A Comparison of Approaches to On-Line Handwritten Character Recognition

by

Robert Howard Kassel

S.M., Electrical Engineering and Computer Science
Massachusetts Institute of Technology, 1990

S.B., Computer Science and Engineering
Massachusetts Institute of Technology, 1986

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

at the
Massachusetts Institute of Technology
June, 1995

Signature of Author .....
Department of Electrical Engineering and Computer Science
May 12, 1995

Certified by .....
Dr. Victor W. Zue, Senior Research Scientist
Thesis Supervisor

Accepted by .....
Frederic R. Morgenthaler
Chair, Department Committee on Graduate Students

A Comparison of Approaches to On-Line Handwritten Character Recognition

by Robert Howard Kassel

Submitted to the Department of Electrical Engineering and Computer Science
May 12, 1995 in partial fulfillment of the requirements for the degree of
Doctor of Philosophy.

Abstract

Speech and handwriting are manifestations of a common need for linguistic communication. The similar nature of speech and handwriting recognition problems suggests that a largely shared solution may be possible. Recent advances in speech recognition can be partly attributed to changes in the research paradigm. These changes include using large corpora of common training and testing data, adopting statistical modeling over rule-based approaches, and ensuring meaningful comparisons between candidate technologies. The resulting improvements in system performance and robustness permit the study of increasingly difficult recognition tasks.

The primary goal of my thesis is to compare handwriting representations for on-line, printed, alphanumeric character recognition without striving to construct the highest-performance system. My studies are based on a carefully collected body of data containing some 87,000 characters from 150 writers. Material was selected automatically to ensure compact coverage of significant letter sequences. Subjects were instructed and prompted so as to minimally influence the writing they produced. A time-aligned transcription was entered for all of this data. I conducted an authentication study to understand better the classification difficulty of this writing. Only 81.7% of testing characters were identified correctly.

I examined a number of potential representations for handwriting classification including bitmaps, projections, transforms, chain codes, and point-sampling, paying particular attention to pen motion as an information source. All experiments were based on Gaussian mixture models because of their flexibility. The best representation features Cartesian coordinates of 10 equally-spaced samples along the pen trajectory. Without the benefit of relative size information, this representation resulted in 77.2% correct character classification on testing data.

Finally, I adapted the SUMMIT segment-based speech recognition system developed at MIT to handwriting. Segmentation is based primarily on pen-lifts, but strokes are divided to account for connected character pairs. The parameter described above is computed for each segment and the resulting graph passed to the recognition engine for classification and search. This system was able to correctly recognize 65.1% of the test-set characters. Incorporating a bigram character grammar with perplexity 11.3 improved this performance to 76.4%.

Thesis Supervisor: Dr. Victor W. Zue
Title: Senior Research Scientist

Extended Abstract

Speech and handwriting are manifestations of a common need for linguistic communication. Both may be viewed as encoding linguistic information in a time varying signal to ensure its transmission through a noisy channel. Both require the composition of fundamental units into endless combinations according to structural rules. Both may benefit from modeling contextual, user, and environmental variations. Both may be simplified by inherent as well as artificial constraints. The similar nature of speech and handwriting recognition problems suggests that a largely shared solution may be possible.

Spoken language is natural, pervasive, efficient, and can be used at a distance. Written language does not have any of these properties, but unlike speech it can be covert, incorporate positional and graphical information, and resist corruption by acoustic noise. The natural advantages of these communication modes along with their complementary characteristics suggest that both will be used in future human-machine interfaces. In particular, both will have significant impact on computer systems as they become smaller, more mobile, and more consumer-oriented.

Recent advances in speech recognition can be partly attributed to changes in the research paradigm. These changes include using large corpora of common training and testing data, adopting statistical modeling over rule-based approaches, and ensuring meaningful comparisons between candidate technologies. The resulting improvements in system performance and robustness permit the study of increasingly difficult recognition tasks. Applying these paradigms to handwriting could yield similar gains.

In this thesis I address the problem of on-line printed character recognition for an alphanumeric symbol set. Although it is generally assumed that the off-line recognition problem subsumes on-line handwriting recognition, I wanted to investigate if the temporal information available to on-line systems could be exploited to improve performance. Furthermore, developing recognition technology for interactive systems will require the study and modeling of writing phenomena that are not present in off-line data. The alphanumeric symbol set selected is reasonably small, yet covers a wide-range of task domains and provides an opportunity to observe highly confusable glyph pairs.

The primary goal of my thesis is to compare handwriting representations for on-line, printed, alphanumeric, character recognition without striving to construct the highest-performance system. The key to such a project is a carefully crafted body of data. The specification for data was primarily based on significant letter sequences,

discovered automatically to limit the number of glyph contexts required. Words were selected using another automatic procedure to ensure a compact coverage of these sequences. Subjects were instructed and prompted so as to minimally influence the writing they produced. A total of approximately 87,000 characters were collected from 150 writers. Boxed characters were also collected from each subject for comparison purposes.

A hand-checked, aligned transcription was produced for the writing data in this corpus. The transcription conventions included symbols for connected characters, minor noise such as spurious pen contacts, and major noise such as corrections and doodles. At this point there is much that can be learned from the data. In particular, I examined gross characteristics on a per-subject basis to assess subject's compliance with data collection instructions. For example, although subjects were instructed to print their responses, 8.2% of the words contained at least one pair of connected characters.

I divided the data by subject into training, development, and evaluation sets. In order to understand better the classification difficulty of our task, I conducted an authentication study using the development and evaluation data. The characters in these subsets were shuffled and presented in isolation to one of three authenticators. The authenticators were instructed to record labels for each token in preferential order. Approximately 81.7% of the development data was identified correctly, with roughly 6.1% of the responses an error in letter case only. Interestingly, subjects were better able to identify characters written in strings compared to those written in boxes.

Next, I examined a number of potential representations for handwriting classification including bitmaps, projections, transforms, chain codes, and point sampling, paying particular attention to the value of pen motion as an information source. Handwriting representations can be divided into two classes: static representations which are based on a pixmap and dynamic representations which are based directly on the ink signal. It is difficult to ensure a fair comparison between these classes. Accordingly, I investigated hybrid representations in which dynamic information is quantized and represented within a pixmap. The representations explored include bitmaps, projections, transforms, chain codes, and point-sampling. However, I eschewed rule-based feature extraction since such approaches have proved limiting in speech recognition. All experiments were based on Gaussian mixture models because of their flexibility. The best of the many representations and variations investigated was based on the Cartesian coordinates of 10 equally spaced points along the pen trajectory and correctly classified 77.2% of development-set characters. This result is achieved without the use of preprocessing techniques and without the benefit of relative size information.

The final area this thesis examined is the use of a segment-based recognizer for handwriting. To do this I adapted the SUMMIT speech recognition system developed at MIT. The handwriting segmentation was based primarily on stroke boundaries. Rather than treating boundaries equally, I found that a simple classifier could cor-

rectly identify inter-glyph boundaries 92.5% of the time. Strokes were also split to account for connected characters. Parameters were computed for each segment, and the resulting graph was passed to the recognition engine for classification and search. Using the representation described above, the system was able to correctly recognize 65.1% of the development-set characters. Constraining the result using a bigram character grammar with perplexity 11.3 improved this performance to 76.4%.

While this thesis is primarily about handwriting recognition, I believe that this investigation will prove informative for speech recognition as well. By forcing systems to perform tasks beyond their intended domain, by researching a different but related realm, and by examining the process by which research is performed, one is required to look at old issues from a new perspective.

Acknowledgments

My deepest thanks go to my family, especially my Mom, for many years of love and support while I have been in school.

I am deeply grateful to my thesis supervisor Victor Zue. He has funded, encouraged, educated, and guided me since soon after I arrived at M.I.T. as a freshman. As with any long-lasting, close relationship there have been good times and bad, but the difficulties have only served to reinforce our bond. While I'm sorry my time with you as a student is at an end, I look forward to continuing our friendship in the years to come.

I would like to thank the other members of my thesis committee, Larry Frishkopf, Jim Glass, and Tomaso Poggio, for their insights and suggestions. Our good spirited meetings helped to make the thesis process more enjoyable. I also thank Vicky Palay, Joe Polifroni, Stephanie Seneff, Nancy Kelly, and Patrick Kelly for their comments on drafts of this document.

My thanks also go to the many subjects who provided handwriting for my experiments, and to Todd Boutin, Jay Grabeklis, and Tarik Saleh for their participation in the authentication study.

The spoken language systems group, and before that the speech communications group, has always been an excellent place to work because of the comradeship of its members, the sharing of skills and expertise, and the emphasis on fine research facilities. While my thanks go out to the group as a whole, I am particularly grateful for the tremendous yet often unrecognized contributions made by Vicky Paly, Sally Lee, Joe Polifroni, and Christine Pao. In addition, Jim Glass and Mike Phillips have provided much technical assistance in performing my thesis experiments, and Joe Polifroni assisted with transcribing.

A long, long time ago I was told that the greatest influence on my work would be the people I shared an office with. I was lucky to have had Nancy Daly (now Nancy Kelly) as an officemate for much of my time at M.I.T. She has been a good friend,

particularly through the tough times, and along with her husband Patrick Kelly is a good neighbor too!

Dave Whitney, Hal Herhold, Melissa Lea and Lori Lamel have all been particularly kind and encouraging over the years, as have the Kennedy's, the Hattons, and many epochs of F-entry Vigilantes.

Finally, I would like to thank the Microsoft Pen Computing Group, particularly Greg Slyngstad and Sung Rhee, for their time, donations, and words of wisdom.

This research was supported by ARPA under contract N 66001-94-C-6040, monitored through the Naval Command, Control, and Ocean Surveillance Center, and by a grant from Apple Computer, Incorporated.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

As computers become smaller, more powerful, and less expensive, human factors play an increasingly important role in system development. The most visible aspects of the computer are sure to undergo a radical change. For while machines are growing ever smaller our fingers are not. The keyboard is already a limiting factor in computer packaging. Does this mean we cannot progress any further?

The answer will come from the technologies of speech and handwriting recognition. Recording speech or handwriting requires additional hardware, but the volume needed for these transducers is small relative to the size of practical keyboards. At the same time, speech and handwriting recognition will provide powerful new ways of using computers. Both depend on forms of communication that are used by people daily and are accessible to the majority of the population. Applications aimed at the general population must be effective for the general population rather than only a few technically sophisticated individuals. The same technology that will allow us to make smaller, consumer-oriented computers will make it easier for customers to use the products.

Speech and handwriting are manifestations of a common linguistic process. One is based on a set of sounds, the other on a set of characters, but in both cases these fundamental units are combined to form meaningful expressions. Since transmission of the intended message is important, both have evolved guards against corruption from noise. Yet both exhibit a high degree of variability stemming from a wide-range of sources.

Despite their similarities, speech and handwriting are complementary in nature.

Unlike handwriting, speech can be used to control a system without making contact with it. However, handwriting is more effective at providing commands covertly. Speech provides a natural and efficient means of supplying text, but handwriting can embellish text with spatial and graphical information. Speech provides an ideal means of communication when our hands are busy. Handwriting allows us to express ourselves while we are listening.

The nature of a task may strongly favor one mode of interaction over the other, but in general users will want to make a choice based on their immediate needs. Moreover, users may take advantage of the synergy between speech and handwriting, performing tasks more efficiently than they could with either mode alone.

If such systems are to become reality we must achieve a greater understanding of the component technologies: We must learn how to extract the salient signal characteristics. We must identify the sources and nature of the variability encountered. We must understand the trade-offs involved in system development. And we must characterize the errors and their effect on the user.

Within the past decade, advances in automatic speech recognition have reduced word error rates to levels acceptable for practical technology deployment in a range of applications. These improvements partly can be attributed to the research paradigm adopted by speech scientists. A large corpus of data, collected from many speakers, is divided into training and testing portions to prevent biasing the results in favor of learned subject characteristics. The same data is used across all experiments so that differing performance can be attributed to the procedures alone. While speech knowledge is applied in the design of systems, statistical modeling ensures that internal parameters accurately reflect the data. This data-driven approach has resulted in improved system performance and robustness despite increasingly difficult tasks.

In this thesis I hope to apply these techniques to handwriting recognition. My goal is not to construct the best possible system for a real application. Instead, I strive for a meaningful comparison of representations within a reasonably useful domain. In particular, I examine whether the temporal information present in online handwriting data is a source of information or noise. By bridging the realms of speech and handwriting recognition, I hope to stimulate ideas that will benefit both fields.

1.1 Handwriting Recognition Taxonomy

Automatic handwriting recognition is an extremely general term. The only key requirements are that the data be produced by a person with a stylus and that there is some higher level meaning to be extracted by a computer. In this section I describe some of the important distinctions to be made in specializing the handwriting recognition problem.

1.1.1 Data Dimensions

The most important distinction we can make is the manner in which handwriting is captured for recognition. In *off-line* recognition, the writing is treated as an image captured by a document scanner, video camera, or the like. This contrasts with *on-line* recognition, in which the trajectory of a stylus is recorded while the user writes. Since a static image is sufficient for reconstruction of the writer's intent, it is reasonable to question whether on-line data parameterization provides additional information that can aid recognition or adds noise that hinders recognition. While access to pen movement can elucidate overlapping strokes to reduce confusability, a particular image can be produced with pen trajectories of varying direction, speed, and acceleration, reducing data consistency.

Another important distinction is the *domain* of the written signal to be recognized. Most commonly the concern is written text, and the task is to extract the underlying linguistic intent. A similar area which has been reported on is the recognition of shorthand notation [46]. Specialized forms of symbolic communication can be the subject of handwriting recognition; there has been some interest in understanding musical scores [69] and mathematical notation [10], both of which differ from text in their dependence on 2-dimensional arrangement. The recognition task may be pictorial rather than verbal. For example, systems have been constructed to produce "clean" drawings from sketches drawn by the user [1].

While other areas have much potential, the focus of the remainder of our discussion will be recognition of texts. Given this we must still narrow our distinction to a particular *language*, class of languages, or writing system. Texts written in Japanese, Arabic, and English have fundamentally different properties. Written language sym-

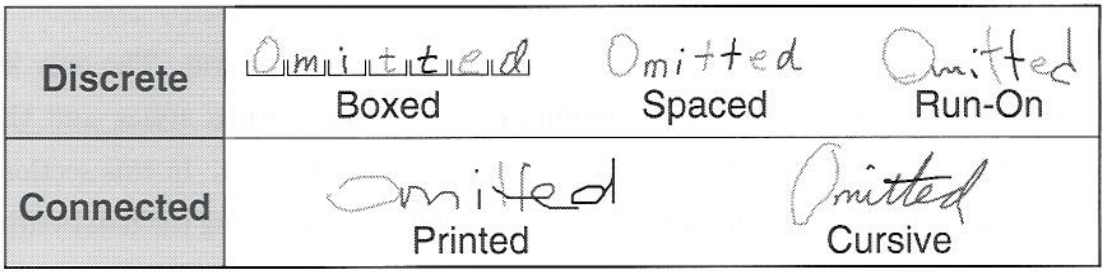


Figure 1.1: Examples culled from the research corpus, illustrating various writing styles. Strokes are demarked by varying intensities.

bols can correspond to words, syllables, or phonemes. The number of distinct symbols can range from tens to thousands. Beyond a language's alphabet we may need to encode symbols for digits, punctuation, and even editing gestures, and we may need to encode variants of characters such as upper- and lower-case.

Let us further limit our discussion to written English. There are a number of ways the same set of characters can be reproduced by a particular writer as shown in Figure 1.1. *Boxed* writing is the most constrained, where subjects are required to print each character in a separate box. This obviates the need to locate and separate individual characters. Somewhat more relaxed is *spaced* writing where writers are required to ensure that characters do not overlap. Further relaxed is *run-on* writing in which we allow overlap. All of these styles require that each character is produced discretely, i.e., the pen is lifted between characters. If we eliminate this constraint we allow *connected* writing. This includes a *printed* style in which few characters are connected and a *cursive* style which uses different character shapes and connects most characters. Two additional distinctions which are fairly common are pure cursive, which requires that all characters be connected, and mixed, which allows both printed and cursive forms. This list is not exhaustive. For example, calligraphic forms are sometimes used for formal documents.

1.1.2 Technology Dimensions

In describing handwriting recognizers we must also distinguish between different recognition technologies.

