and the located reference lines. Figure 4.22 shows the area partitions thus computed.

4.8.2 Word Shape Feature Extraction

A set of features that is referred to as the *stroke direction distribution* is used to describe the shape of a word. It captures the spatial distribution of black pixels belonging to strokes of various directions.

The features are extracted using the *local direction contribution* method suggested in [93]. At each black pixel in the image, the length of the current run in each of the four directions east-west, northeast-southwest, north-south, and northwest-southeast is computed. The pixel is labeled with the direction in which the *run*

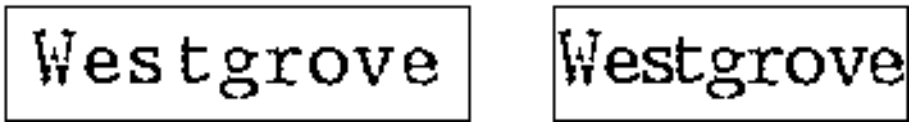


Figure 4.20: An image before and after white gap packing.

length is a maximum. That is, each black pixel is labeled as part of a stroke of one of the four directions. Figure 4.23(a)-(e) shows an example of such a pixel labeling.

For each of the 40 cells in the image area, the labeled black pixels of each type in that cell are counted. The counts are then normalized by the total number of black pixels in the image. The stroke direction distribution is represented by a 160-dimensional feature vector, which stores the normalized counts of black pixels of each of the four types in the 40 cells.

4.8.3 Prototype Matching and Classification

Models or feature prototypes for the words in the lexicon are needed to match the feature vector computed from the input image. It is impractical to collect a large set of image samples for each word in the lexicon, therefore we adopt an alternative strategy, that is, to synthesize the words from a set of *character* samples of various fonts. The features of the synthesized words are then extracted and stored as prototypes. It is also possible to extract the features from the characters first, and then synthesize the word features with the knowledge of the locations of the character features in a word.

A city-block distance [33] is used to compare the feature vector of an input



Figure 4.21: An image before and after smearing, and the located top line and base line.

word with that of a word in a lexicon. The distance is the sum of the absolute differences of corresponding feature components. That is, if $\mathbf{x} =< x_1, x_2, ..., x_n >$ and $\mathbf{y} =< y_1, y_2, ..., y_n >$ are two feature vectors, then the distance

$$d = \sum_{i=1}^{n} \mid x_i \ - \ y_i \mid$$

where $n = 160$ for the stroke direction distribution vector.

For an input image, the computed feature vector is matched to the prototype vectors for each word in the lexicon synthesized with each font sample. If there are M words in the lexicon, and N font samples are used to synthesize the words, then NM distance computations are needed for each input. For each word in the lexicon, the minimum distance among the N distances for the N fonts is determined. The M words in the lexicon are then arranged in order of increasing distances. This gives a ranking of the lexicon.

4.8.4 Alternative Feature Sets

Baird Features

Similar word shape analysis techniques can be developed using other feature sets. One example of alternative features is a set of templates proposed in [8], where 32 templates are described that define 32 local shape features. These features are detected by convolution and thresholding. Each nonzero response after thresholding



Figure 4.22: An image and its 40 area partitions.

represents that a feature is detected at that position. The outputs are described by a 1280-dimensional feature vector, which stores counts of the 32 features detected in the 40 cells. The city-block distance and a nearest-neighbor classifier can be applied to these feature vectors.

Structural Features

Word shape can also be described by higher level structural features and symbolic descriptors. Examples of such features include edges, end points, vertical bars, horizontal bars, holes, dots, and arcs connecting adjacent vertical or horizontal bars.

Edges. Edges can be computed by first identifying the edge pixels. This is done by checking, for every black pixel, if there is any horizontal or vertical *run* starting or ending at that pixel. In this process, each detected edge pixel is labeled as belonging to a horizontal edge or a vertical edge. Figure 4.24(a) shows the identified edge pixels for the image in Figure 4.23(a). The edges are given by connecting the detected edge pixels of the same orientation. Since the edges appear in pairs, only one is needed to

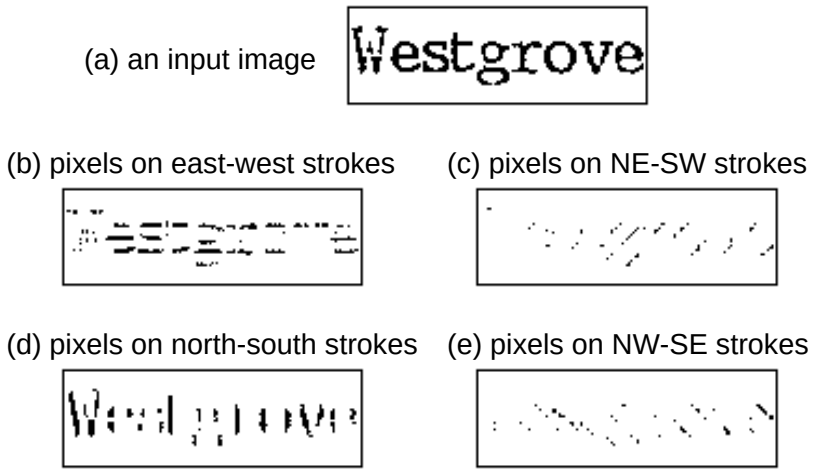


Figure 4.23: An example of stroke direction distribution.

represent each stroke. Therefore the longer side of each pair is selected. Short edges that are one or two pixel long are treated as noise and ignored. The selected edges are shown in Figure 4.24(b). These selected edges are registered with their orientation (horizontal or vertical) and location.

End Points. End points are where strokes start and terminate. They can be computed by convolving a set of end point templates with a skeleton image. The skeleton is computed by consecutive applications of the thinning algorithms given in [8] and [80]. Figure 4.25 gives an example of skeletonization by these methods as well as the endpoints detected by convolution.

Other Shape Features. Examples of other shape features include some higher level perceptual features, such as *dots* and *holes*, computed by connected component analysis on the input image and the negated image; *ascenders, descenders, short vertical strokes*, and *horizontal strokes*, computed by run length analysis; *diagonal strokes*, computed by chain code analysis on the skeleton image; *arcs*, computed by a multi-scale curvature analysis [30] on the chain codes obtained from the skeleton image; and *bridges between neighboring vertical and horizontal strokes*, computed by checking whether two neighboring strokes belong to the same connected component in the input image.

Feature Representation by A Location Code String



Figure 4.24: Example of edge detection and selection.

The feature vector description is subject to discretization error in representing the feature locations. For the edge, end point and letter shape features, the presence or absence of each feature is important so that the descriptors should allow matching of corresponding features located in *neighboring* cells. Therefore a more robust descriptor is needed.

We can first categorize these features according to their vertical positions. For instance, a vertical edge may be labeled as an 'ascender region vertical edge,' a 'descender region vertical edge,' or a 'middle region vertical edge.' Then their locations are represented by numeral strings specifying the horizontal regions they are in. For instance, a string 'O: 256' describes that there are three holes in the image, and that they are located at the 2nd, 5th and 6th horizontal regions respectively. One numeral string is used for each of the feature types.

These numeral strings are compared by the *minimum edit distance*, computed by a *dynamic programming* algorithm given in [140]. The edit costs are derived from the reliabilities of detection measured using a training set. The substitution costs are the numerical differences between the digits in the strings. Hence a string '256' can be matched with another string '146' with a distance of 2 (i.e., $(2-1)+(5-4)+(6-6)$). Though there are some slight differences in their locations, these two sets of features can still be matched.

Feature Representation by A Symbol String

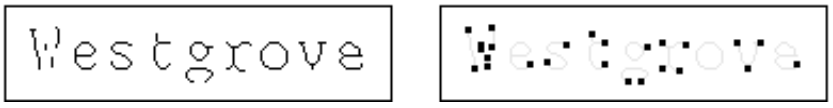


Figure 4.25: A skeleton image and the end points detected by convolution.

The co-existence and relative locations of structural features of different types are also important for recognition. This information is captured by descriptors in the form of a symbol string. For instance, a string 'AOOD' is used to represent that an ascender is followed by two holes and then a descender. Some features are located above others. Such features are better described in separate strings. These symbol strings are also compared by minimum edit distances. The edit costs are derived from the reliabilities of detection as well as substitution frequencies observed with a training set.

Some of the structural features in a word image are shown in Figure 4.26, where (a) shows an input word image, (b) shows the vertical bars extracted by run length analysis, (c) shows the edge points detected, (d) shows the edges selected from each pair, (e) shows the skeleton image computed by a thinning algorithm, (f) shows end points on the skeleton image that are detected by convolution, (g) shows the same set of end points placed in a reference frame, and (h) shows the symbolic descriptors for these end points.

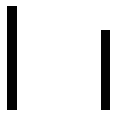
Evaluation of Feature Effectiveness in Word Recognition

Though the initial selection of features are often subjective, the effectiveness of a particular set of features can be evaluated by their contribution to classification correctness.

In our experiments with degraded images, it was observed that the best performance was given by the features that are defined locally yet contain certain simple shape information, such as the stroke directions and the Baird templates. In other words, neither the signal level features, such as the pixel values, nor the highest level structural features, such as the holes and horizontal runs, are satisfactory for degraded images. Significant shape information is missing in the lowest level features



(a) an input image



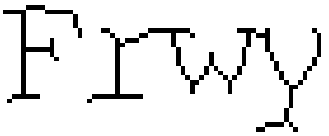
(b) extracted vertical bars



(c) detected edge points



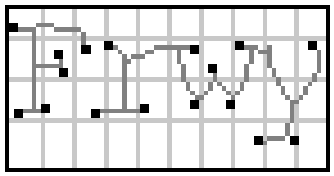
(d) selected edges



(e) skeleton image



(f) detected end points



(g) end points in a reference frame

$alluuululululuddu$ [endpoint type symbol string]
a 0 [location : ascender region endpoints]
d 78 [location : descender region endpoints]
u 11235679 [location : upper middle region endpoints]
l 012456 [location : lower middle region endpoints]

(h) symbolic descriptors for detected end points

Figure 4.26: Examples of structural features and their descriptors.

and they are often subject to too much variability that complicates their comparison. On the other hand, detection of the structural features is often unreliable so that the descriptors computed from an input image are often far different from those computed from idealized prototypes. For structural features encoded in compact descriptors, a similarity measure that is tolerant for unreliability in detection is often difficult to obtain.

Nevertheless, in a problem where discrimination among a small number of classes is desired, one may find the structural features useful. For instance, when recognition is confined to two classes, it is possible that a single structural feature is sufficient to tell the difference.

4.9 The Control Strategy

The objective of the control strategy is to simplify the recognition task and avoid unnecessary computations without causing a negative effect on recognition correctness. It achieves the objective by two selection procedures: selection of a minimum number of words in the lexicon that match some reliable descriptions of an input image, and selection of a minimum number of classifiers that are needed to recognize the image.

The effects of these selections are on both efficiency and correctness. By activating only the needed classifiers and comparing the input with only the likely words, the computation effort is minimized. By selecting only the likely words in the lexicon, a smaller number of words need to be discriminated and hence the risk of error is reduced. By applying only the suitable classifiers, the risk of errors made by other classifiers is avoided. The selection of the classifier is much more difficult than the selection of words. The details of each selection procedure are described in Sections 4.9.1 and 4.9.2.

4.9.1 Lexical Activation by Global Features

Global features are simpler aspects of a word shape that can be measured only in the scope of the whole word. Some of them can be easily and reliably measured. They are used to reduce a given lexicon to simplify subsequent classification tasks. Global features that are useful for this purpose include estimates of the word length and word case. If the estimates are accurate, only the words of the estimated length need to be activated, and the words can be converted to only the estimated case.

It is difficult to estimate word length precisely, therefore an interval estimate is used instead. This is done by first performing character segmentation, and then relaxing the character count to be an interval by examining the variations in the character sizes.

The word case is estimated by examining variations in the sizes and the vertical alignment of the connected components. It can also be estimated by an analysis of the heights from the located top line and base line to the upper and lower boundaries. To improve reliability, the estimates given by the two methods are compared. The word case is left undetermined if there is no agreement. The word case can also be pre-determined by higher level constraints, such as the syntactic category of the word.