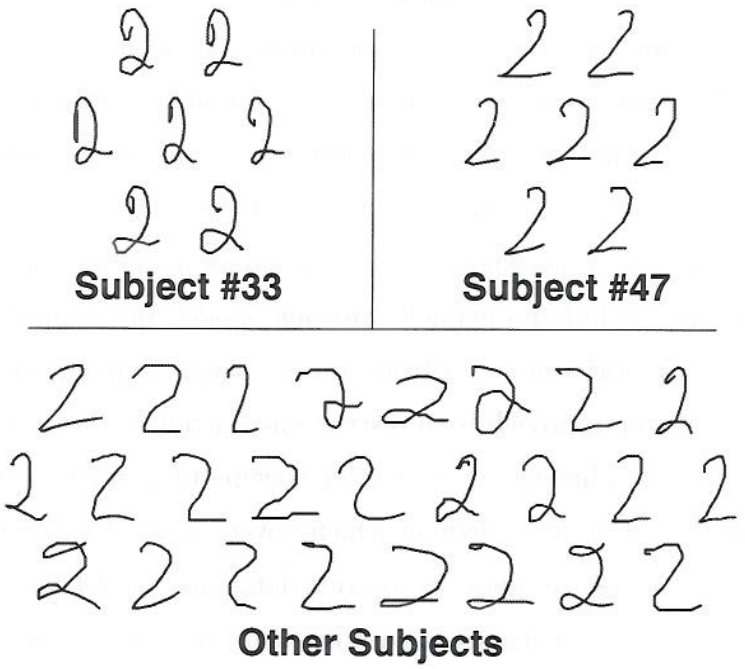


Figure 1.2: Examples of the digit "2," culled from the research corpus, illustrating intra- and inter-writer variability.

The most important distinction is the form of *enrollment* users must make for the system to perform as claimed. The most stringent systems are *writer-dependent*, requiring a potential user to provide sufficient data for training. The amount of such writing required may be substantial, but such systems can offer the best performance by avoiding the need to capture inter-writer variability, demonstrated in Figure 1.2. At the other extreme are *writer-independent* systems which require no enrollment data. Such a system can be tuned to a particular writer during use, in which case it is said to be *writer-adaptive*.

Another important distinction is the inventory of *units* modeled by the system. We must first ask what aspect of the data is modeled. While it is natural to think of classifying characters, there can be data sharing advantages to using a smaller unit such as strokes. On the other hand, using larger units such as words can make the distinctions between classes greater. Using larger models implicitly incorporates contextual variation in the data. A system with *context-dependent* models tries to capture this variation directly. Details of the implementation, for example the clas-

sification technique used and the parameters which control it, will also affect the results observed. Some recognition strategies implicitly segment the input as a part of recognition. In other systems the segmentation and classification are performed independently. Algorithms for explicit segmentation can produce results with a wide-range of branching factors, and this can impact the results greatly.

Finally, we must ask what higher-level constraints are imposed upon the recognizer. A system may include no explicit *language model*, but typical modeling techniques capture the a priori statistics of the training data. Explicit language modeling is also used. A common approach is to restrict the output of the recognizer to words appearing in a lexicon. This can work well for specific tasks with small vocabularies, but it is difficult to construct a lexicon which covers general English text. A more flexible approach is to use a character n-gram language model [33], estimating the probability of n-character strings from data in a suitable text corpus. As we increase n we expect this model to provide greater constraint, but the amount of data needed to train the model increases dramatically. Tree-based grammars are also possible to provide structural constraints.

1.1.3 Summary

There are many factors which can influence the performance of handwriting recognition systems. These include differences in the way handwriting is captured, how the task is constrained, and what technology is used. Thus it is impossible to reduce our evaluation of a system's performance to a single number. Rather, we must understand what aspects of a handwriting system work well in the context of a particular application and ask ourselves how well such techniques can be applied to new domains.

1.2 Previous Work

There is an extensive body of literature in the field of handwriting recognition. In order to limit the scope of our discussion, I will concentrate on systems for English alphanumeric texts. It is important to note that substantial research in on-line handwriting recognition has taken place in industry. This work is often considered

proprietary and remains unpublished. Wherever possible I will quote performance in terms of character accuracy. To avoid redundancy, descriptions of some papers will be brief.

1.2.1 Survey Papers

An early survey of work in character recognition was written by Harmon [30] in 1972, but at that time on-line recognition was extremely rare. By 1980, Suen et al. [76] were able to report on over two dozen such systems.

Tappert et al. [79, 80, 82] have written comprehensive reviews of on-line handwriting recognition. They begin with the hardware requirements, basic handwriting properties, and fundamental recognition problems. Next, they describe preprocessing techniques for segmentation, noise reduction, and normalization. A number of classification techniques are presented including decision trees based on features and dynamic programming [64] to match shapes, but statistical methods are mentioned only briefly. In discussing system performance, they note "it is difficult, if not impossible, to compare" experimental results due to the differences in and vague documentation of testing protocols. They cite human reading performance of 96.0–98.8%[1] for isolated block characters but cover only two systems using an *alphanumeric* character set. In addition, they describe a number of commercial technologies and discuss future challenges for researchers.

Nouboud and Plamondon [58] survey on-line handprinted character recognition. They describe dynamic programming as a common technology – it is found in 40% of the systems reviewed – with syntactic methods as an alternative. When considering the evaluation of recognition systems, they note that data collection procedures are "quite unnatural" and that it is very difficult to compare the results from different tests due to a lack of standardized benchmarks. While most character recognition results reported are above 92% correct, they observe that the tests are often performed on small data sets from few subjects, making the results less applicable to the general population. Even for a system which they feel had been tested adequately, they found

[1]Speech recognition results today are usually given in terms of error rate, but I have adopted accuracy since it is more common in the handwriting field. Either can be derived from the other. However, reporting error rate helps make apparent performance gains when mistakes are rare.

the claimed performance of 93.4% to be above their observed performance of 89.8%.

While it is not a survey paper per se, Wang and Gupta [85] describe several approaches to handwriting recognition. They begin by examining fundamental properties of handwriting and give examples of the ambiguity encountered. They then describe several syntactic and structural means of representing characters. No implementations of these schemes are presented; only a qualitative comparison is made.

Govindan and Shivaprasad [26] take a broader view and survey many forms of character recognition. This paper is notable for its references to other survey papers and its coverage of non-English character recognition. Wakahara et al. [84] examine on-line recognition primarily for Japanese writing.

1.2.2 On-Line Handwriting Recognition

Early recognition systems were severely limited by the available computational resources. Often these systems were deterministic in nature, incapable of providing alternates should the most likely result be in error.

For example, Kurtzberg and Tappert [42] describe an approach for segmenting discrete characters which may overlap. Their technique is based on comparing the distance between pen strokes with thresholds to build segments. This yields a single segmentation of the handwriting data. If this segmentation is wrong, it may be difficult or impossible to correct the error at a later time. This problem is compounded by the fact that the proper segmentation itself may be ambiguous without higher-level knowledge.

One way to counter this difficulty is to enforce some form of separation between characters. This separation need not be spatial. Nouboud and Plamondon [59] describe a system which requires a brief pause between characters. After smoothing the recorded pen trajectory, they construct a chain code [2] which includes quantized direction and position information. These codes are compared using a string similarity measure. In writer-dependent recognition of a 59-symbol task their system correctly recognized 96.0% of the characters.

Ward and Blesser [88] describe some of the basic issues they encountered in the deployment of an early commercial system for a discrete 95-character task. In a later

paper [87], Ward and Kuklinski give details of this system's construction. Characters are represented as a chain code of pen trajectory extrema. Base-forms are initialized to cover potential allographic[2] variability, notably due to differences between North American and European styles. Additional base-forms are created to account for differences in stroke order, shape, and direction. This results in an extremely large number of variants, over 15,000 for an allograph of upper-case "A" alone. Input data is compared against these models using syntactic pattern matching. Although the evaluation of systems is discussed at length, no performance figures are given.

In order to create a handwriting recognition system, we must first understand what distinguishes one character from another. Once we have obtained this knowledge it is tempting to encode it in the form of rules to be used for character recognition.

Kerrick and Bovik [37] describe a system for a 69-character alphabet written discretely. Because efficiency is paramount for their application, they use a binary decision tree to eliminate unnecessary parameter computation. The characters are represented using local structural primitives such as "tees" along with more global features such as stroke shape and aspect ratio. Thresholds are introduced to allow for imperfect placement of the stylus in connecting strokes. The decision tree rules test only for the presence of required properties. Candidate characters are then verified against models. The authors do not describe any formal evaluation of their system, but they do say it "has been demonstrated to be highly effective and efficient."

Rather than segment the writing stream into characters, we might try to segment and model an alternative set of basic units. Such an approach is described by Fujisaki et al. [20, 21] in which strokes are classified according to their order within a particular character. Thus, a vertical stroke might be labeled "1/4 of E" to indicate that it is the first of four strokes within an "E." Permissible paths are traversed and potential characters are verified using template matching to discriminate between cases which are identical at the stroke level. A character grammar provides further constraints. This system achieved a character accuracy of 87.6% for an 82-symbol, writer-dependent task. However, subjects for this study were coached to write similarly shaped characters so that they were distinguishable.

[2]Allographs are symbols which differ in graphical form but are not linguistically distinctive. An analogous concept defines allophones in spoken language.

A comparison of stroke- and character-based techniques was conducted by Schomaker [67] for writer-dependent, lower-case cursive handwriting recognition. In both cases a Kohonen feature map [39] was used for classification based on Cartesian coordinates. His results suggest the stroke-based approach is superior. However, it is not clear that the full potential of either method was realized because the search space of results was pruned,[3] potentially discarding correct responses.

Unfortunately, our knowledge of a recognition task is often incomplete. Even when we have great insight into the problem, it can be difficult to codify this knowledge so that it is useful within a system. Statistical pattern recognition techniques hope to circumvent these issues by modeling a set of training data and delaying decisions as late as possible in the recognition process. This allows the most amount of information to be used in reaching the decision. We depend on poor classification scores for non-character segments to eliminate incorrect paths.

Two approaches to stochastic segmentation are compared by Schenkel et al. [66] for handprinted words composed of capital letters. In their first experiment, strokes are combined into hypothesized segments based on simple heuristics. Although this approach requires pen-lifts between characters, it was noted that "very few people did not separate their characters" in this manner. Their second experiment avoids this constraint by advancing a fixed-size window along the input to produce the segmentation. In both cases a time-delay neural network (TDNN) was used to classify the segments, and the resulting graph was searched for the best response using the Viterbi algorithm [18]. The two approaches resulted in character recognition rates of 92% and 89% respectively. The performance of the better system could be increased to 95% by constraining the result to an 80,000-word lexicon. It is worth noting that the authors report the error rate was reduced "by more than a factor of two" by training the classifier to identify non-character segments as such. In a similar system for boxed data, Guyon [29] reported classification rates of 96.2% for upper-case letters.

Hidden Markov models [61] (HMM) are a popular stochastic modeling technique. For example, Bellegarda et al. [4, 56] describe an HMM-based system for an 81-symbol discrete character task. In their system, the input stream is resampled at equally

[3]Pruning may be needed to limit the computation and memory requirements of a deployed system, but for research purposes it is an additional source of error to be characterized.

spaced points and frames are constructed to include slope and curvature information from multiple points. These frames are fed through Gaussian-mixture models and the resulting feature vectors are used to train a one-state HMM for each character. This system results in character recognition rates of 86.6% in a writer-dependent mode and 80.9% in a writer-independent mode. Both of these results compare favorably to a template-based approach used in the IBM ThinkPad product.

Stochastic segmentation and modeling have been applied to cursive writing recognition as well. Nag et al. [55] described a small vocabulary HMM system in 1986. A more recent application of these techniques is described by Makhoul et al. [48, 73] for an 86-character task. A feature vector was constructed for each sample in the input to represent movement of the pen. These were fed into a vector quantizer and the results used to train the HMM's with linear topology. The system was trained and tested on sentences from six subjects. On average, this writer-dependent system correctly recognized 95.9% of the words, with an estimated 98.6% correct character accuracy. However, to achieve this level of performance the recognizer was constrained using a bigram language model covering approximately 25,000 words.

Manke and Bodenhausen [49] apply a TDNN to writer-dependent cursive lower-case letters. The system is trained on feature vectors which attempt to capture both static and dynamic information. Dynamic time warping is applied to the outputs of the network to match word models. On a 20,000 word task, this system achieved a word recognition rate of 83.0%. When applied to a writer-independent, isolated character task, the system correctly recognized 91.5% of lower-case letters.

Schenkel et al. [65] combine a TDNN for feature vector classification with an HMM for the search to recognize writer-independent lower-case words. The TDNN input window approximates the width of a single character and it has an output for each symbol of the alphabet. These outputs are used as the observation vector of an HMM. A fast-match procedure reduces the computational requirements of the system. When tested on words written by 25 subjects, the authors report a 71.8% character accuracy. This result improves to 89.1% when a 25,000-word lexicon is used to limit the results.

In order for statistical modeling to be effective, sufficient training data must be available so that pertinent variations can be observed. Kuklinski [41] and Wing [90]

describe some of the factors introducing this variability. One way to reduce the amount of training data needed is to normalize the data. Guerfali and Plamondon [27] outline a number of ways this can be done for on-line handwriting. They discuss techniques for noise reduction (to smooth the writing), baseline correction (to orient the writing horizontally), deskewing (to normalize the characters' slant), and zone detection (to determine character height). These techniques were not evaluated in the context of a recognition system, but they were judged by a panel of subjects to be generally effective.

As we have seen, recognition systems can be constructed from many potential representations, models, and algorithms. Although high accuracy is an often sought goal, the highest accuracy may make unrealistic demands on memory, computation, and application constraints. A research system must be carefully pared to reduce these requirements while maintaining a high performance level.

Tappert [78] examines some of the trade-offs possible for an on-line boxed character recognizer. He first notes that by adopting an improved representation and distance metric the character error rate of the system was halved, raising the performance to 97.3% on writer-dependent data. Part of the testing material was used to develop these improvements. A "significant" part of the improvement was due to the elimination of a parameter which could be construed as noise. Tappert then reduces the computational requirements of the system by simplifying the preprocessing, using more restricted models, and pruning the search space. These techniques increased the speed of the system by an order of magnitude with negligible effect on accuracy.

We must remember that the requirements for any recognition system are ultimately driven by the application and its users. For example, while real-time recognition performance is required in some instances, there may be situations where selecting a slower yet higher-accuracy system is more prudent. Similarly, an extremely fast but low-accuracy system may be perfectly acceptable if the domain can be constrained sufficiently. Thus there is not necessarily a single, best approach to handwriting recognition.

1.2.3 Off-Line Handwriting Recognition

Many of the issues surrounding off-line handwriting recognition are applicable to the on-line problem. It is simple to treat dynamic pen data as static by scan conversion. The resulting data should be easier to recognize than scanned images due to the lack of visual noise and pen-width variability. Conversely, it is possible to treat image data as if it were collected on-line by inferring the dynamic information. Suen et al. [75] present a good overview of recent advances in off-line handwriting recognition.

Before video display terminals became commonplace, a popular application for recognizing handprint was to replace keypunching for FORTRAN programs. More recently a driving task has been locating and recognizing routing information from handwritten addresses. Srihari [71] illustrates many of the issues within this domain. Because of their keen interest in this problem, the U.S. Postal Service has funded large common corpora of digits for training and evaluating postal address processing systems. The U.S. Census Bureau is also extremely interested in off-line handwriting recognition and has collected a corpus of alphanumeric data which has been distributed widely. An overview of off-line corpus development issues is presented by Hull and Fenrich [31], including descriptions of some existing character image resources.

Srihari [72] gives a summary of performance for digit recognizers for the zip code task. The high degree of accuracy required necessitates the incorporation of rejection criteria. However, varying rejection sensitivity makes it more difficult to compare systems. For example, Nadal et al. [54] describe a recognizer for zip codes which independently classifies the character's skeleton and contour. The outputs of these algorithms are combined using a decision rule to produce a character accuracy of 84.9%. The remaining 15.1% of the data was rejected. Suen et al. [77] update this system to use two additional independent classifiers. The new system correctly recognizes 93.1% of the characters while rejecting the remaining 6.9%.

Confidence in automatic recognition systems can be greatly enhanced if redundant information can be applied to verify results. For example, a check digit can reduce the probability of misrecognizing a number string. Zip codes do not incorporate such

a feature, but redundant information is available from other parts of the address, particularly the city and state. Chen et al. [6] created an HMM-based system to recognize city names scanned from real envelopes. When constrained by a 271-word lexicon, the system achieved a 72.3% word accuracy. Identifying the components of an address is itself a difficult task. Cohen et al. [8] describe how the underlying structure can be extracted. An important component of this is locating the breaks between words or other syntactic elements.

There are other popular applications for off-line handwriting recognition. For example, Di Zenzo et al. [9] examine the problem of recognizing characters at any orientation and size as extracted from maps. Gupta et al. [28] consider recognizing currency values scanned from bank checks.

Another noteworthy paper, written by Smith et al. [70], studies digit classifier performance as a function of the training data used. They also compare k-nearest neighbor classifiers [13] using several different distance metrics. The authors report that an order of magnitude increase in training data results in a decrease in error "by half or more." Just over 60% character accuracy was possible using only a single, randomly selected prototype per digit.

1.2.4 Related Fields

There are a number of other fields which can provide information useful to handwriting recognition. In this section I list a few papers of note.

Optical character recognition (OCR) is similar to off-line handwriting recognition although its concern is machine printed texts. Pavlidis [60] gives a brief but practical overview of the field. In particular, he notes five sources of errors: shape similarity, print quality, digitization distortion, feature detection, and classifier design. Mori et al. [53] trace the development of OCR and off-line handwriting recognition systems including descriptions of common processing techniques. Impedovo et al. [32] concentrate on the capabilities of commercial systems.

Speech recognition is perhaps the field most similar to on-line handwriting recognition. Rabiner and Juang [62] provide an introduction to the modern practice of speech recognition.

We can also learn from studies of human production and recognition of handwriting. Suen [74] surveys these fields and describes a number of experiments. He notes "solid evidence" demonstrating that printing is more legible than cursive writing. Although cursive writing is shown to be somewhat faster than printing, at least one study suggests that the two forms of writing are equally efficient with sufficient practice. In a study of human classification covering 26 printed letters, subjects were able to correctly identify 97.6% of the characters. Schoonard et al. [68] analyze data collected from individual subjects' handwriting and survey their attitudes toward a recognition system. Then they compare the performance of their system to that of humans reading the same data.

Human factors are an important aspect of application design. As such they affect the requirements made of a recognition system. Gould and Alfaro [25] compare a traditional text editor with handwriting and speech recognition for the purposes of document revising and found handwriting to provide the preferred interface. Wolf [91] examines the user's explanation of recognition errors.

It is important to understand how the transducer affects the handwriting recorded. Ward and Phillips [89] provide a comprehensive review of digitizer technology and performance with an eye toward how this affects handwritten text. In a sidepiece to the article, they note that difficulties in using existing digitizer technology motivated them to produce their own hardware. Meeks and Kuklinski [51] compare the dynamic characteristics of digitizers.

Finally, in 1977 Kay and Goldberg [36] outlined the capabilities of a prototype computer system, the Dynabook. While this system did not incorporate handwriting recognition, many of the ideas expressed in this paper are worth revisiting.

1.2.5 Summary

Handwriting recognition has a history that stretches nearly to the start of electronic computing. Recently, one can observe newfound interest in the field, instigated by new applications on faster computers with practical transducers and encouraged by speech recognition successes.

Despite a rich array of approaches, research progress is hindered because it is

difficult to evaluate on-line technologies against one another given the information in the literature. Systems are trained and tested on corpora of varying difficulty. In some cases, the same data serves as both the training and testing material, yielding results which may not generalize to additional writing. Often evaluations take place on only a small amount of handwriting from a few subjects who are not representative of the general population. Writers may be instructed to make their writing more consistent. Potentially troublesome data may be discarded due to overly strict subject compliance criteria, yielding falsely optimistic results. Data processing may include steps of questionable value, yet their effects on the results is rarely isolated. Even result reporting can be suspect, obscuring the facts by excluding automatically rejected material or including answers other than the best.

If we are to integrate handwriting and speech recognition technologies, we need to begin addressing these problems. We must have an understanding of what works and, more importantly, the nature of the errors with which we must contend. Systems must be compared using common training and testing sets. The data should include a large amount of handwriting from as many subjects as possible, collected under the most natural conditions as is practical. Little if any data should be excluded from study, and only then because it strays beyond a well-defined scope. Individual algorithmic differences should be isolated to properly attribute performance gains determined in a straightforward manner by identical metrics. These are all techniques now common in the speech recognition field.

1.3 Thesis Scope

The primary goal of my thesis is to demonstrate how speech recognition research techniques can be applied to handwriting recognition, seeking a fair comparison of algorithms to advance our understanding of relative efficacy. In particular, I hope to determine better the value of dynamic information. In all cases I stress simplicity and reproducibility so that my results can be used as a starting point for further studies.

It is impossible to examine all aspects of handwriting recognition within the bounds of this work. I have limited my thesis to on-line handwriting recognition because my ultimate interest is in systems that interact through both speech and

handwriting. I selected a 62-symbol vocabulary consisting of upper-case letters, lower-case letters, and digits. This domain is capable of supporting a wide-range of applications and includes several highly confusable pairs, but avoids punctuation and symbols whose intrinsic properties may be quite different from alphanumerics. Although the structuring of handwriting into words, lines, and larger blocks is an important component of many applications, such work is beyond the scope of this study.

I have chosen to examine handprinted characters because they are often requested when clarity is required. Recognizers will be most acceptable to the broadest population when they impose the fewest constraints on writing style. Accordingly, I have not restricted character size, orientation, shape, overlap, or connection, provided the basic requirement of natural printing is met. Because no suitable corpora were available, I have collected and transcribed data from a relatively large number of writers.

The capabilities of the classifier used within a recognition system can greatly influence the selection of features to be extracted. I have favored comparing representations over comparing classifiers, but this requires ensuring that the classification procedure is sufficiently flexible. All of my studies are based on Gaussian mixture modeling [50], a technique proven to work well for speech recognition. I examined static representations, based on an image of each character, and dynamic representations, based on the pen's trajectory. Experiments I performed on human classification of the data provide a baseline for my evaluation.

The complete recognition system is built around the classification and search components of the SUMMIT [93] speech recognizer developed at MIT. Unlike most speech recognizers, this system is segment based. The segmentation may provide multiple paths to be selected from in the search phase. I take advantage of this fact by hypothesizing many potential segmentation points. The explicit segmentation allows me to examine a wider range of representations.

In the remainder of my thesis I describe and discuss the studies I have performed. In Chapter 2 I describe the design, collection, and transcription of a relatively large handwriting corpus. I have approached each of these steps with a degree of care common in speech studies, but more rare in the handwriting field. In Chapter 3 I discuss a selection of handwriting representations and compare their usefulness for

character classification. To better understand the difficulty of this task, I review an authentication study that establishes human character classification accuracy on the identical data. In Chapter 4 I develop an automatic segmentation algorithm and incorporate it in a handwriting recognition system. I also touch upon constraining the results with a character bigram grammar. Finally, in Chapter 5 I summarize what I have learned and consider some possible extensions to this work.

Chapter 2

Data Collection and Preparation

A critical aspect of any classification or recognition study is the data examined. In this chapter I describe the handwriting corpus used for my experiments. I discuss how I designed the corpus to ensure its efficiency and collected the data to minimize unwanted influences. I present the conventions used in transcribing the data and enumerate basic properties of these transcriptions. Finally, I list the criteria applied to identify and thereby eliminate data unsuitable for my studies.

2.1 Data Collection Issues

The data corpus serves two primary purposes with respect to classification and recognition experiments. First, it must provide sufficient examples of each class so that regularities and variability may be characterized. Second, it must offer ample opportunity for evaluating a system in a meaningful manner. It is difficult, if not impossible, to ensure that these goals are met in general. Our knowledge of handwriting production is at best incomplete, making it troublesome to predict the conditions required to evoke particular variants. As an alternative, we should work towards understanding the sources and nature of variability that we observe. By understanding the influences on our data we can qualify the relevance of our results.

As a practical matter, the overriding factor affecting the data corpus will be its cost. No project has unlimited resources, and thesis projects are particularly constrained. The collection and preparation of data can have a relatively high cost, limiting the size of the corpus. This, in turn, limits the variability that can be

observed. Thus we must be selective in the corpus design to manage the variability covered.

2.1.1 Task-Related Variability

One can imagine a generalized handwriting recognition system which works well for the entire population using a variety of hardware platforms to accomplish many different tasks. Such a system is difficult to construct, in part because the data variability is so great. In addition, more specific systems should be able to provide higher recognition accuracies by taking advantage of the inherent constraints of a particular task.

The most natural way to ensure that data collection closely matches the target task is to record handwriting within a prototype application. This guarantees that major handwriting influences reproduce the deployment conditions without requiring an explicit accounting. However, it may not be practical to develop a fully functional prototype in time for recording subjects. In lieu of this a simulated application, perhaps using a human to perform recognition, can be used. Such forms of data collection are sure to capture spontaneous writing events which would not otherwise be observed. While this is important in the development of real-world systems, it is a source of variability which may be controlled through scripting the responses of subjects. The nature of these responses will affect variability caused by coarticulation, the influence of a character on its neighbors. For the lowest level of coarticulation, one might ask the subjects to write individual letters in alphabetical order. Changing the order requested from each subject would result in greater variability. Recording strings of characters should provide greater variability still.

The writing style permitted is another major source of variability. In the least restrictive cases one would simply instruct subjects to write their responses. As a result, a range of writing styles might be observed. Alternatively, one can request that the subject use a particular style such as fully connected script or handprint. Within a particular writing style many symbol styles or allographs are possible. European forms may differ from North American, as shown in Figure 2.1 for the letter "z." Small capitals may be favored over true lower-case shapes. Some characters, such as

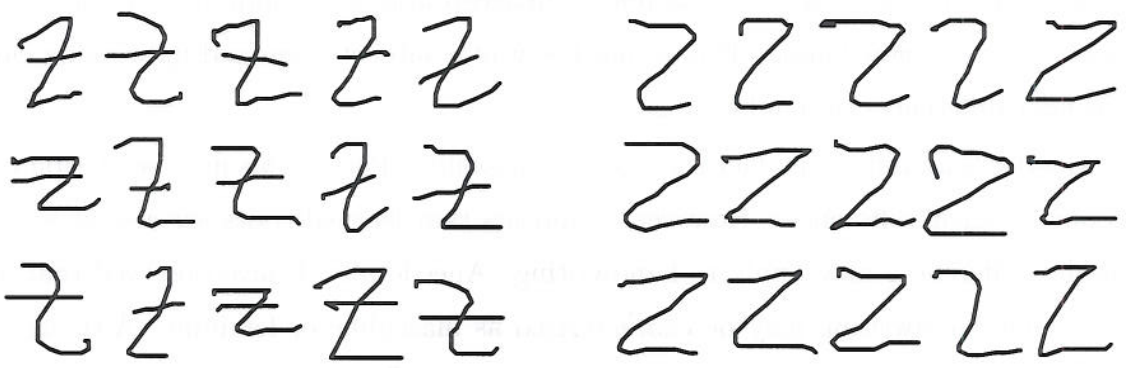


Figure 2.1: Two allographs of lower-case "z" found in the research corpus.

the asterisk, are less standardized than others. Restricting the symbol set will reduce the variability observed. One can further limit the writing style and character shapes by instructing subjects as to what is sought and correcting their practice.

2.1.2 Subject-Related Variability

Subjects themselves are a major source of handwriting variability. All such factors may be obviated by constructing writer-dependent systems. The danger then is that the within-subject variability may be better modeled for some individuals. Were systems to be developed using data from too few subjects, we could obtain a false impression of system performance. Considering handwriting recognition for consumer products, writer-dependent technology is reasonable for devices which are personal. In fact, the user may be well-motivated to train the system to obtain the highest recognition accuracy. However, it may still be desirable for these systems to work reasonably well directly after purchase and without enrollment.

Writing is a learned skill which is taught using many techniques. Over time different approaches have been favored in different locales. Thus a subject's age and place of schooling have an indirect effect on their writing.

A more natural influence is the hand favored by the subject. Left-handed writers may adopt a style to minimize ink smearing. This can result in differences in pen grip, character shape, and stroke direction. More subtle psychological factors may play a role as well. It is important to remember that at times, in many cultures,

favoring the left hand was undesirable or unacceptable and children were forced to switch to their right hands. This is another way a subject's age and upbringing can engender differences in handwriting.

Gender is an influential factor in speech variability. Part of this may be attributed to physiological differences. However, it appears that learned traits are also at work. Similar differences may apply to handwriting. Anecdotally, I have observed that at least some handwriting may be characterized as masculine or feminine. Although I am unaware of any studies trying to quantify these differences, it is an influence that can be easily controlled for in data collection.

Other subject-related factors influence the handwriting recorded. Some subjects may strongly favor a particular writing style and find it difficult or impossible to write any other way. Subjects in certain professions may be trained to write using particular conventions. As subjects age their motor control may degrade and with it their writing. Even factors that vary in the short term, such as fatigue, will influence the data collected.

The possible combinations of these factors suggest that, as a minimum requirement, data should be collected from a large number of subjects. If the amount of data in a handwriting corpus is fixed, there is a fundamental trade-off between the number of subjects recorded and the amount of handwriting from each subject.

2.1.3 Methodology-Related Variability

The experimental procedures themselves will also influence the data collected. This is not necessarily bad, but one must take care to avoid unwanted influences.

I have already mentioned that the subjects may be instructed to produce writing of a particular style. The instructions can influence the subject in other ways. For example, a subject may write differently depending on whether the instructions indicate that legibility is sought. Rewards can further reinforce a desired behavior.

The area used to collect writing can greatly influence the results. Too small an area can result in cramped and illegible writing. Larger areas may result in a wide variety of character sizes, particularly if some subjects believe they must fill the space provided. This can be controlled using writing guides such as lines within the input

area, but this will also influence the location and slant of character baselines.

For scripted data collection, the manner in which prompts are presented will affect the handwriting process. Writing uses the vision system to provide feedback during production. Visual prompting for data requires that the subject shift their focus from the prompt to the writing area, perhaps repeatedly. The effects of such a "copying" task have not been studied formally and may be subtle. I have observed that even the *font* used for prompting can alter the letterforms produced. This influence, in particular for dollar signs, has been observed at another site [63]. Visual prompting could inadvertently limit the variability of character shape and size.

The nature of the writing surface and stylus, as well as the digitizing technology, will affect handwriting production. Some of these effects have been studied by Tappert et al. [81]. For example, the angle between the stylus and the writing surface can influence the position sensed when using some transducers. The stylus may be unusually bulky, altering the subject's grip. A stylus tethered to the tablet may have a mass distribution quite different from a cordless device. The writing surface and stylus tip, particularly for tablets with integrated displays, may produce a "feeling" different from paper and pen. Even the speed and accuracy of inking feedback will alter the subject's handwriting.

The procedural influences on handwriting data collection are numerous and wide-ranging. Care is required to design and collect a handwriting corpus to ensure maximum utility of the data.

2.1.4 Summary

There are numerous sources of handwriting variability, including those related to the task performed, the subjects recorded, and the experimental methodology. Capturing this variability can require recording a large amount of data, yet we are bound by time and budget. In collecting data we must try to exclude those sources of variability which are irrelevant to our experiments while recording as much handwriting as practical, keeping in mind the intended application.

2.2 Corpus Design Overview

I decided to collect a handwriting corpus because I could not identify readily available data suitable for my studies that could be made available to others for further investigation. My primary goal in collecting data was to provide a reliable basis for comparing representations used in writer-independent handwriting recognition. As a secondary goal I hoped to provide a rich source of handwriting for study on its own. These goals required collecting data from a relatively large number of subjects under as few constraints as practical.

The limited scope of this study necessitated concentrating on a particular writing style. I selected handprinting because this style is requested often when clarity is required. However, I provided minimal instruction to subjects on what handprinting entailed. By permitting connected printing and multiple allographs, I hoped to strike a balance between totally unconstrained writing and unreasonably rigid restrictions.

I selected a 62-character set comprised of upper-case letters, lower-case letters, and digits. This alphabet supports a wide range of tasks and includes highly confusable symbols such as the letters "O" and "o" and the digit "0." I chose to focus on individual character strings to include coarticulation effects while avoiding special handling for spaces and line breaks. For similar reasons I excluded error correction from this corpus.