4.9.2 Dynamic Selection of Classifiers

The objective of a dynamic selection of classifiers is to improve recognition correctness and efficiency. The selection is based on the quality characteristics of the input image.

Improvement on efficiency is achieved by activating a minimum number of classifiers needed to recognize an image. For instance, if an image is well-printed such that character segmentation is highly reliable, and several character recognizers agree on the identity of each character, it is likely that character recognition methods alone

will provide a correct decision. In such a case, there is no need to apply other approaches. Though, the estimate of image quality must be conservative to avoid possible mistakes.

Improvement on recognition correctness is achieved by applying only the classifiers that are suitable for images of the same type as the input. This is based on the assumption that the word images can be partitioned into several types, and suitable classifiers for each type are available. A measurement is needed that pre-classifies an image into a particular type. For example, a count on the connected components may be used to pre-classify a word image as in a dot-matrix font or a solid-faced font. Knowledge about the correctness of each classifier for images of each type is also needed to perform dynamic selection. Therefore the dynamic selection procedure is likely to be a heuristic design and specific to each application.

Ideally, a control strategy selects the best classifier for each input image, and only that classifier is applied. If such a selection is always successful, and if the selected classifier always gives a correct decision at the top choice, there is no need to apply other classifiers to the same image. However, this is possible only in a limited number of cases where the image quality is exceptionally good. When it is impossible to predict which single classifier will recognize an image correctly, a number of classifiers should be applied in parallel. In such cases, decision combination is needed to derive a consensus classification.

4.10 Decision Combination

The decision combination mechanism uses the results of the activated classifiers to derive a consensus decision. The decisions of each classifier are represented by a ranking of the activated words in the lexicon. The rankings are combined using the functions proposed in Chapter 3.

To recognize words of multiple font types and variable quality, the class set reordering methods are more useful since they are less affected by outlying worst cases.

The highest rank method is applied to rerank the words and thereby reduce a large lexicon to a small neighborhood. A score is assigned to each word that is the highest of the ranks it receives from all the classifiers. Therefore, a word receives a good score as long as at least one classifier ranks it high. The combined ranking is given by sorting the words by their scores. Words ranked above a threshold in the combined ranking are taken as a neighborhood.

The Borda count method is used to rerank the words included in the neighborhood computed by the highest rank method. The Borda count for each word is the sum of the number of words ranked below it by each classifier. The combined ranking is given by arranging the words in descending Borda count. Intuitively, if a word is ranked near the top by more classifiers, its Borda count tends to be larger and will be closer to the top in the combined ranking. It is a measure of agreement among the classifiers.

The logistic regression method can be used to rerank the words close to the top of each ranking. The logit of each word is computed as a weighted sum of the ranks it receives from the classifiers. The weight for each classifier is estimated by logistic regression analysis, using a set of images allocated for training.

The final ranking of the lexicon is therefore a concatenation of three rankings: first the ranking given by the logistic regression function, then the remaining words in the neighborhood that are ranked by the Borda count, and finally, the words outside the neighborhood that are ranked by the highest rank method. This final ranking is the output of the model.

4.11 Remarks

Various classifiers can be constructed taking each of the three approaches to word recognition. As there are infinitely many possible definitions of features, corresponding similarity scores, and classification procedures, choices of methods are virtually unlimited. A method is potentially useful as long as it produces a ranking of the lexicon.

Rank-based decision combination hence provides a unifying tool, that can bring together the contributions by various classifiers, no matter whether they are statistical classifiers, syntactic classifiers, or structural classifiers, using feature descriptors in the form of vectors, symbol strings, graphs, or others.

The effectiveness of a classifier is evaluated by a new criterion in the context of a multiple classifier system. The focus is on the classifier's significance in the context of other classifiers, instead of its performance in isolation.

Chapter 5

An Experimental Word Recognition Algorithm

5.1 Algorithm Descriptions

An algorithm that implements the computational model for visual word recognition described in Chapter 4 has been developed. The algorithm consists of several modules: a preprocessor, activation control, five parallel classifiers, and a decision combiner.

A preprocessor is used to enhance the quality of an input word image. Lexical activation is by estimates of the word length and word case. This implementation does not include dynamic selection of a single classifier, so that the classifier activation control is trivial. Five parallel classifiers are applied to all images that produce five independent rankings of the lexicon. The rankings are then sent to a decision combiner, which produces a consensus ranking that is the output of the algorithm.

In this chapter we focus on some implementation considerations and the experimental results. The word image preprocessor is described in Section 5.2.1. The five parallel classifiers are based on the methods discussed in Chapter 4, and are summarized in Section 5.2.2. Section 5.2.3 discusses how a set of font samples are selected for word prototype synthesis. Section 5.2.4 discusses some clustering methods that are needed to reduce the computational complexity in word shape analysis involving a large lexicon. Section 5.2.5 is devoted to speed considerations in a classifier using a segmentation-based method. Section 5.3 describes a set of experiments using a

collection of images obtained from live mail, and our observations. In Section 5.4, an example design of a dynamic selector and its performance are shown. The dynamic selection is at the decision combination stage.

5.2 Implementation Considerations

5.2.1 Word Image Preprocessing

The objective of the preprocessor is to estimate and enhance the image quality. The image quality is estimated by a measure referred to as the *density* of the image. The density is defined to be the number of black pixels divided by the image area which is the size of the minimum bounding rectangle that contains all the black pixels.

The computed density is categorized into one of six levels. An image at density level 0 is too faint to recognize is therefore rejected. An image at density level 5 is thick enough so that no enhancement is needed. For other images, two procedures, *smear* and *fill*, are applied to enhance the image quality. Procedure *smear (x,y)* shifts and overlays an image x times horizontally and y times vertically onto itself. Procedure *fill (x,y)* fills white gaps between black pixels x spaces apart horizontally and y spaces apart vertically. Horizontal smearing or filling is applied only to low density level images, since if overly applied they may cause a loss of important white spaces between neighboring characters, which in turn causes difficulty in character segmentation. The definitions of the density levels and corresponding enhancement procedures are summarized in Table 5.1. Figure 5.1 shows some example images before and after enhancement.

The density gives an indication of the image quality. However, this indication is not very reliable, since the density value is affected by the shape of the characters in the word as well as by the spaces between neighboring characters. It is effective in

Table 5.1: Definition of density levels and corresponding enhancement procedures.

Level	d: % of Black Pixels in Image Area	Enhancement Procedures
0	$0 \leq d < 3.2$	reject
1	$3.2 \leq d < 15.0$	*fill* (1,1) then *smear* (2,2) then *fill* (1,1)
2	$15.0 \leq d < 20.0$	*smear* (1,1) then *fill* (1,1)
3	$20.0 \leq d < 25.0$	*smear* (0,1) then *fill* (0,1)
4	$25.0 \leq d < 30.0$	*fill* (0,1)
5	$30.0 \leq d \leq 100.0$	no action



(a) An image of density level 1.



(b) An image of density level 2.



(c) An image of density level 3.



(d) An image of density level 4.

Figure 5.1: Example images of different density levels and results after enhancement.

detecting very bad images that should be rejected. The enhancement procedures described here are most effective in connecting fragments in a dot matrix printed image. For other images, the performance is satisfactory, except that undesired connections between strokes are occasionally produced.

Reference lines including the top line and the base line are then estimated for the enhanced image, by computing the contours in the smeared image as described in Chapter 4. The locations of the reference lines are stored in the header of the image and passed to the feature extraction algorithms.

5.2.2 The Five Parallel Classifiers

Five parallel classifiers are used to rank the activated words in the lexicon. The classifiers are based on the three approaches to word recognition: the character-based method, the segmentation-based method, and word shape analysis. The methods used in the five classifiers are summarized in Table 5.2. Each classifier produces a ranking of the lexicon that has been filtered by word length and word case in the activation controller. The five rankings are then sent to the decision combiner.

Table 5.2: Summary of methods used in the parallel classifiers

Classifier	Method
Classifier 1	A fuzzy character template matcher and a heuristic contextual postprocessing algorithm.
Classifier 2	Six character recognizers with decisions postprocessed by regular expression matching.
Classifier 3	A segmentation-based word recognizer using pixel values as features.
Classifier 4	A word-shape analyzer using stroke direction features.
Classifier 5	A word-shape analyzer using the Baird template features.

Classifiers 1,2, and 3 take their input from a character segmentation algorithm

applied to the preprocessed image. Character segmentation is performed by an analysis of the vertical projection profile. Columns containing one or more black pixels are labeled in the image. Neighboring columns are then merged in a right to left scan while the width of each resultant component is kept within a threshold. The components are then re-examined. If the width-to-height ratio exceeds that of a regular character, the component is split into two at a valley of the vertical projection profile. The image is segmented into these components. Each component thus obtained is considered as a character.

In classifiers 3,4, and 5, a set of font samples are used to synthesize the shapes of the words in the lexicon. The selection of the font samples is discussed in Section 5.2.3.

5.2.3 Selection of Font Samples for Word Prototype Synthesis

In classifiers 3,4, and 5, a set of sample font types are needed to synthesize the prototype feature vectors for each word in the lexicon. The font types used affect the probability of correct recognition to a large extent. Because of possible confusions between similar shapes of different characters, using more fonts does not necessarily improve recognition correctness. Therefore it is important to determine how many, and which font types should be used to generate the prototypes.

The problem of selecting sample font types to train a general purpose page reader has been addressed previously [10]. To guarantee a broad coverage of most commonly seen fonts, statistics of font type usage are needed. These statistics are difficult to obtain and verify. Even if the study is restricted to a specific collection of text samples, manual identification of the fonts used in a specimen is not necessarily easy.

We therefore consider an experimental solution to this problem. This solution is based on the assumption that a training set is available, and that the training set represents well the stylistic variations in the images expected by the recognition algorithm.

The number of font samples needed is influenced by the sensitivity of the features and the classifiers to the shape variations in these samples. Therefore our font selection is intended to meet the needs of a particular recognizer instead of all possible ones.

Assume that a large collection of font samples is available. The selection process is illustrated with classifier 4 (word shape analysis using stroke direction features) as an example.

Assume also that the true words are known for each of a sufficiently large set (say, 1,000) of word images available for training. The prototype vectors of the true word of each image are synthesized using all available font samples, and matched to the vector computed from that image. A distance is calculated between the input vector and each prototype vector.

The fonts are then ranked by the distances from the input vector to the corresponding prototypes generated with those fonts. The font that best matches the input is identified. The same procedure is applied to all images in the training set. As a result, for each training image, the font which best matches it is determined.

For each font in the collection, the number of times it is identified as the best font for an image is counted. The fonts can then be sorted by these frequencies in descending order. The cumulative frequencies are computed for this ordered list. The fonts are then selected from the beginning of the list, until a threshold on the cumulative frequency (say, 90% of all the training images) is reached.

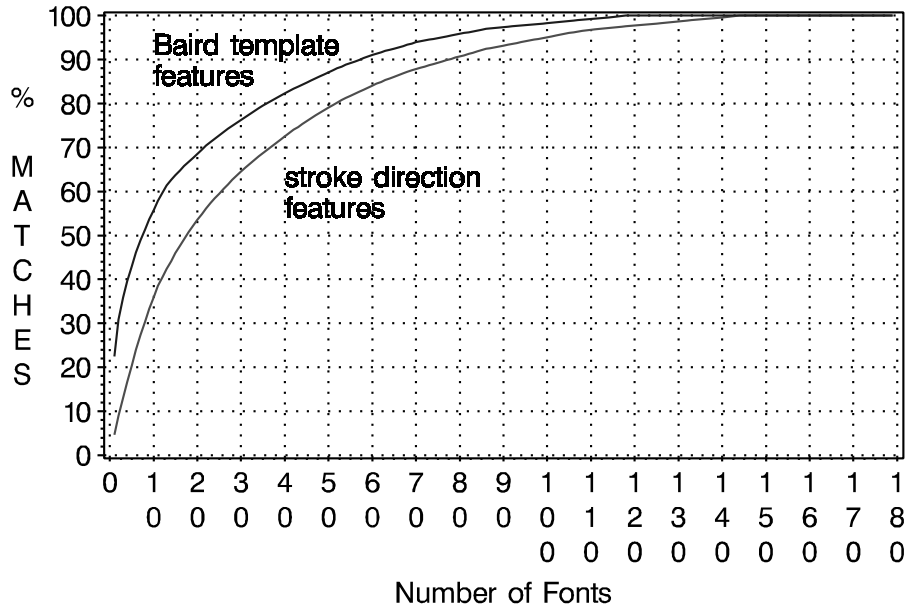


Figure 5.2: Results of font matching using 1,055 training images.