Within these broad restrictions there are a number of ways to collect data. Under the most relaxed conditions, subjects would freely expound text of their own choosing. This offers the greatest variability in data captured, but requires a large corpus to ensure satisfactory coverage of rare linguistic events. In a more directed protocol, subjects would respond to questions. The likely replies can be controlled by carefully choosing the queries. For example, a question to elicit "pp" in the response might be "What fruit is commonly given to a teacher?" Because the question is open-ended, a variety of replies are possible. In the most restrictive method, subjects reiterate each prompt. This is the technique I have used because it gives the tightest control over the data recorded, a strong benefit when the size of the corpus is extremely limited.

When corpus size is limited, it may also be desirable to ensure that its specification is *compact*. A design stressing this criterion attempts to provide a more dense

population of linguistic phenomena than would be observed typically. This reduces the amount of material to be collected but distorts the statistical distributions of the data. In addition, the corpus design process itself is then more complicated. An alternative is to randomly select material from a larger collection of appropriate texts. This approach is simple and approximates the statistical properties of the sampled texts, but on average a large amount of material will be required to achieve coverage of rare events. Note that a compact design is distinct from a *balanced* design, which provides an identical number of occurrences for each phenomenon.

I chose to select prompts in order to compactly cover a particular set of character sequences. This approach has been applied successfully for speech corpora. For example, the TIMIT corpus [43] was designed to provide speech for studying acoustic phonetics. As such it was deemed important to capture the mutual influences of neighboring phonemes. Half of the prompts presented to each subject, designated as "SX" sentences, were hand-crafted to contain relevant phoneme pairs more frequently than expected by random word selection[1]. Covering longer sequences, even whole words or phrases, may be desirable to capture contextual effects more completely. However, as the sequences grow longer they rapidly increase in number while the contextual effects diminish.

2.3 Selecting Character Sequences

The approach I have taken is different from previous designs in that it selectively covers character sequences of variable length rather than exhaustively covering those of a fixed length. The sequences of interest are selected because of their *significance*, which I define as the ability of a sequence to function as a unit itself within the language. By applying an appropriate metric to a text corpus the most significant sequences can be identified. Focusing on these sequences for my design, I hoped to capture the more relevant multi-character strings without needlessly bloating the corpus.

For this approach to work an appropriate body of text is required. I used a lexicon

[1]In fact, I considered using this part of the TIMIT design for a handwriting corpus but found that the compactness property did not extend to the orthography.

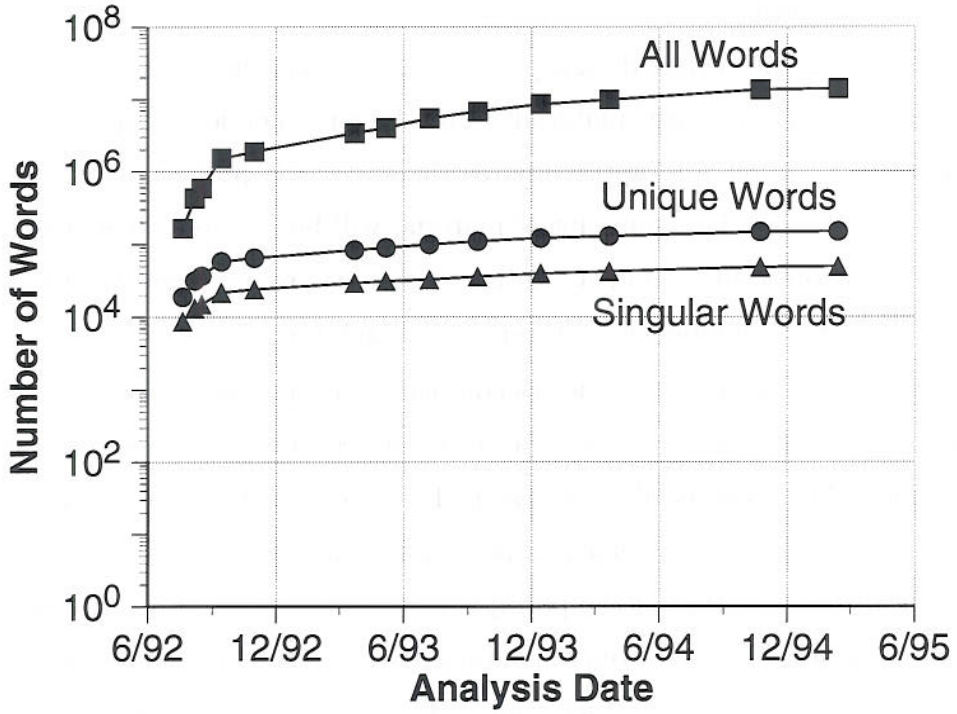


Figure 2.2: Summary of data collected over time from a New York Times newswire service showing the total number of words gathered (All Words), the size of a lexicon containing these words (Unique Words), and the number of words seen only once (Singular Words).

derived from two sources. The first source consists of articles appearing on a New York Times wire service [23], processed automatically to remove control information, editorial alerts, and non-text data. The remaining texts, covering topics from world news to cooking, were divided into words (retaining case and related punctuation). Such data were recorded over several years as shown in Figure 2.2, but this portion of my study was conducted early in the collection effort. At the time, approximately 3.4 million words were available. These were filtered to include only lower-case words occurring at least twice. The second source consists of 360,000 words in a commercially available lexicon [86]. The two sources were intersected to reduce the number of foreign terms, proper names, and typographical errors, producing a lexicon of nearly 33,000 words along with their relative frequencies.

2.3.1 Pair Cohesiveness

The definition of significance given above is too imprecise to be used in defining a metric. Instead I used an exemplar sequence to test potential metrics. For English orthography, the sequence "QU" should receive a high score because "Q" is followed by "U" except in borrowed words. I will call the measured property *cohesiveness* to distinguish it as not necessarily reflecting true significance.

For the moment, consider scoring only sequences of length 2. Frequency is an intuitive measure of cohesiveness, with more common character pairs being more cohesive. Unfortunately, "Q," and so "QU," is relatively rare in English.

Instead a cohesiveness metric must measure the frequency of a sequence relative to that of its constituents. This suggests computing the mutual information [22] between adjacent letters. This approach is similar to that used by Church [7] to identify related words within a text. Such a metric seems to work in that it ranks "QU" second only to "ZZ" as highly cohesive. However, it tends to favor any sequence in which the constituents are rare. This can explain the ranking observed, because "Z" is far more rare than "U."

The bias could be corrected by normalizing mutual information against a sequence's probability of occurrence. I found that this did not work well in some cases due to the combination of log and linear terms. The solution then is to use the negative log of the probability for normalization. I call this "pair cohesiveness," and define it as:

$$C(x_n x_{n+1}) = \frac{\log \dfrac{P(x_n x_{n+1})}{P(x_n)P(x_{n+1})}}{-\log P(x_n x_{n+1})}$$

where x_n denotes the appearance of a character in position n. An alternate explanation of this metric is the ratio of the mutual information between characters to the self information of the pair. When both logs are taken using the same base this measure is dimensionless. The top ranked letter pairs in my lexicon as scored by this metric are shown in Table 2.1. Based on the ranking of "QU" this metric is certainly acceptable and many of the other pairs make sense as well.

The pair cohesiveness metric can be used to construct longer sequences of interest. One may iteratively use the metric to find the most cohesive pair in the corpus and

Rank	Pair	Rank	Pair	Rank	Pair	Rank	Pair	Rank	Pair
1	QU	11	VI	21	OM	31	VU	41	NO
2	TH	12	OF	22	LY	32	TO	42	GN
3	ZZ	13	ND	23	BY	33	OW	43	BE
4	NG	14	IN	24	BU	34	WI	44	BL
5	JU	15	ON	25	BJ	35	HA	45	BO
6	WH	16	OU	26	VO	36	ZV	46	ED
7	CH	17	IZ	27	VA	37	OJ	47	IX
8	GH	18	FO	28	VV	38	EN	48	AX
9	XP	19	CK	29	VY	39	AN	49	OX
10	VE	20	JO	30	EX	40	UN	50	UX

Table 2.1: Some character pairs from a 33,000 word lexicon as ranked by pair cohesiveness.

treat it as its own unit. It is important that constituent units are then dissolved in case they were merely intermediaries to longer, more cohesive units. The beginning of such a run is shown in Table 2.2. Note that in step 13 the "NG" unit is dissolved while in steps 29 and 30 the "TH" unit is dissolved but immediately reconstituted. This suggests that "NG" serves only as an intermediary in creating "ING" while "TH" is in fact a strongly cohesive pair. Although I have developed this technique to form cohesive letter sequences, the metric is defined in general terms that can be applied to other atomic units such as words. For example, Table 2.3 shows the initial steps on sentences from the VOYAGER domain [92] for navigating within the city of Cambridge.

2.3.2 Sequence Cohesiveness

A shortcoming with the approach I have described is selecting an appropriate stopping criterion, particularly due to the dissolution of intermediate sequences. At any time the next sequence formed can be more cohesive than its predecessors. It would be preferable to compute the cohesiveness of *all* sequences simultaneously. This can be done by extending the pair cohesiveness metric to measure sequence

Step	+	−	Step	+	−	Step	+	−	Step	+	−
1	Q U		11	V I		21	J O		31	V O	
2	T H		12	O F		22	O M		32	V A	
3	Z Z		13	I NG	NG	23	C K		33	V V	
4	N G		14	N D		24	L Y		34	V Y	
5	J U		15	I N		25	B Y		35	A ND	ND
6	W H		16	O N		26	B U		36	E X	
7	C H		17	O U		27	B J		37	V O	
8	G H		18	I Z		28	C OM	OM	38	V U	
9	X P		19	F O		29	TH E	TH	39	T O	
10	V E		20	FO R	FO	30	T H		40	O W	

Table 2.2: Identifying variable-length cohesive sequences from a 33,000 word lexicon by iteratively applying pair cohesiveness. The "+" column indicates new units created while the "−" column indicates old units dissolved.

Step	+	−	Step	+	−
1	ice + cream		11	john f + kennedy	john f
2	hong + kong		12	royal + east	
3	mount + auburn		13	border + cafe	
4	mass + ave		14	cafe + sushi	
5	cajun + yankee		15	memorial + drive	
6	post + office		16	phone + number	
7	post + offices		17	telephone + number	
8	john + f		18	two + twenty	
9	f + k		19	two + fifty	
10	j + f k	f k	20	seven + seventy	

Table 2.3: Identifying cohesive word sequences in a corpus of transcriptions from a geographic navigation task. Sequences formed in steps 2, 3, 4, 5, 10, 11, 12, 13, 14, and 15 are all significant as landmarks in the domain.

Rank	String	Rank	String	Rank	String	Rank	String	Rank	String
1	E	11	D	21	W	31	VE	41	RE
2	T	12	P	22	K	32	LY	42	OU
3	I	13	U	23	IN	33	J	43	COMP
4	N	14	M	24	NG	34	ER	44	MP
5	S	15	H	25	X	35	XP	45	ZZ
6	R	16	G	26	QU	36	ON	46	ED
7	A	17	Y	27	ING	37	CO	47	JU
8	O	18	V	28	TH	38	THE	48	EXP
9	L	19	B	29	Q	39	HE	49	QUI
10	C	20	F	30	Z	40	EX	50	COM

Table 2.4: Character strings from a 33,000 word lexicon ranked highly by sequence cohesiveness.

cohesiveness:

$$C(x_n \ldots x_m) = \frac{\log \frac{P_s(x_n \ldots x_m)}{\prod_{i=n}^{m} P_c(x_i)}}{-\log P_s(x_n \ldots x_m)}$$

where $P_s(x_n \ldots x_m)$ is the probability of the sequence $x_n \ldots x_m$ and $P_c(x_i)$ is the probability of the character x_i. There are many ways to estimate these probabilities. Based on empirical studies I adopted simple estimates – dividing the frequency of a sequence by the total number of sequences and the frequency of a character by the total number of characters. Provided these totals are unequal, this results in a non-zero cohesiveness for individual characters. Table 2.4 shows the top-ranked sequences based on this metric. The ranking of letter pairs does not match that of pair cohesiveness because the methods of estimating probabilities differ. It is encouraging that individual characters are generally ranked high on the list since they are by definition fundamental units. It is interesting that the character string I considered an exemplar of significance is the only case of a sequence being more cohesive than its character constituents: "QU" has been ranked higher than "Q" alone.

Having scored all character sequences against one another using this metric, a subset of the sequences must be selected for inclusion in the corpus design. To do so I computed the cumulative lexicon coverage as a function of the number of multicharacter sequences selected in order of cohesiveness. A graph of this function is shown

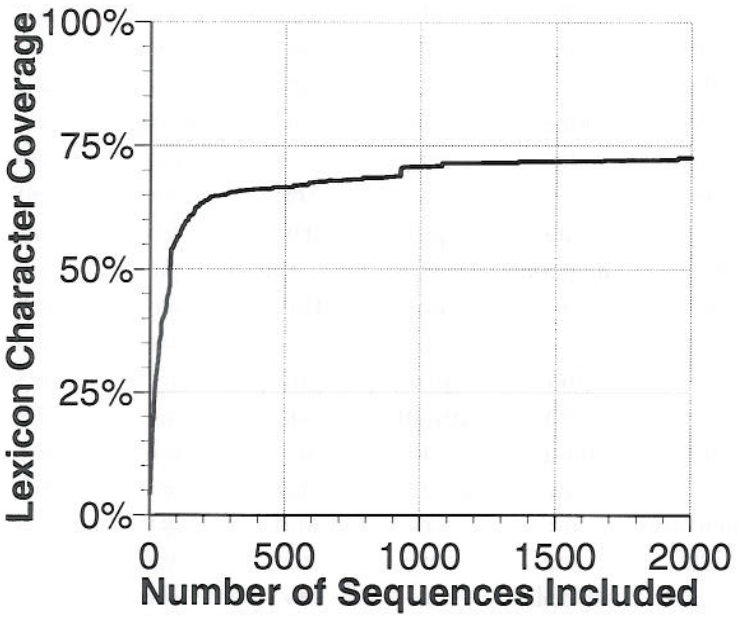


Figure 2.3: Percentage of characters in a large lexicon covered by the most cohesive sequences. Only the first 2000 multicharacter sequences are shown.

in Figure 2.3. Based on this I selected the 200 most cohesive sequences to account for most of the gain in coverage. Some of these are not needed due to subsumption by other members of the list. For example, the sequence "GRAPH" may be eliminated because it is wholly contained in "OGRAPHY."

Using cohesiveness to select character sequences for coverage provides a foundation for corpus design but it does not take into account all of the criteria of interest. Accordingly, I included a number of other sequences specifically to elicit handwriting confusions and contrasts:

- all 26 characters in the word-initial position, so that they may be used as a source of capital letters;
- the 23 allowable characters in the word-final position, to capture effects associated with finishing writing;
- the 16 available doubled characters (such as "tt"), both to capture effects associated with the doubling and to allow for a side-by-side comparison of letter

ability	dd	izing	ol	squ	vu	#o	h#
able	de	ju	oo	ss	vv	#p	i#
ably	ding	ke	ously	st	wa	#q	k#
alized	ee	king	over	ta	work	#r	l#
an	equ	la	ow	ted	zzl	#s	m#
ar	es	lc	pa	ter	#a	#t	n#
ate	exp	ling	pe	th	#b	#u	o#
ations	form	lization	pl	tically	#c	#v	p#
back	fully	ln	po	ting	#d	#w	r#
bb	gg	lo	pp	tively	#e	#x	s#
bu	ha	ma	pro	tr	#f	#y	t#
cc	he	mb	qualif	tt	#g	#z	u#
ch	hing	ment	que	uff	#h	a#	w#
ci	ho	mi	quizzic	um	#i	b#	x#
cl	ification	nc	re	und	#j	c#	y#
comm	ight	nn	ring	ur	#k	d#	z#
comp	ii	ography	rn	uv	#l	e#	
con	ingly	oi	rr	uzz	#m	f#	
ction	is	oj	sh	vi	#n	g#	

Table 2.5: Character sequences to be covered in designing a handwriting corpus. The # character indicates a word boundary.

forms;

- a set of 15 letter pairs which may be difficult to segment properly because they can be confusable with single characters, as shown below.

ci	in	ln	oj	vu
cl	io	lo	ol	rn
ic	lc	oi	uv	ri

These result in a total of 272 character strings, of which only 149 are required after allowing for subsumption. A complete list of these sequences is shown in Table 2.5.