Results of font selection for classifier 4 (using stroke direction features) and classifier 5 (using Baird template features) are shown in Figure 5.2. The selection was performed using 1,055 word images as a training set. Prototypes of the true words were synthesized using a collection of 179 font samples. It was shown that, 90% of all the 1,055 best matches were obtained using 77 of all the 179 fonts, when the stroke direction features were used; and the same number of best matches were obtained using 57 fonts when the Baird template features were used. In both cases, a large number of fonts in the collection (35 for the stroke directions and 61 for the Baird features) were never a best match to any word image in the training set. These fonts can therefore be removed from the sample set. If the 90% threshold is used, we need to use only 77 font samples in classifier 4 and 57 in classifier 5. An observation from this study is that the Baird templates are more robust than the stroke directions for variations in font styles.

5.2.4 Prototype Clustering in Word Shape Analysis

The word shape analysis method is applied in classifiers 4 and 5. In classifier 4, the stroke direction features described in Chapter 4 are used to characterize the word shape. Using the 40 area partitions and the four stroke directions, a vector of 160 dimensions is obtained for each word. The vector is matched to the prototype vectors of words in the lexicon using the city block distance.

The computation requirement is very intensive for this algorithm. For a lexicon of 10,000 words and using 50 sample fonts, the algorithm performs 160 (feature components) $\times$ 10,000 (words) $\times$ 50 (fonts) = 80,000,000 integer difference calculations for each image. A more feasible design is to use the average vector of all sample fonts instead. That takes 160 $\times$ 10,000 = 1,600,000 calculations and is 50 times faster than the original design. Experimental results show that degradation in performance resulting from prototype averaging is acceptable, especially if this classifier is used together with some others.

The features defined by the Baird templates [8] are used in another word shape analysis procedure in classifier 5. There are 32 templates, each is convolved with the input image to detect a local shape feature. A vector of 32 (features) $\times$ 40 (area partitions) = 1,280 dimensions is computed for each image. The vector is matched to prototype vectors of words in the lexicon using the city block distance.

For a lexicon of 10,000 words and using 50 sample font types, the algorithm takes 1,280 $\times$ 10,000 $\times$ 50 = 640,000,000 integer difference calculations and is very slow. Even with the averaged vector design, it still takes 1,280 $\times$ 10,000 = 12,800,000 difference calculations and needs to be further improved in speed. One useful technique is to collect the word vectors into clusters. The input vector is compared to only one representative vector from each cluster. The clusters are then ranked by the

distances to the input. Only the clusters ranked near the top are expanded and the members of which are matched to the input. The words represented by these vectors are ranked by the distances to the input. This method does not give a full ranking of the lexicon. However, in practice, with a lexicon of 10,000 words, it is rarely needed to look at the tail portion of the ranking, so the method does not hurt the performance by large.

5.2.5 Speed Considerations for the Segmentation-Based Method

The segmentation-based method is applied in classifier 3. Each character extracted from a word is normalized to 24×24 in size. The binary pixel values in each normalized character are represented by a vector of 576 dimensions. These vectors are concatenated to form a feature vector for the word. For instance, the concatenated vector for a word with 6 characters has 576×6 = 3,456 dimensions. The vector is matched to the prototype vectors for words in the lexicon, which are synthesized with a set of sample fonts. The Hamming distance is used in the comparison.

The matching algorithm can be improved in speed by a careful design. For instance, matching a vector of 576 (feature components) × 6 (word length) dimensions to 10,000 words of 50 sample font types involves 1,728,000,000 component difference calculations. This can be avoided by using the fact that the words are various combinations of the 52 characters (upper and lower cases) in the alphabet. The features of each input character need to be matched only once to the features of each character in one font type. The differences can be stored in arrays indexed by the character and the font type. This involves 576 (feature components) × 52 (character classes) × 6 (word length) × 50 (font types) = 8,985,600 component difference calculations. The distance to each word in the lexicon can then be computed by summing up the stored differences from the input to each character in the word of each font type.

This step involves 6 (word length) $\times$ 10,000 (words) $\times$ 50 (font types) = 3,000,000 summations. Hence this design gives a much faster algorithm.

5.3 Experimental Results

5.3.1 Background of the Application

The application is in the context of a project aimed at automation of mail sorting. Automation of mail sorting in the United States is facilitated by the introduction of the ZIP Code system. Early systems attempt to locate and recognize the ZIP Code printed on an envelope. As one has virtually no control over the font type and print quality of the incoming images, such ZIP Code recognition is not always reliable. It is becoming obvious that in order for the system to be tolerant to errors in postal code recognition and possible patron errors, it is necessary to perform a complete redundancy check on all the information available in an address image. This involves reading the addressee information, such as the personal name or the business name, the street address, such as the street number, street name, street suffix and directional prefix, as well as other words such as the city name and the state name.

The objective of our study in this project is to design a robust word recognition algorithm. The algorithm takes in a word image extracted from an address block image, and a lexicon containing words in the syntactic category that the word has been labeled, and produces a ranking of the lexicon for further contextual analysis. An example lexicon is the collection of all street names in a predefined area. The lexicons are compiled from the postal databases. Problems unrelated to visual recognition, such as aliases and alternate spellings, are ignored in the scope of this study.

5.3.2 The Image Database

The image database was collected from live mail on a postal optical character reader. The images are all machine-printed address blocks, scanned at a resolution of 212 pixels per inch and were binarized. Words were extracted from the address block images in a semi-automatic process. The true word was entered to each image header manually so that the recognition correct rate can be measured. The font and quality of the images vary. The images shown in Figure 1.4 are examples from this collection. The image database was biased to include more bad images than could be found in a random sample, therefore our results could be underestimates of the achievable performance.

5.3.3 System Training

The feature extractors were developed and modified by observing their performance on a small initial training set of 200 images. The character recognizers were trained on a set of 19,151 sample characters extracted from 400 address blocks. The prototypes of stroke direction vectors were produced by averaging those from 77 font samples. For Baird feature vectors, 57 fonts were used. A set of 76 fonts was used to construct prototypes for the segmentation-based method using pixel features.

The word classifiers were applied to a set of 1,055 word images using a lexicon of 33,850 postal words. The lexicon included all the true words in the images. Each word in the lexicon is represented in both upper and mixed cases. In the application domain, purely lower case is not used. The results of this run were used to estimate the regression parameters for decision combination. Rankings from the five classifiers as well as their combination by the highest rank method were used in the regression analysis. Hence six rankings are to be combined by a logistic regression function,

and the estimated weights are 0.1775, 0.2098, 0.3292, 0.0539, and 0.1880 for the five classifiers respectively and 0.1864 for the ranking given by the highest rank method.

5.3.4 Test Results

Another set of 1,671 images was used to test the recognition algorithm. The same lexicon of 33,850 words was used. Table 5.3 summarizes the performance of the five word classifiers and the results at several stages in decision combination. In order to measure the performance of the classifiers using a lexicon of a fixed size, lexical activation by word length and word case is applied after recognition.

Three subsets of sizes 10,000, 5,000, and 1,000 were then selected from the input lexicon. The objective was to determine the potential effect of a global contextual knowledge source that could provide reduced lexicons to word recognition and thereby improve its performance. Each of the subsets of the original lexicon contains all the true words in the images. The rest of the words in the subsets were randomly selected.

Table 5.3: Summary of performance on 1,671 test images using a 33,850 word lexicon

	% Correct in Top N Choices						
Classifier / Combination	1	2	3	10	50	100	500
1) char recog with heuristic postprocessor	79.2	86.1	88.2	90.5	92.7	93.5	94.8
2) char recog with regular expression matcher	76.9	83.2	85.4	88.3	91.8	93.2	95.0
3) segmentation-based method	74.8	84.1	86.3	90.5	93.4	94.6	95.5
4) word-shape with stroke direction features	42.4	53.7	59.5	72.1	81.9	84.9	90.8
5) word-shape with Baird features	58.9	70.0	74.5	82.9	88.5	90.2	93.2
6) combination of 1)-5) by the highest rank	46.6	73.8	87.0	94.9	97.3	97.6	98.6
7) combination of 1)-6) by the Borda count	83.1	88.1	90.7	94.6	96.3	97.0	98.6
8) combination of 1)-6) by logistic regression	88.4	91.2	92.7	95.1	97.4	98.1	98.6
9) results of 8) after case (upper,mixed) merging	88.7	91.4	92.9	95.2	97.6	98.5	98.9
10) results of 9) after word length filtering	88.9	91.6	92.9	95.3	97.8	98.4	98.7
11) results of 10) in a 10000 word subset	92.6	94.0	94.9	97.0	98.6	98.8	98.9
12) results of 10) in a 5000 word subset	94.1	95.0	96.0	97.8	98.7	98.9	98.9
13) results of 10) in a 1000 word subset	95.5	97.1	97.9	98.7	98.9	98.9	98.9

5.3.5 Observations

Using the highest rank method in decision combination, the lexicon was reduced to a neighborhood of 10 words with a 94.9% accuracy and to 500 words with a 98.6% accuracy, which was not achieved by any of the individual classifiers. This method did not give a high correct rate at the top choice because of ties among the top five decisions. Better consensus rankings were achieved by the Borda count and the logistic regression method. There was a 9.2% gap between the top choice correct rate of the combination by logistic regression (line 8) and that of the best individual classifier (line 1). This shows that the integrated algorithm is significantly better than any of the individual classifiers in performance. The filtering stages had little effect on the top choice correct rate. The results on the random subsets (lines 11,12,13) suggest that other higher level contextual constraints on the lexicon can be used to improve the final rankings significantly.

5.4 Dynamic Selection of Classifier Combinations

The idea of dynamic selection was tested in another experiment. The selection is performed at the decision combination stage, based on a condition computed from the class rankings with respect to an input image.

Four classifiers were used to recognize a set of word images from the same source as described in the previous section. Classifiers 3,4,5 used in the previous experiment were included.

The fourth classifier uses a polynomial discriminant method for character recognition, and a confidence based method for contextual postprocessing. The confidence values are computed by the character recognizer. Top five class choices and their confidences are considered for each extracted character. For each word in the lexicon,

a joint confidence is computed which is the sum of the confidences of the characters. Words of one character longer or shorter than the number of extracted characters are modified by adding a null character or removing a character from each position in turn. Words with larger length differences are ignored. The lexicon is ranked by the joint confidences.

The words were recognized using a lexicon of 1,365 words. The lexicon has 2,707 entries after each word was converted to both upper and mixed cases. Therefore we can consider that 2,707 classes are given. Case merging is applied only at the last step.

The four classifiers were applied to a set of 4,624 training images, and the class rankings were used to estimate a logistic regression model. Besides, for each image, the top choice by each classifier is compared to the top choice of other classifiers and an *agreement class* is assigned accordingly.

The agreement class is assigned by observing if there are two or more classifiers making the same top choice, and which classifiers they are. This is motivated by the observation that the classifiers tend to agree at the top choice for good quality images, and tend to disagree for bad images. Therefore the top choice agreement can be used as an indicator of image quality.

There are 12 ($C_4^4 + C_4^3 + C_4^2 + C_4^0$) possible states of agreement among the four top choices, ranging from "all four agree" to "all four disagree". Hence 12 agreement classes are defined. The training set is then partitioned into 12 subsets, each of a different agreement class. A logistic regression model is estimated for each subset. The definitions of the agreement classes, as well as the estimated model parameters are given in Table 5.4.

Note that the agreement class is defined with no reference to the correctness

Table 5.4: Definitions of agreement classes and estimated model parameters

Agreement	Which Classifiers Agree (x)				Size of	Estimated Model Parameters			
Class	poly	segb	sfv	bfv	Partition	poly	segb	sfv	bfv
1	x	x	x	x	1320	0.3584	1.5048	2.8836	1.0190
2	x	x	-	x	325	0.3955	7.0355	0.1686	0.3563
3	x	x	x	-	136	5.7066	5.7528	0.1362	-0.1444
4	x	-	x	x	454	0.2327	0.0529	5.4321	0.1187
5	-	x	x	x	712	-0.1032	1.8564	0.8661	0.8590
6	x	-	x	-	204	0.4422	0.0001	2.5731	0.0288
7	x	-	-	x	166	0.5860	0.0421	0.1122	0.4395
8	-	x	x	-	80	0.0311	3.2486	0.5132	-0.0637
9	-	x	-	x	82	0.1268	0.9901	0.1603	0.3009
10	x	x	-	-	131	2.3105	1.0479	0.0983	0.0656
11	-	-	x	x	46	0.2261	0.1683	0.3747	0.3051
12	-	-	-	-	968	0.2964	0.3184	0.2418	0.2013
all					4624	0.3230	0.2850	0.2627	0.1905

of the top choices. Therefore it is always obtainable even for an unseen image. A regression model estimated for the particular class can then be applied to combine the rankings.

A set of 1,384 images were used to test this dynamic selector. Five examples from this experiment are given in the Appendix. Table 5.5 summarizes the correct rates of each of the classifiers and their combinations by each method.

Table 5.5: Comparison of results by static and dynamic combination

	% Correct in Top N Choices				
Classifier / Combination	1	2	3	5	10
1) character recognition and postprocessing (poly)	84.9	88.4	90.3	91.2	92.3
2) segmentation-based method (segb)	86.1	90.0	90.9	91.8	92.8
3) word-shape with stroke direction features (sfv)	65.2	74.5	78.5	82.4	85.5
4) word-shape with Baird features (bfv)	50.9	59.0	62.2	66.3	70.9
5) combination by the highest rank	50.9	84.7	96.2	98.6	98.9
6) combination by the Borda count	87.4	95.8	97.2	98.2	99.0
7) combination by static regression model	90.7	96.2	97.5	98.5	99.0
8) combination by dynamically selected model	93.9	97.2	97.9	98.3	99.0
9) after case merging	94.3	97.3	98.0	98.5	99.0
10) oracle	98.1	98.8	99.0	99.1	99.3

Comparing lines 7 and 8 in Table 5.5, we can see that the dynamic selector can further improve the performance over the static logistic regression method. This ex-

periment shows that i) the agreement class is a good indicator of the input condition; and ii) dynamic selection of a combination function using this condition is effective.

The agreement class can also serve as an indicator of confidence in recognition. To illustrate this point, the correct rates of the combined decisions (line 9 in Table 5.5) are broken down by the agreement class and shown in Table 5.6. It can be observed that, errors occur more often for certain classes, say, classes 11 and 12. In particular, class 11 contains many bad images which are difficult to segment. With knowledge from such a table, a subsequent contextual analysis procedure may decide on the number of choices to use from a combined ranking based on the agreement class.

Table 5.6: Breakdown of results by agreement class

Agreement Class	Number of Images	% Correct in Top N Choices				
		1	2	3	5	10
1	456	99.6	100.0	100.0	100.0	100.0
2	83	98.8	100.0	100.0	100.0	100.0
3	257	99.6	99.6	99.6	99.6	99.6
4	59	98.3	100.0	100.0	100.0	100.0
5	59	96.6	100.0	100.0	100.0	100.0
6	17	82.4	88.2	88.2	94.1	94.1
7	15	93.3	100.0	100.0	100.0	100.0
8	26	92.3	100.0	100.0	100.0	100.0
9	10	80.0	100.0	100.0	100.0	100.0
10	271	97.0	99.3	99.6	99.6	99.6
11	24	79.2	91.7	91.7	95.8	95.8
12	107	52.3	72.0	79.4	84.1	90.7
all	1384	94.3	97.3	98.0	98.5	99.0

Chapter 6

Applications to Other Domains

Our theory of multiple classifier systems is applicable to any pattern recognition problem, as long as multiple solutions exist for that problem, and each of them provides a ranking of a given set of classes with respect to an input pattern.

The advantage of using multiple classifiers is best demonstrated in problems that involve a large set of classes. We have seen that word recognition is one such problem, where the size of the lexicon determines the number of classes. Another text recognition problem with this characteristic is the recognition of Chinese characters.

The theory can also be applied to other object recognition problems involving large class sets and multiple approaches, such as face recognition, fingerprint recognition, as well as certain areas of medical diagnosis.

In Section 6.1, an application of multiple classifier systems to handwritten digit recognition is described. In this application, four classifiers are used, each of which outputs a unique class as its decision. That is, the rank scores are given on a two-level scale. Nevertheless, logistic regression analysis is applicable to this special case.

In Section 6.2, a multiple classifier system is used to recognize machine-printed characters. This application shows how logistic regression is applied to evaluate the relative importance of the classifiers.

6.1 Example Application to Handwritten Digit Recognition

This section describes an application of logistic regression to decision combination in handwritten digit recognition. The inputs are binary images of handwritten digits extracted from live mail. They were scanned at 300 pixels per inch, and normalized to 16x16 pixels in size.

Four classifiers are used to recognize the digits. The first is a second order polynomial classifier (referred to as P). It uses pairs of pixel values as features [113]. The second classifier uses a mixed approach (referred to as M) based on the methods described in [102]. The third classifier is based on stroke analysis (referred to as S) [75]. The fourth classifier is based on analysis of chain codes that represent character contours (referred to as C) [25].

In this application, each classifier outputs one class as its decision. The decision can still be represented as a ranking of the class set, though the rank score for each class is binary. For example, if a classifier decides that an input digit is an '8', then the class '8' receives a score '1' and all other classes receive a '0'.

A logistic model was estimated using 6,432 images as a training set. Table 6.1 summarizes the results of parameter estimation by the SAS procedure CATMOD [112] based on this set of data. The results indicate that all the four classifiers have significant contribution to correct classification, since each of them receives a substantial weight at similar levels of estimated standard errors.

The four classifiers and the estimated logistic model were applied to a set of 2,711 test images which are distinct from all those used in classifier training or model estimation. Table 6.2 summarizes the correct rate of each individual classifier, as well as the top choice correct rate of their combination using the estimated model. It

Table 6.1: Analysis of maximum likelihood estimates in logistic regression.

Effect	Parameter	Estimate	Standard Error	Chi-Square	Prob
INTERCEPT	1	-4.7185	0.2086	511.90	0.0000
P	2	2.7592	0.1701	263.15	0.0000
M	3	1.5955	0.1644	94.21	0.0000
S	4	3.2707	0.1727	358.74	0.0000
C	5	1.4672	0.1659	78.24	0.0000

can be seen that the combination performs better than each individual classifier.

A threshold can be applied on the estimated logit of the top class choice for each test image. If the estimated logit for the top choice is below the threshold, the system can refuse to make a decision. To show an example of the tradeoff between the error rate and the reject rate, three thresholds are manually selected. Their effects are shown also in Table 6.2, where both the correct rate and the error rate refer to only the top choice in each combined ranking. The last row in Table 6.2 shows the performance achievable by this collection of classifiers if the best classifier for each input image could be dynamically selected by an oracle.

Table 6.2: Performance of digit classifiers and their combination on test set.

	% Correct	% Error	% Reject
Polynomial	93.6	5.6	0.8
Mixed	84.2	9.0	6.8
Stroke	89.4	9.8	0.8
Chaincode	86.0	2.6	11.4
Logistic Model	95.65	4.35	0.00
with Threshold 1	94.87	2.51	2.62
with Threshold 2	94.21	1.66	4.13
with Threshold 3	73.48	0.04	26.48
Oracle	97.89	2.11	0.00

6.2 Example Application to Machine-Printed Character Recognition

A multiple classifier system is used to recognize machine-printed characters. The system is intended to recognize degraded character images in multiple font types.

Sample images are collected from live mails, scanned at a resolution of 212 pixels per inch, binarized and normalized to 24×24 in size. The characters are in 62 classes, including the upper and lower case English alphabets and the 10 Arabic numerals. Because of similarity in shape after the images are normalized, several groups of classes are merged into single classes. These include the groups {o,0,O}, {1,l,I,i,j}, {c,C}, {p,P}, {s,S}, {u,U}, {v,V}, {w,W}, {x,X}, and {z,Z}. Therefore, only 48 distinct classes are used.

Six classifiers are used to recognize the characters. They use two sets of features and three classification procedures.

The two sets of features include pixel values and Baird template features, and they are described by two vectors. The first vector contains the binary pixel values and has 576 dimensions. The second vector contains thresholded responses obtained by convolving the image with 32 templates designed by Baird [8]. For each of the 32 templates, the 24×24 convolution response map is partitioned into 3×3 cells. The value for a cell is set to be one if there is any positive response to that template in the cell. The resultant vector has $32 \times 3 \times 3 = 288$ dimensions.

Each of the two feature vectors is input to three different classifiers, including a Bayesian classifier with feature independence assumption [33], and two nearest-neighbor classifiers. The Bayesian classifier ranks the classes according to the *a posteriori* probabilities.

Both of the nearest-neighbor classifiers compute a Hamming distance between each input vector and each prototype vector. The two classifiers differ in the way the classes are ranked. In the first classifier, the minimum distance between the input vector and all prototypes of a particular class is taken as the distance of the input to that class. The classes are then ranked by these minimum distances. This classifier

is referred to as the *modified nearest-neighbor classifier.*

In the second classifier, the distance from an input to a class is computed by averaging the distances from the input to the two closest prototypes for that class. The classes are then ranked in ascending order by these averaged distances. This classifier is referred to as the *modified 2-nearest-neighbor classifier.*

Table 6.3 summarizes the six classifiers used. A set of 19,151 sample characters was used to train each of the six classifiers.

Table 6.3: Six classifiers for character recognition.

Name	Features Used	Classification Procedure
PBC	pixel values	Bayesian with independence assumption
PNC	pixel values	modified nearest-neighbor
P2N	pixel values	modified 2-nearest-neighbor
BBC	Baird features	Bayesian with independence assumption
BNC	Baird features	modified nearest-neighbor
B2N	Baird features	modified 2-nearest-neighbor

The six classifiers were applied to 8,000 sample characters which are distinct from those used in classifier training. A logistic regression analysis was performed using the rankings of the 48 classes given by the six classifiers.

For each sample image, an observation vector of the form $< Y, R_{PBC}, R_{PNC}, R_{P2N}, R_{BBC}, R_{BNC}, R_{B2N} >$ was obtained for each class, where Y is 1 if that class is the true class for that image and 0 otherwise, and R_C is the rank assigned to that class by classifier C for that image. The ranks are represented by a descending score, so that R_C is 48 if that class is ranked at the top by classifier C, and R_C is 1 if it is considered the least similar to the input image by C. 48 such vectors were obtained for each image. To simplify the analysis, only the top 10 decisions from each classifier were considered, that is, a vector was used in the analysis only if any of the six rank scores is larger than 38.

Nine different models were attempted in the analysis. Table 6.4 summarizes the results produced by the SAS procedure CATMOD [112]. For the pixel features, the results indicate that the decisions of both PNC and PBC are significant when only these two classifiers are used (model 1). However, when P2N is introduced (model 2), PNC becomes insignificant. Since its weight estimate becomes close to zero, it has almost no influence in promoting the rank of any class, and can therefore be ignored. The results for the Baird feature based classifiers are similar (models 3,4). Results with model 9 shows that when all six classifiers are used, PNC and BNC become insignificant (estimated magnitude of the parameter is small and the standard error is large) and the combination should use only the four other classifiers.

A set of 12,000 samples were used to test the performance of these models. Table 6.5 shows the performance of each classifier and their various combinations by the regression method using parameters given in Table 6.4. The parameter was set to zero if it was determined to be insignificant.

Table 6.5 also gives the performance of the combinations by the Borda count and the highest rank method. Each combination method was applied independently to the classifier outputs. The last row in Table 6.5 shows the theoretical maximum performance achievable if the best classifier for each image could be selected by an oracle.

Note that the combination (PBC,P2N) performs better than either PBC or P2N individually. This indicates that even when the same features are used, different classification procedures give independent decisions that can be combined and improved, though the improvement is not as remarkable as a combination of classifiers that use different features. This suggests that different information contained in the feature vectors is utilized in each classification procedure. Such information is effectively used in a multiple classifier system.

Table 6.4: Analysis of maximum likelihood estimates in logistic regression.