2.3.3 Summary

One approach to corpus design is to cover a set of units. Typically these units have been fixed-length sequences of characters. I have proposed an alternative using sequences of variable length. These sequences are identified using a measure

of cohesiveness motivated by information theory. For the handwriting corpus being developed, I augmented 200 such letter sequences with strings of interest to my study.

2.4 Prompt Selection

Having decided on the character sequences to be covered in the corpus design, I now turn to selecting material used in recording the corpus itself. I will depend on a large body of text to provide potential material. The aim of the selection process is to choose a subset of texts from this body to cover the desired sequences as efficiently as practical. Maintaining character balance in the design is desirable provided it does not compromise the primary goals of coverage and compactness. In the past material selection has been done through introspection with computer assistance. The key to my approach is to recognize the task as a search problem which can best be performed by computer (with the researcher granting final approval).

2.4.1 Algorithm Development

A straightforward means of selecting material from the source lexicon is to do so randomly until full coverage is achieved. This procedure does not result in a particularly efficient design. In fact, if even a few rare sequences are to be covered, on average random selection will require that nearly the whole source be included. Much of the material chosen will contain only common sequences which have already been covered. Correcting this is trivial: simply reject randomly selected material that does not provide at least one previously unseen sequence. This results in a vastly more compact design than truly random selection.

This procedure still treats all material as equally desirable. Yet, some texts advance the completion of the design more than others. A more efficient approach favors candidates which provide the greatest number of needed sequences. At each step, score the possibilities according to the gain in coverage they would provide and select randomly from amongst the winners. This results in an even more compact design, but it still treats each *sequence* to be covered as equally needed.

When linguistic material contains an infrequent character sequence it tends to contain frequent sequences as well. In the extreme case of a sequence appearing only

once, the string which contains this sequence *must* be chosen to achieve complete coverage. In doing so the more common sequences in that text are covered without additional cost. This suggests favoring the selection of candidates containing infrequent sequences to avoid duplicating coverage of common sequences, a strategy opposite to that of the search procedure described above. This can be corrected by weighting character sequences inversely proportional to their frequency in computing the individual scores. Once a sequence has been covered its weight is zeroed.

The scoring procedure I used is somewhat more complex to improve compactness and balance. A merit score is computed for each word in a lexicon as described above. In addition, a demerit score for each word counts the number of redundant sequences provided. Rather than combine these scores, they are applied sequentially to favor compactness over balance. To achieve an optimally compact or truly balanced design requires a search procedure exploring multiple paths. I have sided with simplicity and instead perform a best-first search to achieve a step-optimal result with complete coverage. My algorithm for selecting words at each step is:

- Select the word with the most merits;
- When there is a tie, break it by selecting the candidate with the fewest demerits;
- When there is a further tie, randomly select from the shortest candidates;
- Adjust the merits and demerits of other words to reflect the selection;
- Verify the acceptability of the selected word by presenting it to the researcher.

2.4.2 Algorithm Evaluation

In order to evaluate the selection algorithm, I employed a task which is readily duplicated. The evaluation uses a 20,000-word lexicon, based on the Merriam Webster Pocket Dictionary [52], which has been used in other lexical studies. The sequences to be covered are letter pairs found in the lexicon. This list is sorted by frequency. The unweighted, equally weighted, and frequency weighted approaches described above are applied to cover the most frequent deciles of these pairs. The number of words and characters needed to achieve coverage is recorded. Both of the weighted cases use

the same merit-demerit selection algorithm. However, the unweighted case is handled differently to prevent a bias towards extremely short words. The experiments are repeated using the least frequent deciles.

Each test is based on 10 runs with the lexicon reshuffled between them. The mean results are shown in Figure 2.4. Fewer than 200 words, just 1% of the lexicon, can provide complete coverage with the frequency weighted search. In fact, all pairs can be covered using the same number of words as is needed to cover only the least frequent 60% of pairs (though somewhat longer words are required, as shown by the number of characters selected). In all cases the effect of rare pairs dominates. Frequency weighting provides the greatest advantage when we have a mix of common and rare letter pairs to cover. I believe this is because there is little latitude in selecting words to cover rare pairs and much freedom in covering frequent pairs. With only rare pairs the words required are virtually prescribed. With only common pairs nearly any selection will be reasonable. Any of the techniques investigated is preferable to true random selection, which required an average of 19,017 words to cover all letter pairs.

2.4.3 Algorithm Application

The first step in selecting words for my corpus design is to acquire an appropriate source of text. I began with the 33,000 word lexicon described above. Some of these words are unsuitable for the design. Words with fewer than 3 or more than 15 characters were excluded. Because of the way I anticipated collecting data, words that are hard to spell or ambiguous [40] were also removed from contention. Finally, words which may be deemed offensive, controversial, or otherwise unacceptable were eliminated during the selection process. The remainder, approximately 23,000 words, comprise the source material to choose from.

The frequency-biased selection algorithm was applied using the variable length sequences previously selected. These units were permitted to overlap, and single-character units were added to fill the remaining voids in each word. Sequences were excluded from the word-initial position unless they appeared nowhere else. Complete coverage of these units could be achieved using only 53 words containing 474 charac-

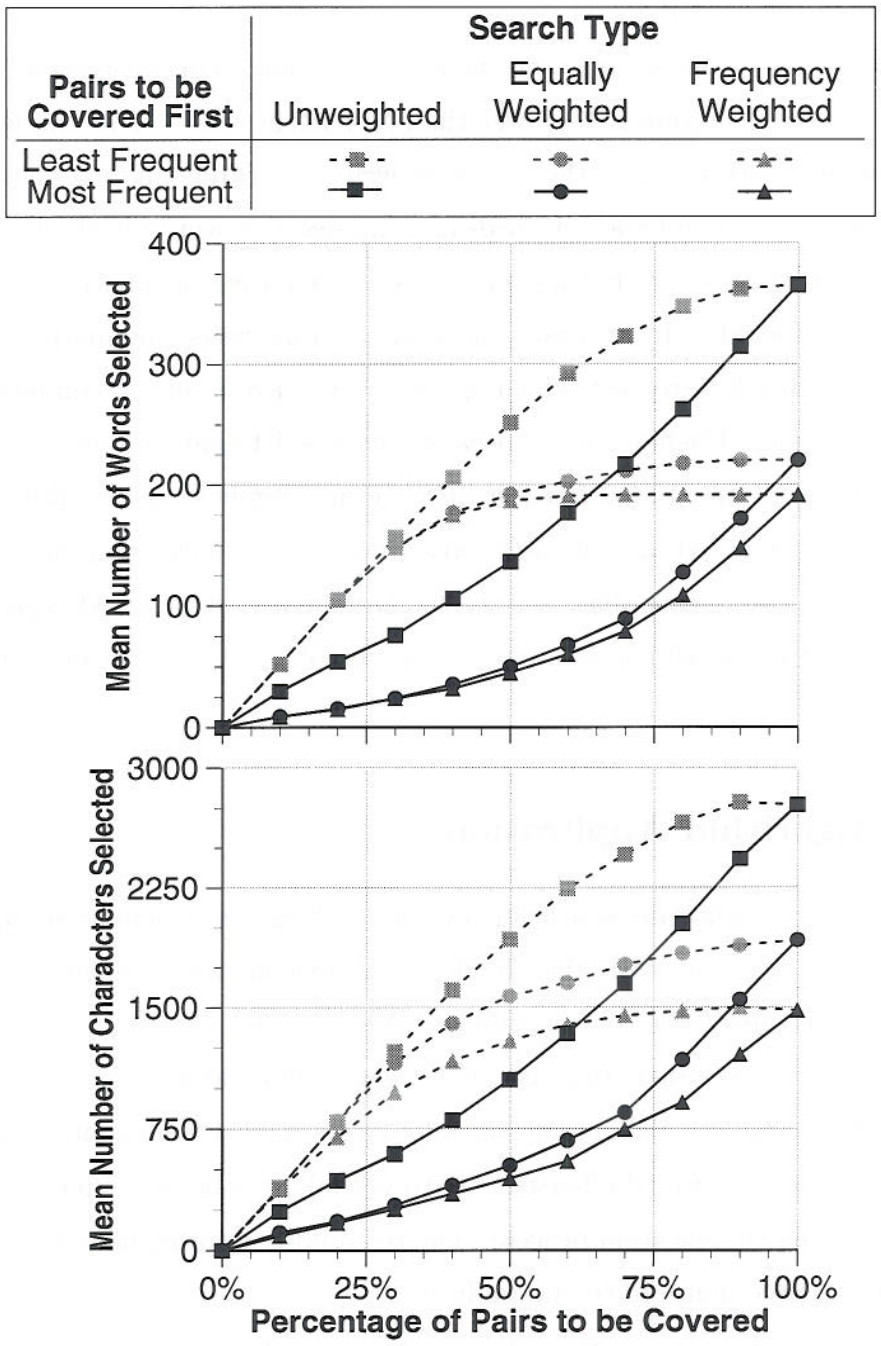


Figure 2.4: Mean number of words (top) and characters (bottom) needed to cover character pairs extracted from a 20,000 word lexicon as a function of the percentage of pairs considered.

Accountability	Disqualified	Justifications	Rejuvenating	Transform
Agonizingly	Embraces	Kidding	Revving	Uncomfortably
Announcing	Fabulously	Lump	Seeker	Unexpected
Approaching	Frightfully	Mate	Shadow	Unworkable
Backing	Fuzz	Menu	Skiing	Vanquish
Cafeteria	Geography	Normalization	Spoiling	Volcanic
Commanding	Governing	Omitted	Surrounded	Wobble
Comparatively	Hugging	Projections	Swab	Xylophone
Complex	Inconsequential	Puff	Sympathetically	Yearbook
Declaring	Industrialized	Puzzlement	Taxi	Zero
Decompress	Invulnerable	Quizzically		

Table 2.6: Words to be used for data collection to achieve compact coverage of significant letter sequences.

02066	16380	35124	54331	79158
05521	23687	45922	60839	86773
07856	27657	47190	61449	88253
10342	29697	48170	72898	94095
13262	30464	50011	74184	99375

Table 2.7: Numbers to be used for data collection to achieve compact coverage of digit pairs.

ters. This is quite compact given that the coverage criteria include observing every letter in both initial and final positions. Even fewer words could have been selected if I had been less conservative in rejecting potentially objectionable material. A list of the words selected is shown in Table 2.6.

My corpus design also includes digit strings. A lexicon was generated containing all 100,000 5-digit numbers. The sequences to be covered included all 100 digit pairs along with each digit in the initial and final positions. The selection algorithm was applied to choose a subset of the numbers for the corpus as shown in Table 2.7. For this contrived task, the selection algorithm managed to cover all sequences in the fewest strings possible.

2.4.4 Summary

I have described a procedure, incorporating multiple criteria, for selecting material to be used in corpus design. The key idea presented is that coverage of rare phenomena should be given precedence to enhance the compactness of the result. I have applied these principles to select prompts for a handwriting corpus, yielding 53 words and 25 digit strings.

2.5 Collection Methodology

As I have described, the procedures used to collect data can have a substantial impact on the handwriting recorded. In collecting my data I have taken particular care to control unwanted influences. This is reflected in the facilities for handwriting capture, the protocol used with subjects, and the task subjects performed. A key difference between my data collection and previous efforts is the way subjects were prompted. The commonly taken approach of presenting prompts visually risks contaminating the handwriting data by influencing character shape and size. Additionally, framing data collection as a text copying task requires the subject to shift their visual attention repeatedly from the prompt to their writing, an action unlike the fixed focus during spontaneously written material. To avoid both of these deficiencies, I elected to present prompts *aurally*. One female speaker recorded the required phrases, both pronouncing and spelling each word. This material was digitized to permit consistent playback at will.

No suitable programs for handwriting data collection were available for this study. To avoid the arduous task of writing a low-level driver for a tablet and display, I based my software on the Windows for Pens operating system. My task was further simplified by using Visual Basic with Pen Extensions for constructing the application. Data collection was performed using a Compaq Deskpro 486/33 computer equipped with a Sound Blaster Pro audio board and Sony MDR-V6 headphones.

Handwriting was digitized using a Wacom model HD-648A tablet with integrated LCD display. This tablet works with a cordless stylus that does not require a battery. Although pressure sensitive styli are available, I chose not to use one due to their

bulkier design.[2] The standard pen for this tablet includes a barrel button which can alter the writer's grip. Accidently pressing this button interfered with data capture. For these reasons, I deemed necessary a special pen, model SP-200A, which lacked this button.

Subjects were recruited primarily through posters placed in hallways throughout M.I.T. Competency in English was required, but subjects did not have to be U.S. natives. Modest compensation was provided in return for participation in the experiment. Prior to data collection, subjects read and signed a release form using ordinary paper and pen. Writers were seated at the digitizing tablet and permitted to position it to their liking.

Additional instructions were presented on the tablet's display as shown in Figure 2.5. Progress was controlled by the subject through on-screen buttons which were located centrally to reduce the left-right bias. Incorporating such buttons in the instruction process begins the user's acclimation to the stylus and tablet. Subjects were instructed they would be asked "to listen to someone speaking and to write down what you hear" on the tablet. The writing would be recorded by the computer for "later analysis." No further details of the experiment's purpose were provided until after data collection.

Next, instructions were provided on the use of the tablet. Subjects were asked to hold the stylus as they would a pen. Only the stylus tip was detected, allowing the hand to rest on the tablet's surface. Subjects were asked to write only in the spaces provided. If they made a mistake or were otherwise unhappy with their response, the subjects were instructed to erase their writing (using an on-screen button) rather than correct it. All responses were to be printed, but it was up to each subject to interpret this writing style. To ensure that they were comfortable with the stylus and tablet, subjects first were encouraged to write and draw whatever they wished within a large area until they were ready to begin.

The data collection process was divided into four phases according to the type of data recorded. The first phase was for words. Subjects were asked to capitalize the

[2]It is reasonable to choose not to capture the pressure information. In preliminary studies I found subjects maintained a fairly constant pressure through much of their writing, perhaps because this is required for everyday writing implements.

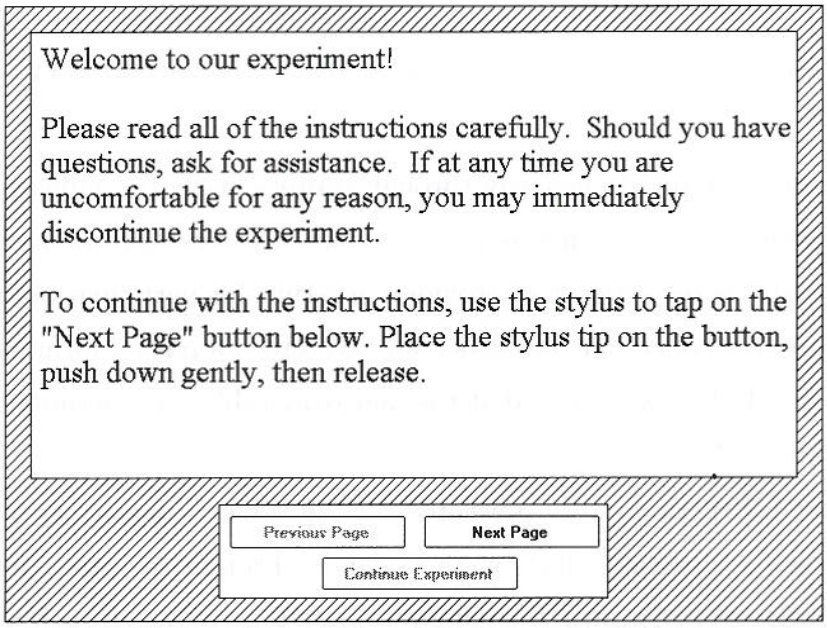


Figure 2.5: Example data collection instruction screen.

first and only the first letter of each word. Because the corpus design featured each letter in the word-initial position, this should provide a complete source of capital letters for study. Each word was played once automatically but could be repeated as often as desired by pressing a button. A second button would play a spelled version of the prompt. Playing either recording erased any writing already present. Subjects could write only after a prompt had finished playing. These measures ensured that the writing process was uninterrupted.