Model	Effect	Parameter	Estimate	Standard Error	Chi-Square	Prob
1	INTERCEPT	1	-23.7605	0.3833	3842.63	0.0000
	PNC	2	0.2951	0.00818	1302.55	0.0000
	PBC	3	0.1861	0.00625	887.27	0.0000
2	INTERCEPT	1	-25.0556	0.4121	3695.84	0.0000
	PNC	2	-0.0107	0.0215	0.25	0.6180
	PBC	3	0.1730	0.00615	790.01	0.0000
	P2N	4	0.3466	0.0238	212.39	0.0000
3	INTERCEPT	1	-24.9299	0.4149	3610.43	0.0000
	BNC	2	0.3425	0.00920	1384.96	0.0000
	BBC	3	0.1633	0.00612	712.72	0.0000
4	INTERCEPT	1	-26.9303	0.4614	3405.92	0.0000
	BNC	2	-0.0100	0.0205	0.24	0.6274
	BBC	3	0.1448	0.00593	597.07	0.0000
	B2N	4	0.4136	0.0235	308.93	0.0000
5	INTERCEPT	1	-21.8316	0.3559	3762.15	0.0000
	PBC	2	0.2277	0.00692	1082.12	0.0000
	BBC	3	0.2107	0.00684	948.43	0.0000
6	INTERCEPT	1	-25.4009	0.4204	3650.28	0.0000
	PNC	2	0.2186	0.00766	814.00	0.0000
	BNC	3	0.2974	0.00910	1068.49	0.0000
7	INTERCEPT	1	-27.5226	0.4652	3499.64	0.0000
	P2N	2	0.2275	0.00845	724.48	0.0000
	B2N	3	0.3337	0.0104	1028.18	0.0000
8	INTERCEPT	1	-26.1740	0.4099	4078.14	0.0000
	PBC	2	0.1085	0.00614	312.51	0.0000
	BBC	3	0.0836	0.00574	212.28	0.0000
	PNC	4	0.1405	0.00764	338.62	0.0000
	BNC	5	0.2076	0.00896	536.83	0.0000
9	INTERCEPT	1	-27.7886	0.4537	3751.44	0.0000
	PBC	2	0.0996	0.00608	268.56	0.0000
	BBC	3	0.0775	0.00563	189.54	0.0000
	PNC	4	-0.0579	0.0220	6.93	0.0085
	BNC	5	-0.0232	0.0212	1.20	0.2741
	P2N	6	0.2130	0.0244	76.40	0.0000
	B2N	7	0.2648	0.0243	119.12	0.0000

Table 6.5: Performance of character classifiers and their combinations on test set.

Model	Classifier(s)	% Correct in Top N Choices					
		1	2	3	4	5	10
	PBC	79.3	87.8	91.2	92.8	94.3	97.6
	PNC	85.3	91.3	93.3	94.7	95.4	97.7
	P2N	85.8	91.9	93.9	94.9	95.7	97.9
	BBC	79.1	87.5	90.7	92.4	93.9	97.1
	BNC	84.3	90.8	93.3	94.6	95.4	97.7
	B2N	85.4	91.7	93.9	95.1	95.7	98.0
	Logistic Regression						
1	PBC,PNC	85.4	92.2	94.5	95.7	96.7	98.5
2	PBC,P2N	86.3	92.8	94.7	95.9	96.8	98.6
3	BBC,BNC	85.3	91.8	94.1	95.3	96.1	98.3
4	BBC,B2N	86.0	92.3	94.5	95.7	96.4	98.4
5	PBC,BBC	81.3	89.8	92.7	94.5	95.8	98.3
6	PNC,BNC	86.7	92.6	94.6	95.6	96.2	98.4
7	P2N,B2N	86.9	92.9	95.0	95.9	96.5	98.5
8	PBC,BBC,PNC,BNC	88.1	93.8	95.7	96.6	97.2	98.8
9	PBC,BBC,P2N,B2N	88.8	94.1	95.7	96.7	97.4	98.9
	Borda Count						
1	PBC,PNC	84.6	91.5	94.2	95.7	96.7	98.4
2	PBC,P2N	84.7	91.7	94.3	95.7	96.6	98.4
3	BBC,BNC	84.0	91.5	93.8	95.3	96.1	98.0
4	BBC,B2N	84.5	91.6	94.0	95.5	96.4	98.3
5	PBC,BBC	82.2	90.2	93.0	94.9	96.1	98.1
6	PNC,BNC	86.9	92.8	94.5	95.6	96.2	98.1
7	P2N,B2N	87.3	93.0	94.9	95.9	96.5	98.4
8	PBC,BBC,PNC,BNC	87.9	93.6	95.6	96.5	97.1	98.8
9	PBC,BBC,P2N,B2N	88.0	93.7	95.7	96.7	97.3	98.8
	Highest Rank						
1	PBC,PNC	83.8	92.2	94.3	95.7	96.4	98.2
2	PBC,P2N	84.1	92.3	94.4	95.8	96.5	98.3
3	BBC,BNC	82.8	91.0	93.5	94.8	95.7	98.1
4	BBC,B2N	83.4	91.3	93.8	95.1	95.8	98.2
5	PBC,BBC	80.7	90.2	93.2	94.6	95.7	98.0
6	PNC,BNC	84.7	92.3	94.2	95.3	96.1	98.0
7	P2N,B2N	85.5	92.9	94.8	95.8	96.5	98.3
8	PBC,BBC,PNC,BNC	79.7	91.5	94.8	96.0	96.9	98.5
9	PBC,BBC,P2N,B2N	80.3	91.6	95.0	96.2	97.0	98.6
	Oracle	94.6	97.6	98.5	99.0	99.2	99.8

From Table 6.5, we can observe improvements over the individual classifiers on the top choice correct rate by each combination using the Borda count or logistic regression. The most significant improvement is obtained by combining four classifiers (PBC, BBC, P2N, B2N), which gives a net increase of 3% on the top choice correct rate over the best individual classifier (P2N). Corresponding improvements are also observed in larger neighborhoods. In most of the cases, the combinations by logistic regression are better than those by the Borda count and the highest rank method.

The performance of the oracle shows that, for 94.6% of all the inputs, there exists a classifier which can correctly recognize the character. Hints for possible improvements may be obtained by analyzing the remaining 5.4% of the inputs. Efforts should be devoted to constructing a classifier that works for those difficult cases.

Chapter 7

Conclusions

7.1 Summary

A theory of multiple classifier systems has been proposed. The theory explains the advantages of using multiple classifiers and specifies methods for combining their decisions.

A multiple classifier system is a preferred solution to a complex pattern recognition problem because it allows simultaneous use of feature descriptors of many types, corresponding measures of similarity, and different classification procedures. It also allows for dynamic selection, so that classifiers adapted to inputs of a particular type are applied only when inputs of that type are encountered.

The theory proposes to use rankings of the class set to represent decisions made by an individual classifier. Such rankings are always derivable from the results of feature matching. Rank scores contain more information than unique class choices, and can be given on a common and consistent scale by each classifier for different instances of the problem.

Decisions by multiple classifiers can be combined by different methods that either reduce or rerank the set of classes. Two methods are proposed for class set reduction. They are the intersection method and the union method. Three methods are proposed for class set reordering. They are the highest rank method, the Borda

count method, and the logistic regression method.

The theory is tested in an application to visual word recognition, where many classes need to be discriminated, and the quality of input is highly variable. The redundancy provided by multiple classifiers is needed to obtain reliable decisions.

A computational model for visual word recognition is proposed that uses three approaches to solve the problem. These include a character recognition approach, a segmentation-based approach, and a word shape analysis approach. A control strategy selectively activates suitable classifiers and candidate words in an input lexicon. The decisions are combined by three functions given in our theory.

In an experiment with 1,671 degraded word images of multiple font types and a 33,850 word lexicon, an implementation of the model achieved a correct rate of 88.9% at the top choice and 95.3% in the top 10 choices. When the lexicon is reduced to 1,000 words, a correct rate of 95.5% at the top choice and 98.7% in the top 10 choices was achieved. The performance of the algorithm is significantly better than those obtained by applying each individual classifier in isolation.

7.2 Future Directions

7.2.1 Decision Combination

In this thesis we have outlined several decision combination methods based on rankings. It will be interesting to study other combination methods that may be useful under special conditions. Selection of an optimal subset of classifiers, by methods other than combinatorial search and regression analysis, will also be interesting. This involves a systematic study of the correlation of errors made by the classifiers. The implications of each possible organization of a multiple classifier system are not yet clear. How to achieve an optimal organization is a challenging open problem.

Dynamic selection of classifiers is likely to be domain specific. Descriptors of conditions are needed that can categorize the patterns to be recognized. The performance of a classifier should be related to these conditions.

The theory is applicable to other domains besides word recognition, and is most useful for recognition problems that involve a large class set and multiple solutions. Examples of such domains include Chinese character recognition, fingerprint recognition, face recognition, speech recognition, and medical diagnosis.

7.2.2 Word Recognition

The proposed solution to word recognition can be extended in the following ways:

Implementation Improvements

Our experimental algorithm has been developed and tested in a laboratory environment where speed is a minor concern. The algorithm needs to be optimized in both speed and storage to be useful in a real-time environment. A better organization of a given lexicon and faster classifiers are among the needs. Various prototype clustering methods are useful to improve the speed of the nearest-neighbor classifiers.

Negative Evidences

All the classifiers included in our solution use a similarity score that can be used to rank the words with respect to an image. There are also some methods that eliminate candidate words which do not match a descriptor of the image. Such procedures may be used to reduce the class set before the rankings are computed.

Hypothesis Testing

Class decisions can also be made by hypothesis testing [66]. The difference between this and a simple matching strategy is that, in hypothesis testing, the features and the matching procedures used are influenced by the hypotheses. Comparison between an input image and the prototypes is focused on only those features that distinguish between the hypothesized words.

Dynamic Adaptation to Font Types

The word recognition algorithm may dynamically adapt to the input by estimating the font type used in the text. Such an estimate need not be as specific as the name of a particular font. Instead, font types that share similar shape characteristics may be grouped into a category. Dynamic adaptation requires that these shape characteristics be computable from an input image.

Application to Handwritten Words

In our study we have restricted the input to be machine-printed images. A more difficult problem is to recognize handwritten words, including cursive script. While the basic ideas in our model still apply, the recognition of handwritten words may need a different set of features and classifiers. The difficulties are in obtaining features that are invariant to different writing styles, and in defining a suitable reference frame to describe the locations of the features within a word. An important distinction between handwritten and machine-printed text is that, in handwritten text, the presense of some characters affects the shapes of others, so that the word shape is not a simple composition of the shapes of the isolated characters.

Approximate String Matching

Our model assumes that the true word is contained in a given lexicon. If this assumption fails, and if the exact form of the word as it appears is not needed, approximate string matching may be used to recover an intended word.

Methods for approximate string matching include those for finding strings that are *equivalent* to the input, such as common roots of words in various grammatical forms and expansions of abbreviations, and those for finding strings that are *similar* to the input, such as the correct form of a misspelled word. Which method to use depends on the needs in a particular application. A comprehensive survey of both kinds of methods is given by Hall and Dowling [49].

Integration with Global Context Analysis

Word recognition may be integrated with an analysis of higher level context in the text. Knowledge from analyses of the language syntax and semantics may be used to resolve ambiguities and improve the recognition correctness. Word recognition also needs to be integrated in a control structure that preprocesses a document and analyzes its layout, in a development towards a complete reading device.

7.2.3 Creating A Multiple Classifier System

In this thesis we have been assuming that there exists a group of classifiers for a given recognition problem. Each of them is based on heuristics or some well-studied theories. The performance of their combination, though can be better than those of each individual, will reach an upper limit if there are cases where none of the classifiers' decision is sufficiently close to correct.

A logical question then arises: is it possible to systematically create a multiple

classifier system for a given problem, so that for each possible input pattern, there exists one or a combination of several computationally feasible classifiers that can correctly identify its true class? If this could be done, a perfect solution could be obtained for any given recognition problem.

A promising approach towards this goal is proposed by Kleinberg [73][74]. We do expect, studies towards this end will lead to a much deeper understanding of the essence of pattern recognition.

References

[1] A. Agresti, *Categorical Data Analysis*, John Wiley & Sons, 1990.