The screen used for this phase is shown in Figure 2.6. The writing area was made as large as practical and dominated the tablet's screen with the bottom margin reserved for control buttons. No guides were used, allowing the subject to position and size their responses as desired. The area was cleared between responses as the subject advanced through the recording agenda. The first two words requested from each subject were for calibration purposes and not intended for recognition studies, but the subjects were not aware of this. In addition to allowing the subjects to acclimate to the experiment, the calibration words "Acknowledgment" and "Fake" exhibited the range of string lengths seen in the remaining prompts. Thus the subjects

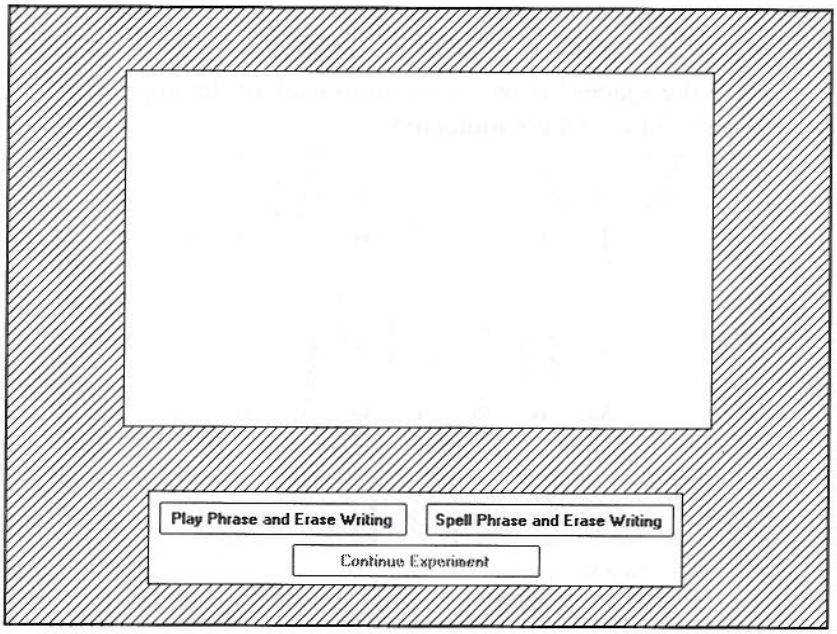


Figure 2.6: Data collection screen for words and numbers.

could adjust their writing size to the task without requiring an abrupt change mid-collection. Subjects were monitored while writing these two words and reminded of the instructions if necessary. Subsequently, no monitoring was performed while the 53 actual prompts were presented in random order.

The second phase of data collection was run similarly to collect the 25 digit strings. Since by this time the subject should have been comfortable with the experiment, no calibration prompts were used.

In the third phase of data collection, subjects were asked to provide examples of how they wrote each character. Three screens were used to collect this data, one each for upper-case letters, lower-case letters, and digits, as illustrated by Figure 2.7. Each screen contained one labeled box per character in the set. Subjects could erase their input one box at a time as needed.

The fourth and final phase of data collection was a single screen form filling task, shown in Figure 2.8. The handwriting captured was not used in my experiments, the true purpose of this phase being to record biographic information. However, the questions were selected so as to provide not only useful information but a range of

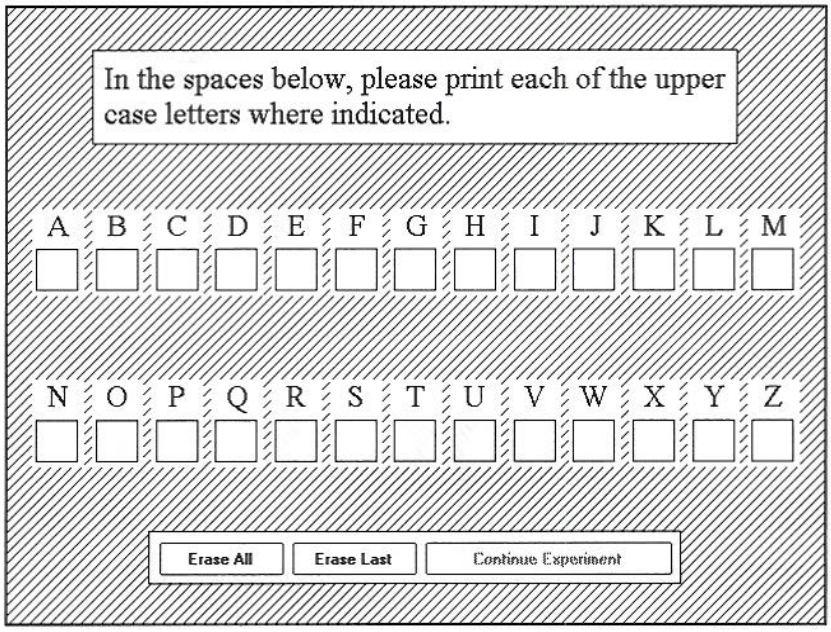


Figure 2.7: Example data collection screen for boxed characters.

response styles. This data provided a number of surprises. For example, subjects were asked to report their gender in a small box. While most subjects responded with an "M" or "F," some used the astronomical symbols for Mars or Venus. Each subject required 20-30 minutes to complete the data collection procedure.

2.6 Data Preparation

Handwriting was collected from 159 subjects. The biographic information was transcribed and entered into a database. Subjects ranged in age from 12.5 years to 63 years, with a median age of 25 years. Approximately 62% of the subjects considered themselves to be students. Non-natives comprised 16% of the writers, dominated by 5 individuals each from Canada, England, and India. 56% of the subjects were male and 87% of the subjects were right-handed. The subjects were assigned to one of 4 categories, training (105 subjects), development testing (30), evaluation testing (15), and spare (9), maintaining the balance of gender and handedness to the degree possible. Both test sets are used to determine system performance on "unseen" data. However, a system will be adapted to the testing material when it is repeatedly

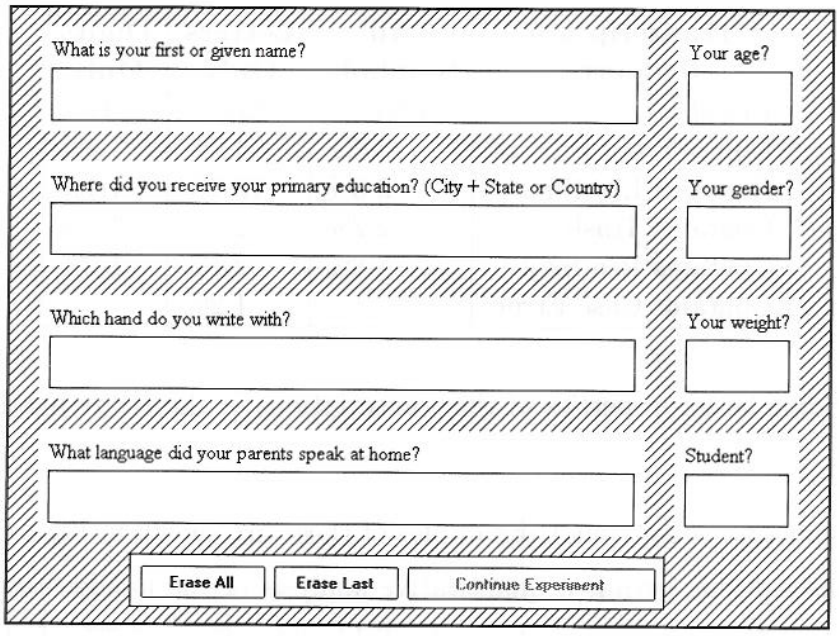


Figure 2.8: Data collection screen for the biographic form.

appraised and improved using a single test set. The evaluation data is held until the very last of my studies to be used as truly unseen data for reliable performance measurement. Ideally, this set would be used for evaluation only one time.

All of the word and number data were transcribed using software written specifically for the task. The responses were displayed individually at the size they had been written. The prompt text was supplied as a default string to be edited by the transcriber as a means of saving effort. Comments could be included to note unusual phenomena. I transcribed the bulk of the data, but potential test material was handled by an impartial transcriber.

The transcription strings were aligned with the handwriting using a tool generously provided by the Microsoft Pen Computing Group. This tool was restricted to aligning data at the stroke level and so could not properly process connected characters. The aligned transcriptions were checked and corrected using another custom application which could divide strokes between characters. Each data sample was assigned to exactly one transcription token. Special symbols were included to designate ink not part of any character, namely ligatures, pen skips, and trash (consisting of

Transcription Property	All Symbols	Letters Only	Digits Only
Count	*12402*	*8427*	*3975*
Differs From Prompt	3.7%	4.7%	1.4%
Contains Ligatures	9.8%	13.9%	1.1%
Contains Trash	2.2%	2.7%	1.1%
Contains Pen Skip	1.8%	2.1%	1.2%
Contains Case Error		1.2%	

Table 2.8: Some basic properties of the transcriptions in the entire handwriting corpus.

embellishments and corrections). Examples of these are shown in Figure 2.9.

The transcriptions permit a quantitative analysis of the handwriting corpus. A summary of some properties is shown in Table 2.8. Some 3.7% of the responses differed from the prompt text, excluding special symbols inserted for transcription purposes. This is a relatively high number compared to what we might expect using visual prompting, and it is high considering that we took care to select prompts which were unambiguous and easy to spell. The most error-prone prompt, "Rejuvenating," was misspelled by over 25% of the subjects with "Quizzically" and "Comparatively" close behind. Most often the error was substituting a single character for another. Approximately one quarter of the word errors were in case alone. The error rate for words was over three times that of numbers. As one might expect, the nature of the numeric errors is quite different from the alphabetic errors. Rather than substitution, most often neighboring digits were exchanged. The pair swapped was not uniformly distributed and almost never appeared in the final position. Interestingly, a transposition error in the initial position was sometimes corrected by writing characters out of order as illustrated by Figure 2.10. The initial character written is the second digit of the prompt. The next character written is the first digit of the prompt, and it is placed immediately to the left of the initial character. The remainder of the string is written in standard order.

Not all of the subjects complied with the data collection instructions. This can be shown through statistics on transcriptions from their data. In Figure 2.11 I have shown the error rate for each subject. Most of the errors are in capitalization and

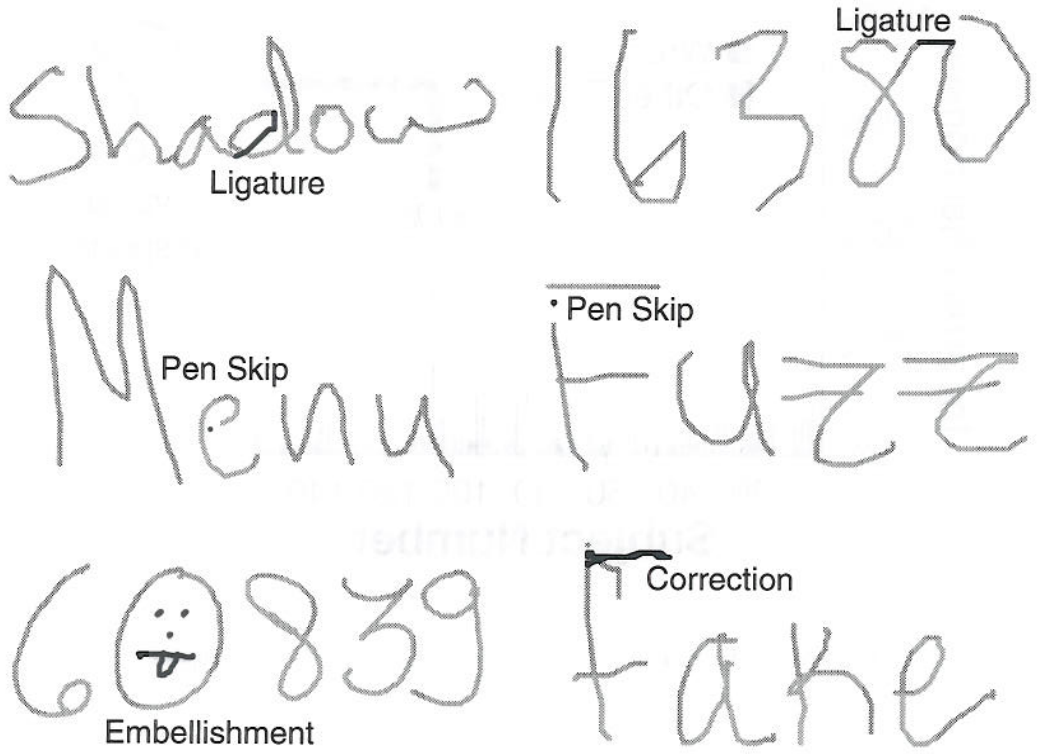


Figure 2.9: Examples of special writing.

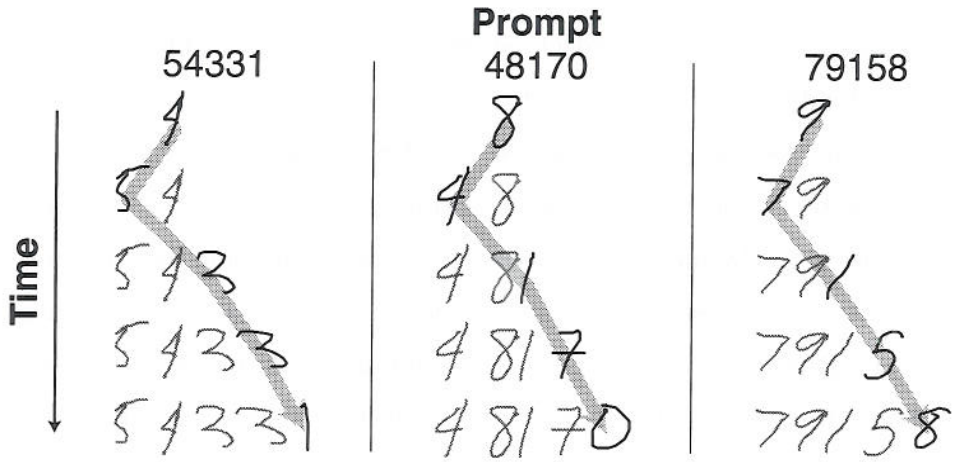


Figure 2.10: Example numbers containing an initial digit pair transposition corrected through altered writing order.

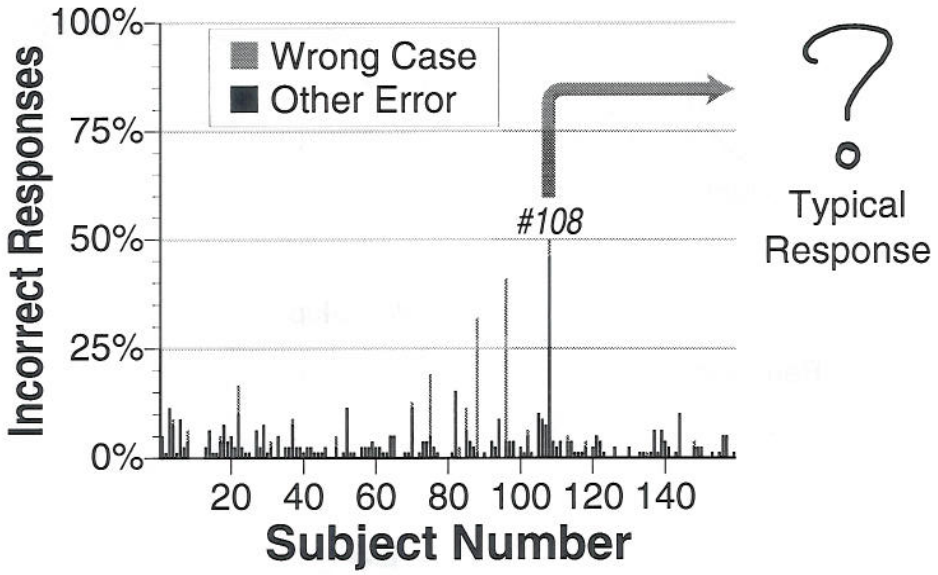


Figure 2.11: Error rate for each subject in the corpus.

should not have significant impact. However, subject 108 not only made more errors than any other subject, these errors were typically more severe. For some prompts the subject responded quite aberrantly, as shown. Roughly 77% of the subjects connected some of their characters. This is permitted in the data, but cursive writing is not. As an objective measure of cursive writing, I examined the percentage of characters connected by each subject, shown in Figure 2.12. One subject, number 133, connected substantially more characters than was typical. To err on the side of caution, four others subjects were marked as potential cursive writers. In transcribing the data, one subject was identified as providing particularly "creative" responses as shown in Figure 2.13. While strictly meeting the data collection instructions, this was viewed as writing quite distant from what is natural or typical. Handwriting from the seven subjects mentioned above was rejected and not examined further. Replacement subjects from the "spare" designation were selected to best match the biographic profile of each undesirable writer. Writing from the two remaining spare subjects was not used. In addition, any responses that contained corrections or embellishments were set aside. Although graceful handling of such data is important for deployed applications, it is beyond the scope of this study. The net size of the handwriting

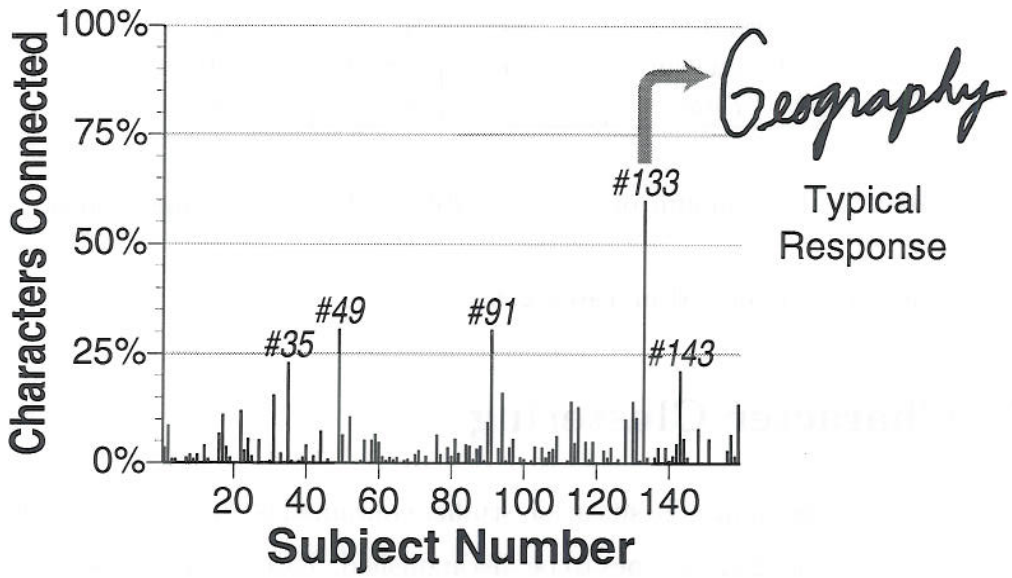


Figure 2.12: Percentage of characters connected by each subject in the corpus.

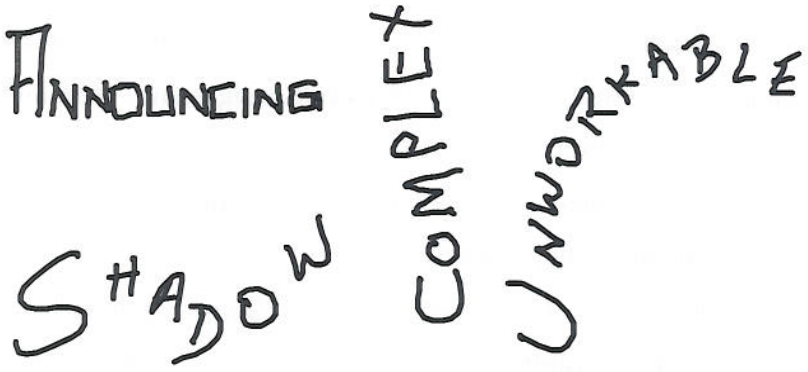


Figure 2.13: Creative writing provided by one subject.

Designation	Subjects	Tokens	
		String	Boxed
Training	105	60,767	6,510
Development	30	17,651	1,860
Evaluation	15	8,734	930
Total	*150*	*87,152*	*9,300*

Table 2.9: The amount of data available in the handwriting corpus.

corpus I collected is described in Table 2.9.

2.7 Character Clustering

The alphabet used in my classification studies contained 62 symbols. This does not imply that there were 62 archetypes to be distinguished. Examples of some characters, such as the letter "O" and the digit "0," may be indistinguishable, decreasing the number of archetypes. Other characters, such as "z" or "7," may be present as several allographs, increasing the number of archetypes. A multitude of factors influence the choice of allograph written. In the extreme, one may claim that *every* example of a character is its own allograph because no two are produced under identical conditions. The question of what constitutes an allograph is really a matter of degree in similarity.

It would be useful to catalog the extent and manner of allographic variation even without fully comprehending its causes. Except for the most common examples, this is a laborious task requiring many subjective judgments. Furthermore, it may be difficult for humans to discern contrasts in handwriting production which do not influence the character's image. This is only a limitation if such variation is viewed as critical to distinguishing some allographs.

If a metric can be defined to compare character shapes, it may be used to objectively identify allographs through a clustering procedure. The results would not be unique; alternate metrics and clustering algorithms might suggest different allographs. A representation I describe on page 112, resampling the handwriting at a fixed number of equally-spaced points, is useful for comparing characters because it permits a one-to-one alignment between pen trajectories. This reduces the problem of measur-

ing the distance between character shapes to measuring the distance between points. I chose to calculate the mean Euclidean distance between corresponding points in each character. To provide clarity in the displayed allograph shapes, each symbol was encoded using 32 points.

All of these experiments were based on the k-means clustering algorithm [83] applied to the training set. An initial cluster was formed containing all characters to be considered. At each iteration of the clustering algorithm, a new cluster was seeded with the worst outlier of any existing cluster. This favors the construction of smaller clusters modeling unusual shapes over improving the fit to the bulk of the data. Clusters and means were re-estimated until convergence was reached. No stopping criteria were used other than visual examination of the clusters formed.

The clusters found for five characters are shown in Figure 2.14. Crosses indicate one standard deviation around each point and an arrow marks the final pen-up. In all cases certain allographs are strongly favored over others as indicated by the number of tokens assigned to each cluster.

Four variants of "T" are identified. In all cases the vertical stroke proceeds downward, but the horizontal stroke may go in either direction. Either of the strokes may precede the other. The second allograph was written only by right-handed subjects while the third and fourth were exclusive to left-handers. Four variants are also identified for "d." The first two have the bowl written before the stem and are quite similar. This demonstrates how small displacements can result in large cumulative distances with this metric. In the third case the stem precedes the bowl. The final case is a "D" written as a small capital. Of the three "a" allographs shown, the second is also a small capital form. Both "9" and "f" show variations in stroke order. The third, rare variant of "9" is particularly interesting in that its bowl is constructed in two parts and its stem greatly curved.

The same clustering procedure was applied at a larger scale to identify prominent character shapes. To prevent lower-case characters from dominating the results, a balanced data set was constructed to contain 92 randomly selected examples of each character, foregoing some variability. From this, 62 clusters were formed as shown in Figure 2.15. Each cluster is identified by the three most frequent labels for the characters it contains, shaded to indicate their frequency. The shape most strongly

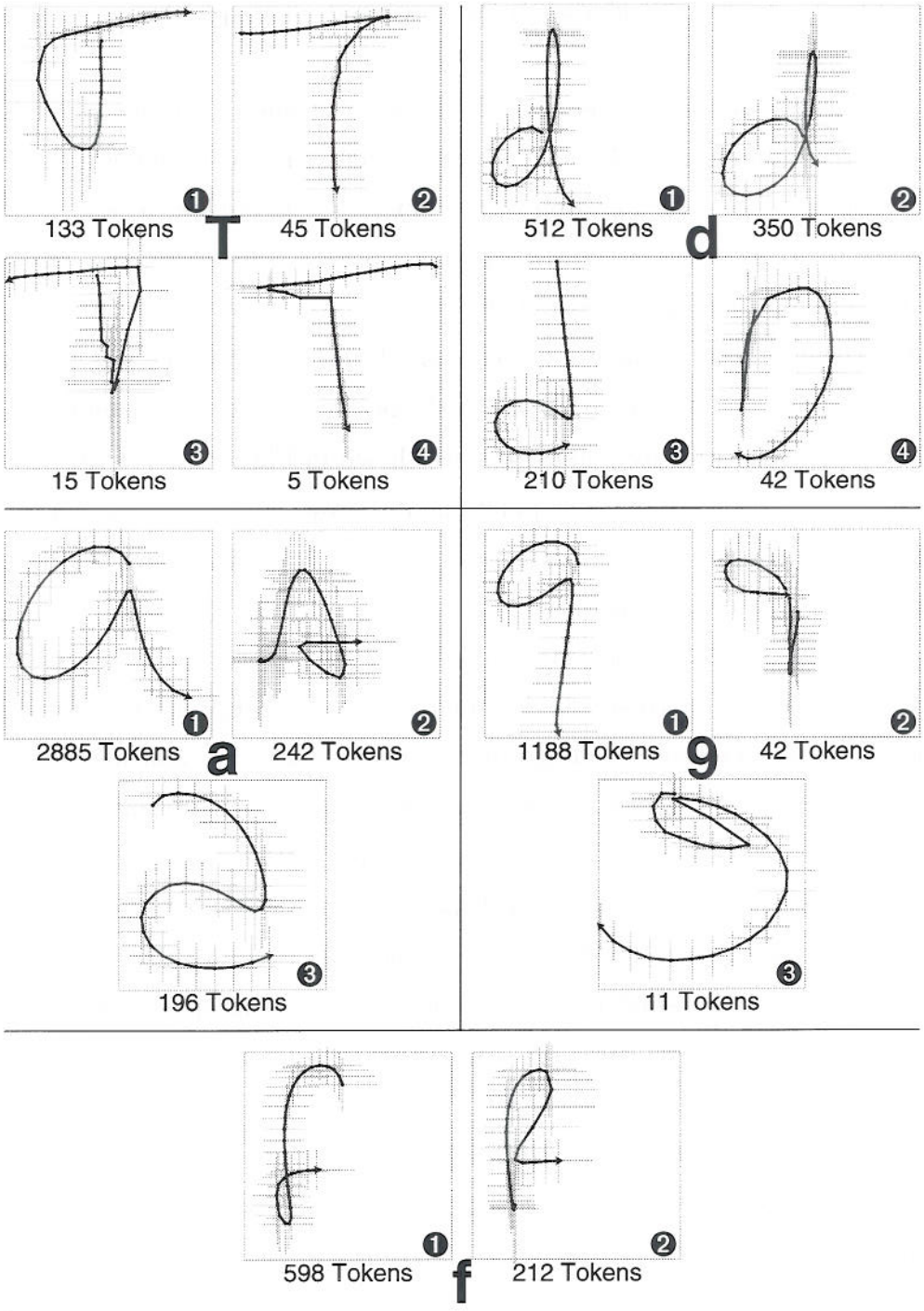


Figure 2.14: Allographs of five characters identified by k-means clustering.

associated with a single label is an "e" in the third row. Several clusters were assigned to circular shapes; in each case different starting points have resulted in displacements along the entire curve, making the shapes distinct under the chosen metric. These clusters are quite different from the mean representations of the 62 characters, shown in Figure 2.16. Unusual shapes, for example those of "I" and "z," indicate that the mean does not fit the data well. Such characters are likely to have two or more common allographs, as can be seen by referring back to Figure 2.14.

2.8 Summary

In this chapter I have described the development of a handwriting corpus suitable for my intended studies. I have described some of the factors which influence handwriting production and should be taken into account when collecting data. The specification of the corpus compactly covers a set of significant character sequences. The sequences were chosen using a metric motivated by information theory. A search procedure identified words to cover these sequences efficiently, and the compactness of this design was improved by favoring the selection of rare constituents first. Additional prompts were constructed in a similar manner for digit strings.

I paid particular attention to the data collection environment to reduce unwanted influences on the subject's handwriting. The digitizer selected provided a stylus similar to common writing implements. A large writing area was provided to ensure that writing was produced at a comfortable size, position, and orientation. A key innovation was the use of aural prompts to avoid exposing subjects to character prototypes. This, combined with describing the required writing style only as "printing," leads to a natural range of letter forms.

Aligned transcriptions were produced for the words and numbers collected. Based on these, I excluded handwriting unsuitable for this study. The result is a data corpus rich in information and reflecting natural handprint variability.

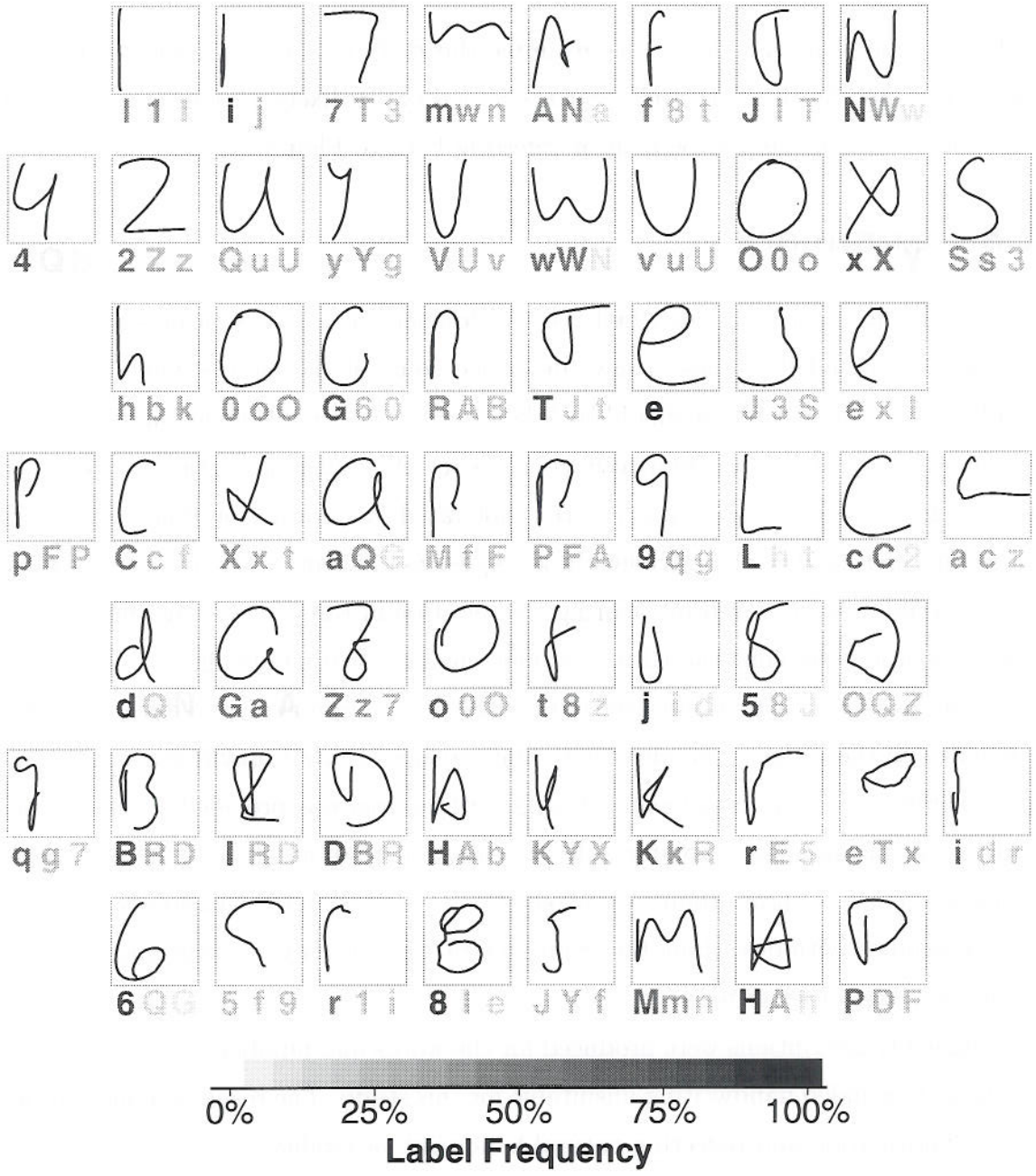


Figure 2.15: 62 prominent character shapes identified by k-means clustering.

Figure 2.16: Mean representations of 62 characters.

Chapter 3

Comparing Representations Through Classification

An important decision in constructing a handwriting recognition system is selecting an appropriate data representation. In this chapter I describe a number of data representations and compare character classifiers based on them. I begin by presenting the results of an authentication study to establish human performance on this task. I then describe the classification approach taken and detail the representations tried, touching briefly on allographic variation and glyph similarity. Having determined the leading representation, I show how tuning its parameters improves classification accuracy.