[2] T. Agui, M. Nakajima, E.T. Takahashi, A Method of Recognition and Representation of Korean Characters by Tree Grammars, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-1**, 3, July 1979, 245-251.

[3] A.V. Aho, B.W. Kernighan, P.J. Weinberger, Awk - A Pattern Scanning and Processing Language (Second Edition), in *UNIX User's Manual, Supplementary Documents*, University of California, 1980.

[4] F. Ali, T. Pavlidis, Syntactic Recognition of Handwritten Numerals, *IEEE Transactions on Systems, Man, and Cybernetics*, **SMC-7**, 7, July 1977, 537-541.

[5] J.A. Anderson, Regression and Ordered Categorical Variables, *Journal of the Royal Statistical Society*, **B46**, 1, 1984, 1-30.

[6] K.J. Arrow, *Social Choice and Individual Values*, John Wiley & Sons, Inc., New York, 1951, 2nd ed., 1963.

[7] H.S. Baird, Feature Identification for Hybrid Structural/Statistical Pattern Classification, *Computer Vision, Graphics, & Image Processing*, **42**, 1988, 318-333.

[8] H.S. Baird, H.P. Graf, L.D. Jackel, W.E. Hubbard, A VLSI Architecture For Binary Image Classification, in J.C. Simon, ed., *From Pixels to Features* , North Holland, 1989, 275-286.

[9] H.S. Baird, K. Thompson, Reading Chess, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-12**, 6, June 1990, 552-559.

[10] H.S. Baird, A 100-Font Classifier, *Proceedings of the First International Conference on Document Analysis and Recognition*, Saint-Malo, France, 1991, 332-340.

[11] D.H. Ballard, C.M. Brown, *Computer Vision*, Prentice-Hall, 1982.

[12] T. Bayer, Segmentation of Merged Character Patterns with AI Techniques, *Proceedings of the Scandinavian Conference on Image Analysis*, 1987.

[13] D. Black, *The Theory of Committees and Elections*, Cambridge University Press, London, 1958, 2nd ed., 1963.

[14] K.P. Bogart, Preference Structures I: Distances between Transitive Preference Relations, *Journal of Mathematical Sociology*, **3**, 1973, 49-67.

[15] K.P. Bogart, Preference Structures II: Distances Between Asymmetric Relations, *SIAM Journal of Applied Mathematics*, **29**, 2, September 1975, 254-262.

[16] R. Bozinovic, S.N. Srihari, A String Correction Algorithm for Cursive Script Recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-4**, 6, November 1982, 655-663.

[17] R. Bradford, T. Nartker, Error Correlation in Contemporary OCR Systems, *Proceedings of the First International Conference on Document Analysis and Recognition*, Saint-Malo, France, 1991, 516-523.

[18] M.K. Brown, S. Ganapathy, Cursive Script Recognition, *Proceedings of the International Conference on Cybernetics and Society*, Cambridge, Massachusettes, October 1980, 47-51.

[19] D.J. Burr, Elastic Matching of Line Drawings, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-3**, 6, November 1981, 708-713.

[20] R.G. Casey, G. Nagy, Recursive Segmentation and Classification of Composite Character Patterns, *Proceedings of the Sixth International Conference on Pattern Recognition*, Munich, 1982, 1023-1026.

[21] R.G. Casey, G. Nagy, Decision Tree Design Using a Probabilistic Model, *IEEE Transactions on Information Theory*, **IT-30**, 1, January 1984, 93-99.

[22] G.L. Cash, M. Hatamian, Optical Character Recognition by the Method of Moments, *Computer Vision, Graphics, and Image Processing*, **39**, 1987, 291-310.

[23] K.R. Castleman, *Digital Image Processing*, Prentice-Hall, 1979.

[24] J.K. Cattell, The Time It Takes to See and Name Objects, *Mind*, **11**, 1886, 63-65.

[25] A. Commike, J.J. Hull, Rule Learning for Syntactic Pattern Recognition, *Proceedings of Fourth United States Postal Service Advanced Technology Conference*, November 1990, 621-633.

[26] D.R. Cox, E.J. Snell, *Analysis of Binary Data*, 2nd. ed., Chapman and Hall, 1989.

[27] T.M. Cover, P.E. Hart, Nearest Neighbor Pattern Classification, *IEEE Transactions on Information Theory*, **IT-13**, January 1967, 21-27.

[28] F.J. Damerau, A Technique for Computer Detection and Correction of Spelling Errors, *Communications of the ACM*, **7**, 5, March 1964, 171-176.

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

[30] K. Deguchi, Multi-scale Curvatures for Contour Feature Extraction, *Proceedings of the Nineth International Conference on Pattern Recognition*, November, 1988, 1113-1115.

[31] P.A. Devijver, J. Kittler, *Pattern Recognition: A Statistical Approach*, Prentice-Hall, 1982.

[32] W. Doster, Contextual Postprocessing System for Cooperation with a Multiple-Choice Character-Recognition System, *IEEE Transactions on Computers*, **C-26**, 11, November 1977, 1090-1101.

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

[34] R.A. Duderstadt, M.J. Cykana, A.V. Monfared, and D.H. Gibbs, Isolated Word Recognition for Postal Address Processing, *Proceedings of the Fourth USPS Advanced Technology Conference*, Washington, November 1990, 233-245.

[35] B. Duerr, W. Haettich, H. Tropf, G. Winkler, A Combination of Statistical and Syntactical Pattern Recognition Applied to Classification of Unconstrained Handwritten Numerals, *Pattern Recognition*, **12**, 3, 1980, 189-199.

[36] B. Efron, The Efficiency of Logistic Regression Compared to Normal Discriminant Analysis, *Journal of the American Statistical Association*, **70**, 352, December 1975, 892-898.

[37] D.G. Elliman, I.T. Lancaster, A Review of Segmentation and Contextual Analysis Techniques for Text Recognition, *Pattern Recognition*, **23**, 3/4, 1990, 337-346.

[38] B. Erdmann, R. Dodge, in *R. Psychologischen untersuchungen uber das lesen, auf experimenteller grundlage*, Halle, 1898, as reported in E.B. Huey, *The Psychology and Pedagogy of Reading*, MIT Press, 1968 (originally published in 1908).

[39] R.F.H. Farag, Word-Level Recognition of Cursive Script, *IEEE Transactions on Computers*, **C-28**, 2, February 1979, 172-175.

[40] P.C. Fishburn, *The Theory of Social Choice*, Princeton University Press, Princeton, 1972.

[41] G.D. Forney, Jr., The Viterbi Algorithm, *Proceedings of the IEEE*, **61**, March 1973, 268-278.

[42] E. Fredkin, Trie Memory, *Communications of the ACM*, **3**, 9, 1980, 490-499.

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

[44] K.S. Fu, A Step Towards Unification of Syntactic and Statistical Pattern Recognition, *IEEE Transaction of Pattern Analysis and Machine Intelligence*, **PAMI-5**, March 1983, 200-205.

[45] T.D. Garvey, A Survey of AI Approaches to the Integration of Information, *Proceedings of SPIE, Volume 782, Infrared Sensors and Sensor Fusion*, 1987.

[46] L.A. Goodman, H. Markowitz, Social Welfare Functions Based on Individual Rankings, *The American Journal of Sociology*, **58**, 1952, 257-262.

[47] A. Goshtasby, R.W. Ehrich, Contextual Word Recognition Using Probabilistic Relaxation Labeling, *Pattern Recognition*, **21**, 5, 1988, 455-462.

[48] G.H. Granlund, Fourier Preprocessing for Hand Print Character Recognition, *IEEE Transactions on Computers*, **C-21**, February 1972, 195-201.

[49] P.A.V. Hall, G.R. Dowling, Approximate String Matching, *ACM Computing Surveys*, **12**, December 1980, 17-38.

[50] J.B. Hampshire II, B.A. Pearlmutter, Equivalence Proofs for Multi-Layer Perceptron Classifiers and the Bayesian Discriminant Function, in Touretzky, Elman, Sejnowski, and Hinton, eds., *Proceedings of the 1990 Connectionist Models Summer School*, San Mateo, Morgan Kaufmann, 1990.

[51] G. Hannan, The RCA Multi-font Reading Machine, in G.L. Fisher, Jr., D.K. Pollock, B. Raddack, and M.E. Stevens, eds., *Optical Character Recognition*, Spartan Books, 1962, 3-14.

[52] L.K. Hansen, P. Salamon, Neural Network Ensembles, *IEEE Transaction of Pattern Analysis and Machine Intelligence*, **PAMI-12**, 10, October 1990, 993-1001.

[53] R.M. Haralick, The Table Look-Up Rule, *Communications in Statistics - Theory and Methods*, **A5**, 12, 1976, 1163-1191.

[54] K.C. Hayes, Jr., Reading Handwritten Words Using Hierarchical Relaxation, *Computer Graphics and Image Processing*, **14**, 1980, 344-364.

[55] T.P. Hettmansperger, *Statistical Inference Based on Ranks*, John Wiley & Sons, 1984.

[56] T.K. Ho, J.J. Hull, S.N. Srihari, Combination of Structural Classifiers, *Pre-Proceedings of the IAPR Syntactic and Structural Pattern Recognition Workshop*, Murray Hill, June 1990, 123-136.

[57] T.K. Ho, J.J. Hull, S.N. Srihari, A Word Shape Analysis Approach to Recognition of Degraded Word Images, *Proceedings of the Fourth USPS Advanced Technology Conference*, Washington, November 1990, 217-231.

[58] T.K. Ho, J.J. Hull, S.N. Srihari, Word Recognition with Multi-Level Contextual Knowledge, *Proceedings of the First International Conference on Document Analysis and Recognition*, Saint-Malo, France, 1991, 905-915.

[59] T.K. Ho, J.J. Hull, S.N. Srihari, A Regression Approach to Combination of Decisions by Multiple Character Recognition Algorithms, *Proceedings of the SPIE/IS&T's Symposium on Electronic Imaging: Science & Technology*, Conference 1661, Machine Vision Applications in Character Recognition and Industrial Inspection, San Jose, February 1992.

[60] R.L. Hoffman, J.W. McCullough, Segmentation Methods for Recognition of Machine-printed Characters, *IBM Journal of Research and Development*, March 1971, 153-165.

[61] D.R. Hofstadter, *Metamagical Themas: Questing for the Essence of Mind and Pattern*, Basic Books, Inc., 1985.

[62] D.W. Hosmer, S. Lemeshow, *Applied Logistic Regression*, John Wiley & Sons, 1989.

[63] J.S. Huang, M.L. Chung, Separating Similar Complex Chinese Characters by Walsh Transform, *Pattern Recognition*, **20**, 4, 1987, 425-428.

[64] J.J. Hull, S.N. Srihari, Experiments in Text Recognition with Binary n-Gram and Viterbi Algorithms, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-4**, 5, September 1982, 520-530.

[65] J.J. Hull, S.N. Srihari, R. Choudhuri, An Integrated Algorithm for Text Recognition: Comparison with a Cascaded Algorithm, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-5**, 4, July 1983, 384-395.

[66] J.J. Hull, Hypothesis Testing in a Computational Theory of Visual Word Recognition, *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI)*, Seattle, July 1987, 718-722.

[67] J.J. Hull, *A Computational Theory of Visual Word Recognition*, Doctoral Dissertation, Technical Report 88-07, Department of Computer Science, SUNY at Buffalo, February 1988.

[68] J.J. Hull, S.N. Srihari, E. Cohen, C.L. Kuan, P. Cullen and P. Palumbo, A Blackboard-Based Approach to Handwritten ZIP Code Recognition, *Proceedings of the Third USPS Advanced Technology Conference*, Washington, May 1988, 1018-1032.

[69] J.J. Hull, A. Commike, T.K. Ho, Multiple Algorithms for Handwritten Character Recognition, *Proceedings of the First International Workshop on Frontiers in Handwriting Recognition*, Montreal, April 1990, 117-124.

[70] S. Kahan, T. Pavlidis, H.S. Baird, On the Recognition of Printed Characters of Any Font and Size, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-9**, 2, March 1987, 274-288.

[71] R.L. Kashyap, B.J. Oommen, Spelling Correction Using Probabilistic Methods, *Pattern Recognition Letters*, **2**, 4, 1984, 147-154.

[72] F. Kimura, M. Shridhar, Handwritten Numerical Recognition Based on Multiple Algorithms, *Pattern Recognition*, **24**, 10, 1991, 969-983.

[73] E.M. Kleinberg, Stochastic Discrimination, *Annals of Mathematics and Artificial Intelligence*, **1**, 1990, 207-239.

[74] E.M. Kleinberg, The Theory of Stochastic Modeling in Pattern Recognition, in preparation.

[75] C.C.L. Kuan, S.N. Srihari, A Stroke-Based Approach to Handwritten Numeral Recognition, *Proceedings of Third USPS Advanced Technology Conference*, Washington, 1988, 1033-1041.

[76] A. Kundu, Y. He, P. Bahl, Recognition of Handwritten Word: First and Second Order Hidden Markov Model Based Approach, *Pattern Recognition*, **22**, 3, 1989, 283-297.

[77] R. Kurzweil, *The Age of Intelligent Machines*, Chapter VII, The MIT Press, 1990.

[78] M.W. Kurzynski, The Optimal Strategy of a Tree Classifier, *Pattern Recognition*, **16**, 1, 1983, 81-87.

[79] J.M. Landwehr, D. Pregibon, A.C. Shoemaker, Graphical Methods for Assessing Logistic Regression Models, *Journal of the American Statistical Association*, **79**, 385, March 1984, 61-71.

[80] M.S. Landy, Y. Cohen, G. Sperling, HIPS: A UNIX-Based Image Processing System, *Computer Vision, Graphics, and Image Processing*, **25**, 1984, 331-347.

[81] S. le Cessie, *Model Building Techniques for Logistic Regression, with Applications to Medical Data*, Doctoral Dissertation, University of Leiden, The Netherlands, 1991.

[82] D.S. Lee, S.W. Lam, S.N. Srihari, A Structural Approach to Recognize Hand-printed and Degraded Machine-printed Characters, *Pre-Proceedings of the IAPR Syntactic and Structural Pattern Recognition Workshop*, Murray Hill, June 1990, 256-272.

[83] H.H. Liu, K.S. Fu, A Syntactic Approach to Seismic Pattern Recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-4**, 2, March 1982, 136-140.

[84] R. Lowrance, R.A. Wagner, An Extension of the String to String Correction Algorithm, *Journal of the ACM*, **22**, 2, April 1975, 177-183.

[85] R.C. Luo, M.G. Kay, Multisensor Integration and Fusion: Issues and Approaches, *Proceedings of SPIE, Volume 931, Sensor Fusion*, 1988.

[86] E. Mandler, J. Schuermann, Combining the Classification Results of Independent Classifiers Based on the Dempster/Shafer Theory of Evidence, in E.S. Gelsema and L.N. Kanal, eds., *Pattern Recognition and Artificial Intelligence*, North Holland, 1988, 381-393.

[87] J. Mantas, An Overview of Character Recognition Methodologies, *Pattern Recognition*, **19**, 6, 1986, 425-430.

[88] V.D. Mazurov, A.I. Krivonogov, V.L. Kazantsev, Solving of Optimization and Identification Problems by the Committee Methods, *Pattern Recognition*, **20**, 4, 1987, 371-378.

[89] J.L. McClelland, D.E. Rumelhart, An Interactive Activation Model of Context Effects in Letter Perception: Part 1. An Account of the Basic Findings, *Psychological Review*, **88**, 5, September 1981, 375-407.

[90] R. Meddis, *Statistics Using Ranks, A Unified Approach*, Basil Blackwell, 1984.

[91] M. Minsky, Logical Versus Analogical or Symbolic Versus Connectionist or Neat Versus Scruffy, *AI Magazine*, **12**, 2, Summer 1991, 34-51.

[92] B.M.E. Moret, Decision Trees and Diagrams, *ACM Computing Surveys*, **14**, December 1982, 593-623.

[93] S. Mori, K. Yamamoto, M. Yasuda, Research on Machine Recognition of Handprinted Characters, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-6**, 4, July 1984, 386-405.

[94] C. Nadal, R. Legault, C.Y. Suen, Complementary Algorithms for the Recognition of Totally Unconstrained Handwritten Numerals, *Proceedings of Tenth International Conference on Pattern Recognition*, Atlantic City, 1990, 443-449.

[95] G. Nagy, State of the Art in Pattern Recognition, *Proceedings of the IEEE*, **56**, 836, 1968.

[96] G. Nagy, Optical Character Recognition, Theory and Practice, in P.R. Krishnaiah, L.N. Kanal, eds., *Handbook of Statistics*, **2**, North Holland, 1982, 621-649.

[97] D.L. Neuhoff, The Viterbi Algorithm as an Aid in Text Recognition, *IEEE Transactions on Information Theory*, **IT-21**, 1975, 222-226.

[98] N.J. Nilsson, *Learning Machines*, McGraw-Hill, 1965.

[99] T. Okuda, E. Tanaka, T. Kasai, A Method for the Correction of Garbled Words Based on the Levenshtein Metric, *IEEE Transactions on Information Theory*, **IT-25**, 2, February 1976, 172-178.

[100] E.A. Patrick, F.P. Fisher, III, A Generalized k-Nearest Neighbor Rule, *Information and Control*, **16**, 2, April 1970, 128-152.

[101] T. Pavlidis, F. Ali, Computer Recognition of Handwritten Numerals by Polygonal Approximations, *IEEE Transactions on Systems, Man, and Cybernetics*, **SMC-5**, 6, November 1975, 610-614.

[102] T. Pavlidis, F. Ali, A Hierarchical Syntactic Shape Analyzer, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-1**, 1, January 1979, 2-9.

[103] T. Pavlidis, A Vectorizer and Feature Extractor for Document Recognition, *Computer Vision, Graphics, and Image Processing*, **35**, 1986, 111-127.

[104] J. Pearl, Bayesian Decision Methods, in *Encyclopedia of Artificial Intelligence*, Wiley Interscience, 1987.

[105] A. Pervez, C.Y. Suen, Segmentation of Unconstrained Handwritten Numeric Postal ZIP Codes, *Proceedings of the Sixth International Conference on Pattern Recognition*, October 1982, 545-547.

[106] S.J. Press, S. Wilson, Choosing Between Logistic Regression and Discriminant Analysis, *Journal of the American Statistical Association*, **73**, 364, December 1978, 699-705.

[107] E.M. Riseman, R.W. Ehrich, Contextual Word Recognition Using Binary Diagrams, *IEEE Transactions on Computers*, **C-20**, 4, April 1971, 397-403.

[108] E.M. Riseman, R.W. Ehrich, A Contextual Postprocessing System for Error Correction Using Binary N-grams, *IEEE Transactions on Computers*, **C-23**, 5, May 1974, 480-493.

[109] A.K. Romney, R.H. Shepard, and S.B. Nerlove, *Multidimensional Scaling, Theory and Applications in the Behavioral Sciences*, Vol. I and II, Seminar Press, 1972.

[110] D.W. Ruck, S.K. Rogers, M. Kabrisky, M.E. Oxley, B.W. Suter, The Multilayer Perceptron as Approximations to a Bayes Optimal Discriminant Function, *IEEE Transactions on Neural Networks*, **1**, 4, December 1990, 296-298.

[111] D.E. Rumelhart, J.L. McClelland, An Interactive Activation Model of Context Effects in Letter Perception: Part 2. The Contextual Enhancement Effect and Some Tests and Extensions of the Model, *Psychological Review*, **89**, 1, 1982, 60-94.

[112] SAS Institute Inc., *SAS/STAT User's Guide*, Version 6, Fourth Edition, Volumn 2, SAS Institute Inc., 1989.

[113] J. Schuermann, A Multifont Word Recognition System for Postal Address Reading, *IEEE Transactions on Computers*, **C-27**, 8, August 1978, 721-732.

[114] J. Schuermann, Reading Machines, *Proceedings of the Sixth International Conference on Pattern Recognition*, Munich, Germany, 1982, 1031-1044.

[115] J. Schuermann, W. Doster, A Decision Theoretic Approach to Hierarchical Classifier Design, *Pattern Recognition*, **17**, 3, 1984, 359-369.

[116] I.K. Sethi, G.P.R. Sarvarayudu, Hierarchical Classifier Design Using Mutual Information, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-4**, 4, July 1982, 441-445.

[117] G. Shafer, *A Mathematical Theory of Evidence*, Princeton University Press, 1976.

[118] D. Sher, Generating Robust Operators from Specialized Ones, *Proceedings of Workshop on Computer Vision*, IEEE, Miami Beach, November 1987, 301-303.

[119] D. Sher, Tunable Facet Model Likelihood Generators for Boundary Pixel Detection, *Proceedings of Workshop on Computer Vision*, IEEE, Miami Beach, November 1987, 35-40.

[120] R. Shinghal, K.J. Siddiqui, A Bottom-up and Top-down Approach to Using Context in Text Recognition, *International Journal of Man-Machine Studies*, **11**, 1979, 201-212.

[121] R. Shinghal, G.T. Toussaint, Experiments in Text Recognition with the Modified Viterbi Algorithm, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-1**, 2, 1979, 184-193.

[122] R. Shinghal, A Hybrid Algorithm for Contextual Text Recognition, *Pattern Recognition*, **16**, 2, 1983, 261-267.

[123] M. Shridhar, A. Badreldin, High Accuracy Character Recognition Algorithm Using Fourier and Topological Descriptors, *Pattern Recognition*, **17**, 5, 1984, 515-524.

[124] M. Shridhar, A. Badreldin, Recognition of Isolated and Simple Connected Handwritten Numerals, *Pattern Recognition*, **19**, 1, 1986, 1-12.

[125] M. Shridhar, A. Badreldin, Context-Directed Segmentation Algorithm for Handwritten Numeral Strings, *Image and Vision Computing*, **5**, 1, 1987, 3-9.

[126] J.C. Simon, *La Reconnaissance des Formes par Algorithmes*, Masson, Editeur, Paris, 1984; English translation by J. Howlett, *Patterns and Operators - The Foundations of Data Representation*, North Oxford Academic Publishers, 1986.

[127] J.C. Simon, Handwriting Recognition as an Application of Regularities and Singularities in Line Pictures, *Proceedings of the First International Workshop on Frontiers in Handwriting Recognition*, Montreal, April 1990, 23-36.

[128] R.M.K. Sinha, B. Prasada, Visual Text Recognition through Contextual Processing, *Pattern Recognition*, **21**, 5, 1988, 463-479.

[129] S.N. Srihari, Reliability Analysis of Majority Vote Systems, *Information Sciences*, **26**, 1982, 243-256.

[130] S.N. Srihari, J.J. Hull, R. Choudhari, Integrating Diverse Knowledge Sources in Text Recognition, *ACM Transactions on Office Information Systems*, **1**, 1, January 1983, 68-87.

[131] S.N. Srihari, ed., *Computer Text Recognition And Error Correction*, IEEE Computer Society Press, Silver Spring, Maryland, 1984.

[132] F.W.M. Stentiford, Automatic Feature Design for Optical Character Recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-7**, 3, May 1985, 349-355.

[133] L. Stringa, LCD: A Formal Language for Constraint-Free Hand-Printed Character Recognition, *Proceedings of the Fourth International Conference on Pattern Recognition*, November 1978, 354-358.

[134] C.Y. Suen, N-Gram Statistics for Natural Language Understanding and Text Processing, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-1**, 2, April 1979, 164-172.

[135] C.Y. Suen, C. Nadal, T.A. Mai, R. Legault, and L. Lam, Recognition of Totally Unconstrained Handwritten Numerals Based on the Concept of Multiple Experts, *Proceedings of the First International Workshop on Frontiers in Handwriting Recognition*, Montreal, April 1990, 131-143.

[136] C.C. Tappert, C.Y. Suen, T. Wakahara, The State of the Art in On-Line Handwriting Recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-12**, 8, August 1990, 787-808.

[137] C.H. Teh, R.T. Chin, On Image Analysis by the Methods of Moments, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-10**, 4, July 1988, 496-513.

[138] Y. Tsuji, K. Asai, Character Image Segmentation, *SPIE Proceedings on Applications of Digital Image Processing VII*, **405**, 1984, 2-9.

[139] A.J. Viterbi, Error Bounds for Convolutional Codes and an Asymptomatically Optimum Decoding Algorithm, *IEEE Transactions on Information Theory*, **IT-13**, 2, 1967, 260-269.

[140] R.A. Wagner, M.J. Fischer, The String to String Correction Problem, *Journal of ACM*, **21**, 1, 1974, 168-173.

[141] K.Y. Wong, R.G. Casey, F.M. Wahl, Document Analysis System, *IBM Journal of Research and Development*, **26**, 6, November 1982, 647-656.

[142] A.K.C. Wong, Structural Pattern Recognition: A Random Graph Approach, in P.A. Devijver, J. Kittler, eds., *Pattern Recognition Theory and Applications*, NATO ASI Series, **F30**, Springer-Verlag Berlin Heidelberg, 1987.

Appendix: Word Recognition Examples

Five examples from an experiment described in Section 5.4 are given in this appendix. For each image, each classifier produces an initial ranking of a given lexicon with 2,707 entries. Four different combination methods are applied independently to the initial rankings. The top 10 choices in each ranking are shown. Note that there are some ties in the rankings that are broken arbitrarily.



```
Character Decisions - Top 5 choices and confidences:

        character 1    ITXV1  0.819  0.383  0.241  0.145  0.106
        character 2    rMPe7  0.527  0.467  0.286  0.278  0.211
        character 3    ReWVM  0.406  0.383  0.340  0.279  0.197
        character 4    iI6VX  0.537  0.387  0.300  0.237  0.221
        character 5    nAeR5  0.517  0.372  0.242  0.202  0.161
        character 6    04W3E  0.459  0.375  0.354  0.339  0.207

Rankings by 4 Classifiers:                Combined Rankings:

                                          highest    Borda    static   dynamic
poly     segb     sfv        bfv          rank       count    model    model

Irving   Irving   Irving     Irving       Irving     Irving   Irving   Irving
Fresno   Loving   Hedwig     Inwood       Inwood     Loving   Loving   Loving
INDIAN   Sring    Lincolnton MANAYUNK     Hedwig     Indian   IRVING   Hedwig
IRVING   IRVING   Loving     MCKNIGHT     Loving     IRVING   Indian   Indian
Tamina   Ington   Indian     MONAHANS     Fresno     Tamina   Fresno   Lincolnton
Greens   Spring   Leanna     MONTAGUE     Sring      Spring   Tamina   Leanna
Indian   Indian   Mexquite   NAVARRO      Lincolnton Sring    INDIAN   IRVING
Spring   Tamina   HEALTH     Insurance    MANAYUNK   Inwood   Spring   Sring
Island   SPRING   Fredonia   MCCAMEY      INDIAN     Fresno   Hedwig   Mexquite
Loving   Taylor   Broadway   ANSONIA      IRVING     Hedwig   Sring    Inwood


Agreement Class = 1 : all_agree
```

Remarks. Though there is a little background noise in this image, all characters are isolated and well-formed. All the four classifiers are able to make a correct top choice. However, we can observe that their other choices are different.



```
Character Decisions - Top 5 choices and confidences:

        character 1    MWEwh  0.384  0.276  0.247  0.176  0.165
        character 2    aMtdh  0.475  0.388  0.343  0.219  0.212
        character 3    4elSA  0.406  0.359  0.294  0.259  0.230
        character 4    -      -      -      -      -      -

Rankings by 4 Classifiers:                Combined Rankings:

                                          highest    Borda      static     dynamic
poly     segb     sfv        bfv          rank       count      model      model

Mail     ACHIE    Atlanta    Atlanta      Atlanta    Atlanta    Atlanta    Atlanta
Wall     Admin    McCamey    PEASTER      Mail       PEASTER    Mail       PEASTER
East     Auto     ALTANTA    HELSTON      ACHIE      HEIGHTS    Wall       McCamey
Manf     ACRES    Richarson  Balcones     PEASTER    LBB        ACHIE      ALTANTA
Dall     Austin   Altanta    Itasca       Wall       Altanta    East       Richarson
Fair     NORTH    JEDDO      WEBSTER      McCamey    ACHIE      Admin      HELSTON
Gail     FORTH    Dickerson  DEBERRY      Admin      Mail       McCamey    Mail
Palo     ADMIN    HEIGHTS    LBB          ALTANTA    McCamey    Auto       Altanta
Palm     WORTH    PEASTER    BELLEVUE     East       Admin      Manf       Balcones
Hall     Arcola   BOGATA     HEIGHTS      HELSTON    Wall       PEASTER    Wall


Agreement Class = 11 : sfv_bfv_agree
```

Remarks. All the characters are touching in this image. This obviously causes problems in segmentation, and the character decisions are far from correct. However, the word shape based methods (*sfv* and *bfv*) are unaffected, and both of them produce a correct top choice. A correct decision is obtained by the combination methods. We can see the differences in the combined rankings produced by the static regression model and the dynamically selected model: the word shape classifiers have more influence on the ranking combined by the dynamic model. This is because the agreement class suggests a model that puts more weights on the word shape classifiers (see Table 5.4 for the parameters estimated for this agreement class).



```
Character Decisions - Top 5 choices and confidences:

        character 1    AlXWO  0.653  0.331  0.257  0.199  0.186
        character 2    AX1Wl  0.559  0.429  0.216  0.211  0.201
        character 3    E2DX6  0.646  0.483  0.258  0.206  0.156
        character 4    SK9Ee  0.536  0.261  0.237  0.193  0.182
        character 5    tIE24  0.488  0.386  0.225  0.219  0.199
        character 6    1A2t7  0.653  0.381  0.293  0.229  0.169
        character 7    U0uDL  0.480  0.361  0.284  0.278  0.265
        character 8    nT3AB  0.385  0.357  0.238  0.207  0.170
        character 9    -      -      -      -      -      -
        character 10   41iEg  0.483  0.354  0.322  0.236  0.166

Rankings by 4 Classifiers:                Combined Rankings:

                                          highest    Borda      static     dynamic
poly       segb   sfv          bfv        rank       count      model      model

WAXAHACHIE Manvel MESQUITE     MESQUITA   MESQUITA   MESQUITA   MESQUITE   MESQUITE
TARKINGTON MUTUAL MESQUITA     MESQUITE   MESQUITE   MESQUITE   MESQUITA   MESQUITA
GARDENDALE Mount  MEXQUITE     MEXQUITE   WAXAHACHIE MEXQUITE   MEXQUITE   MEXQUITE
WASHINGTON Nocona LEESBURG     WADSWORTH  Manvel     WADSWORTH  WAXAHACHIE Manvel
SWEETWATER HUGHES MERCANTILE   MARSHALL   TARKINGTON WASHINGTON TARKINGTON WAXAHACHIE
KENSINGTON Mutual HEDWIG       MARYSVILLE MUTUAL     WAXAHACHIE Manvel     MUTUAL
MANSFILED  Mound  Mesquite     MCNAIR     MEXQUITE   Manvel     GARDENDALE TARKINGTON
Atascocita MOORES BRECKENRIDGE MCKINNEY   GARDENDALE TARKINGTON MUTUAL     Mount
ATASCOCITA Ponder WESTLAKE     MOUNTAIN   Mount      MUTUAL     Mount      GARDENDALE
CARROLLTON Marden DESOTO       MCCAMEY    LEESBURG   MANVEL     WASHINGTON Nocona


Agreement Class = 12 : all_disagree
```

Remarks. The variable pitch in this image causes problems in character segmentation. The first character is split into two, which are recognized separately. Errors of this kind are difficult to correct in contextual postprocessing. Again, the word shape classifiers are unaffected. However, even those two classifiers do not agree on their top choices. In this case, both the static and dynamic regression models favor the decision by the classifier *sfv*.



```
Character Decisions - Top 5 choices and confidences:

        character 1    AaP4G  0.557  0.402  0.202  0.199  0.187
        character 2    N37LS  0.506  0.460  0.274  0.269  0.205
        character 3    T1IrF  0.428  0.381  0.367  0.362  0.220
        character 4    0OSTa  0.448  0.304  0.278  0.244  0.216
        character 5    P9r08  0.491  0.318  0.252  0.212  0.204
        character 6    I1TLR  0.822  0.322  0.275  0.159  0.123
        character 7    J0DOE  0.500  0.483  0.255  0.201  0.176

Rankings by 4 Classifiers:              Combined Rankings:

                                        highest    Borda      static     dynamic
poly     segb    sfv         bfv        rank       count      model      model

Aviation ANTONIO ASTROWORLD  WOODBRANCH WOODBRANCH ANTONIO    ANTONIO    ANTONIO
ANTONIO  Laporte WATAUAGA    WASHINGTON ASTROWORLD ANSONIA    Aviation   Aviation
AVIATION Antonio CREDIT      WOODLANDS  Aviation   WOODBRANCH ANSONIA    Laporte
ANSONIA  Arcadia PHANO       MONTGOMERY ANTONIO    Aviation   ASTROWORLD ANSONIA
ANGLETON Elysian BOULEVARD   RAYWOOD    WASHINGTON ASTROWORLD AVIATION   Antonio
ABINGTON Atlanta FIELD       WYNDMOOR   WATAUAGA   WATAUAGA   Laporte    ASTROWORLD
Colonial Patchin ATLANTA     SOUTHMAYD  Laporte    Laporte    WATAUAGA   AVIATION
Paradise Altanta ASTRODOME   COMPANY    WOODLANDS  WASHINGTON Antonio    Arcadia
ALLENTON ANSONIA CARROLLTON  MONTOPOLIS CREDIT     CHICOTA    CREDIT     WATAUAGA
Memorial Baldwin BLDG        WAXAHACHIE Antonio    WOODLANDS  Arcadia    WOODBRANCH


Agreement Class = 12 : all_disagree
```

Remarks. This image is in a constant pitch font. The character segmentation is correct, and the decisions are very close to correct. However, the word shape classifiers do not perform well for this image, because the degradation in the strokes distorts the distribution of features across the image. Though, we can see that the errors they make are uncorrelated, and the combination functions are able to obtain a correct decision.




```
Character Decisions - Top 5 choices and confidences:

        character 1    ITl1t  0.620  0.449  0.352  0.314  0.178
        character 2    E1Ru7  0.498  0.269  0.256  0.229  0.229
        character 3    0OoeA  0.554  0.490  0.448  0.185  0.173
        character 4    T1lAL  0.510  0.431  0.355  0.305  0.261
        character 5    1lAI2  0.490  0.481  0.396  0.326  0.212
        character 6    Ss5FP  0.765  0.246  0.228  0.219  0.206
        character 7    IiT5N  0.609  0.462  0.329  0.176  0.156
        character 8    rI4Li  0.425  0.354  0.326  0.296  0.280
        character 9    l71AD  0.564  0.413  0.305  0.285  0.174
        character 10   nRD0H  0.673  0.345  0.261  0.231  0.178

Rankings by 4 Classifiers:                Combined Rankings:

                                          highest     Borda       static      dynamic
poly        segb      sfv      bfv        rank        count       model       model

Commission  Stockton  Division MONROE     MONROE      Houston     Houston     Houston
Wallisville STOCKTON  Houston  Houston    Division    DOTSON      Commission  Stockton
Hallsville  Rlington  Dotson   MIDTOWN    Stockton    Division    Wallisville Commission
Lewisville  Franklin  Huron    MORTON     Commission  Commission  Stockton    STOCKTON
Jollyville  Crossing  DIVISION Power      Houston     Stockton    Division    Wallisville
Poolville   Pickton   Colony   FORTH      STOCKTON    MONROE      Hallsville  Rlington
Scottsville Trenton   Brushy   RHAWNHURST Wallisville Wallisville STOCKTON    Division
LEWISVILLE  Arlington Harleton DOTSON     MIDTOWN     Harleton    Rlington    Hallsville
Eliasville  Princeton Bryson   RLINGTON   Rlington    STOCKTON    Lewisville  Franklin
Torresdale  Greggton  WINDOM   Ftworth    Dotson      Rlington    Dotson      Lewisville


Agreement Class = 12 : all_disagree
```

Remarks. The broken characters in this image cause problems in character segmentation and recognition. In fact, none of the classifiers produces a correct top choice. Though, two of them produce a correct second choice, and the top choice is correct in the combined rankings by the Borda count method as well as by the two regression models.