If the primary goals of this thesis involve handwriting *recognition*, why am I performing *classification* experiments? One may view the recognition process as a sequence of three steps. The input signal is first segmented into regions corresponding to characters, perhaps including alternate paths to allow for ambiguity. Next, each segment is classified to determine how well it resembles the character prototypes seen in training. The final step searches through the lattice of labeled segments to find the most likely candidate string.

Unlike the psycholinguistically motivated phonemes of speech recognition, letters are undeniably the building blocks of written text. We are taught writing by forming individual characters rather than whole words. Except for homographs, a text's meaning can be adequately reconstructed from its letters alone. Thus it is safe to assume that some reasonable letter segmentation technique is possible. This allows us to decouple the recognition steps for research purposes by depending on the best

single-path segmentation available: a hand transcription of the data. The deterministic nature of the transcription reduces the number of segments to be considered, greatly decreasing the time needed to perform experiments and allowing more representations to be examined. In addition, fewer factors affect the results because automatic segmentation errors and recognition system control parameters are eliminated. Still, it is worth remembering that an otherwise lackluster representation may be well suited to the idiosyncrasies of a particular segmentation scheme. Components are integrated and optimized collectively in the best recognition systems.

3.1 Data Authentication

The fact that the handwriting corpus could be transcribed demonstrates the legibility of the data. However, reading character strings is a complex process incorporating higher-level knowledge in areas such as vocabulary, grammar, and domain. Even at the symbol level, otherwise ambiguous shapes can be uniquely identified when compared to their fellow characters. Because of the many external constraints brought to bear on the reading task, it would be wrong to presume the data to be equally unambiguous at the symbol level. Yet the degree to which this is true has great impact on the character classification problem's difficulty, since the decision criteria stem only from each symbol itself. Accordingly, I have conducted an authentication study to assess the legibility of individual characters excised from their context.

3.1.1 Procedure

Authentication was performed only for string and boxed data in the development and evaluation sets because these were the potential test sets. This handwriting was divided among three paid authenticators. The experiment was conducted with special purpose software illustrated in Figure 3.1. Each character was displayed in isolation, centered in the uppermost pane, at the size it was written. Tokens were shuffled to ensure that sequential characters had neither the same writer nor potentially confusable labels, but this level of detail was not disclosed to the authenticators.

The authenticators were told that the data would be shown in random order and relative size was unimportant. Each item presented would consist of exactly one

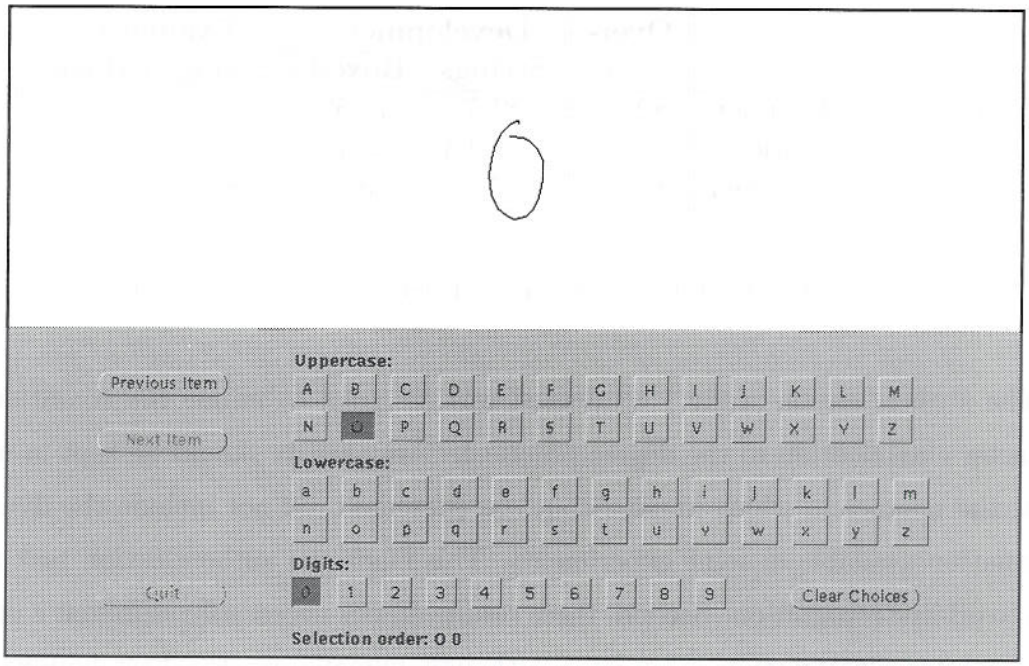


Figure 3.1: The display used for handwriting authentication.

character. On-screen buttons were to be toggled to indicate, in preferential order, any number of labels appropriate for each token. Prior responses could be recalled and edited as necessary. Authenticators were self-paced and paid hourly to ensure that fatigue was not a significant problem.

3.1.2 Results

The authenticators labeled over 29,000 tokens. Their responses were compared to the transcription labels for each token. A summary of the results is shown in Table 3.1. Approximately 1.2 labels were assigned to each character. From the standpoint of the classification experiments I have run, the development strings were the most important portion of the handwriting corpus. Nearly 82% of these tokens were correctly identified by the authenticators according to their top-choice labels. An additional 6% of these answers consisted of the correct letter but were the wrong case. Considering all labels assigned to each character, the correct response was among the authenticators' choices only 87% of the time. Interestingly, case substitution was twice as common for boxed data.

	Over-All	Development		Evaluation	
		Strings	Boxed	Strings	Boxed
Top-choice Accuracy	82.0%	81.7%	76.5%	84.1%	78.9%
Case Substitution	6.6%	6.1%	13.1%	5.4%	12.5%
Correct Label Listed	87.6%	87.0%	85.4%	89.2%	86.4%

Table 3.1: Results of the data authentication experiment.

The authentication study I conducted was designed to mimic the conditions of character classification to the degree practical. Two caveats should be kept in mind. First, the manner in which handwriting was presented did not include the dynamic information available in some experiments. This made the authentication task more akin to off-line recognition. Second, these results do not necessarily represent a ceiling for character classification accuracy. Although humans are the best handwriting recognizers in existence today, there is no reason to assume that a computer could not exceed their level of performance. Also, the task of identifying characters in isolation is quite different from the string recognition which people perform so adeptly. Despite these limitations, the authentication results provide the best means available for gauging the difficulty of the classification task at hand.

3.2 Methodology

There are many factors which influence a classification experiment, even the classifier itself. In theory, one could search through all combinations of all options to the training procedures, classifiers, and representations to identify the highest accuracy system. However, there are far too many possibilities to make this problem tractable. Furthermore, accuracy is only one measure of system performance, and it may not even be the most important property from a user's perspective. Rather than wringing the best possible accuracy from a few representations through incremental gains, I have focused on sampling the sub-optimal performance of a greater number of representations.

3.2.1 Symbol Inventory

One of the basic decisions made was the unit to be classified. I selected characters because they are the smallest unit for which the label inventory is generally agreed upon[1]. Because aligned transcriptions were created for the handwriting corpus, it was a trivial matter to extract the data and label associated with each character.

Other possibilities do have their advantages and were not overlooked. A smaller unit such as strokes could result in more robust models by sharing data from many symbols. Labels could be constructed based on stroke order within each character. Similar strokes could then be clustered. However, care must be taken lest unacceptable combinations of strokes be considered a character. Furthermore, it would be difficult to classify strokes meaningfully without considering their position relative to their neighbors. Units larger than characters are also viable. Digraphs, or even words, could better capture contextual variation and ordering constraints. However, as the size of the classification unit grows the average amount of training data per model decreases, reducing the robustness of the classifier. There is no reason to be limited to a single type of unit. For example, a system might be based primarily on characters but include models for the most frequent multi-character sequences. This approach is sometimes taken in speech recognition [45] by explicitly modeling the short but highly variable function words in an otherwise phoneme-based system.

3.2.2 Classifier Technology

The choice of classifier used in a recognition system is intimately tied to the data's representation. The best combination of accuracy and efficiency is typically achieved when the models' parameterization closely matches the underlying distribution of the feature vectors. Because I was not concerned with system speed, I traded computational efficiency for added flexibility. This allowed a single classifier to be used for a wide-range of representations, eliminating model disparity as a factor affecting accuracy comparisons. In addition, a flexible model could implicitly capture allographic variation.

[1]Although one may argue that that inventory of allographs is unknown, only the character grapheme need be identified for recognition purposes.

Rather than implement a classifier, I have taken advantage of existing technology. My experiments are based on a component of MIT's SUMMIT [93] speech recognition system. This classifier incorporates mixture Gaussian models [50] with diagonal covariance matrices. Input vectors are rotated using a principal components analysis [35] and scaled to normalize the average within-class covariance. The classifier implicitly incorporates unigram statistics of the training data.

Two parameters of the classifier control the number of components used in each mixture model. An absolute maximum limits the possible number of components per class. While more components can improve the fit of the model to training data, too close a fit may not generalize well to unseen data. More components also increase the computation required and the number of model parameters to be estimated. The other parameter specifies a minimum number of training tokens assigned to each mixture. This ensures the robustness of each model at the expense of a poorer fit for less common variants.

Values for these parameters ideally would be chosen independently for each representation to yield the greatest accuracy, but this is a computationally expensive procedure. Instead I applied a single set of values to all experiments. These were determined empirically using a bitmap handwriting representation described in section 3.3.1. The results of this study are shown in Figure 3.2. Note that the accuracy scale is condensed to accentuate differences. Permitting only a single component per class, the equivalent of a Gaussian classifier, did not perform as well as allowing for mixtures. As expected, accuracy waned when conditions allowed for many components trained on very few tokens. The classifier was not particularly sensitive to the values used except for extremes, but somewhat higher accuracy could be achieved by permitting relatively few tokens per component. Based on these results I chose a maximum of 128 mixtures per model and a minimum of ten samples per mixture.

3.2.3 Experimental Procedure

In order to make comparisons between handwriting representations meaningful, it is crucial that identical procedures are maintained across all experiments. Central to this are consistent training and testing data sets. For the writer-independent

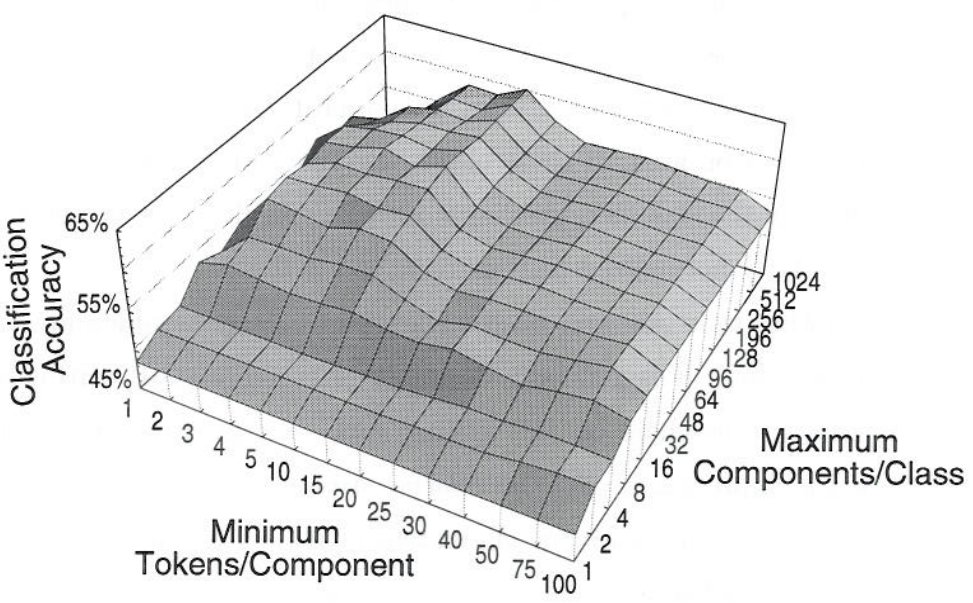


Figure 3.2: Character classification accuracy as a function of control parameters to the Gaussian mixture classifier.

technology investigated, it is equally important that testing subjects be disjoint from training subjects. Except where noted, the classifier for my experiments was trained on alphanumeric characters from strings in the training set and tested on similar data in the development set. Boxed data was treated separately in adjunct studies. The evaluation data was held pristine until the very end of my investigations.

It was also important to manage differences between experiments so that performance gains could be attributed to single factors. Handwriting recognition systems often incorporate one or more preprocessing steps such as slant correction. The intention of these operations is to reduce the data's variability, although errors in preprocessing can introduce their own noise. The value of these algorithms is not necessarily uniform among representations. For example, detecting and correcting character orientation is unnecessary for representations insensitive to rotation. These differences could confound analysis of the results.

For this reason, I eliminated all but a simple preprocessing step: each character was translated and scaled to fit within a unit bounding box, preserving the aspect ratio. This eliminated absolute size and position information. Due to the limited

nature of the handwriting corpus analyzed, such information might otherwise be exploited to improve classification accuracy. For example, because upper-case letters usually occur only at the start of each word, they tend to be written towards the left side of the data collection area. If these biases were not eliminated they could have been incorporated by the classifier, yielding misleading results. The duration of each character was similarly normalized to 1, starting at $t = 0$.

It is worth noting that size and position are particularly troublesome areas for handwriting recognition. It is generally desirable to accommodate writing at a variety of sizes located anywhere on the writing surface, suggesting that some type of normalization is appropriate. Yet the *relative* size and position of characters are vital for discriminating between some characters (such as "C" and "c" or "9" and "g"). The simple normalization I have chosen excludes any contextual information and exacerbates certain confusions.

All that remained was the selection of handwriting representations to consider. My choices were governed partly by what could be found in the literature. I generally eschewed techniques which have proved deficient for speech recognition: those using rule-based features incorporating hand-tuned thresholds. These representations are not only difficult and time consuming to construct, they can also be quite fragile. The better approach is to provide the raw variables to a classifier and allow the system to learn appropriate decision criteria.

I divided the field of handwriting representation into two broad categories. Some representations, such as images of the handwriting, discard the pen's movement. These *static* representations are equally applicable to off-line handwriting classification. Other representations encode information, such as the velocity of the pen, available only from on-line data. Such *dynamic* representations can incorporate pen motion directly, or indirectly by maintaining the order of static features extracted along the pen trajectory. Recall that determining the efficacy of representations requires attributing performance differences to individual factors. Since a goal of this thesis is investigating the usefulness of the pen's trajectory, I paid particular attention to contrasting representations which differed only in their use of dynamic information. For this reason I considered *hybrid* representations, a special case of dynamic representations that encode pen movement in an otherwise static image.

The statistical classifier used in my experiments required that each handwritten token be represented as a bounded-length feature vector. Distilling each character's data into such a feature vector is a key problem in character classification. For a given type of representation, too few dimensions will not provide enough information to permit accurate discrimination, while too many will exceed the information content of the original data. A larger number of dimensions also increases the number of parameters to be estimated in training the classifier. Thus, a longer feature vector need not yield superior accuracy. The degree of detail extracted by a representation typically can be varied. It was important to explore a range of parameters for each representation lest promising representations be passed over.

3.2.4 Summary

Building a handwriting classifier requires many decisions which are mutually dependent. Due to the large realm of possibilities, a truly optimal system is rarely guaranteed. Typically decisions are made sequentially, securing at best a locally optimal solution. I chose to investigate a broad array of potential representations rather than inch toward an elusive performance goal.

3.3 Static Representations

The first class of representations investigated were static in the sense that they did not incorporate pen motion. Instead, they were based on solely the image of a character. These images could be manipulated before constructing a feature vector for classification.

Creating a bilevel image, or bitmap, from on-line handwriting data is fairly simple. Within each stroke, iterate through pairs of sequential pen samples. The line connecting these points can be imaged using Bresenham's scan conversion algorithm [16], taking care to ensure that isolated points contribute a single pixel. A more complex alternative would connect the strokes' points using some type of curve fitting procedure, but I found this had little if any impact on the representations I examined.

3.3.1 Pixelated Images

The simplest manipulation one might perform on an image is *no* manipulation; the image may be classified directly. In some sense this is already proven technology since it is the form of handwriting we read. However, this does not imply that images are necessarily the best representation in any sense.

Because every character in the corpus was normalized to a unit bounding box, a square pixel array was used to hold each image. The single parameter for this representation is the resolution of the array. For purposes of comparison with certain other representations it was advantageous to use powers of 2 for the resolution. The smallest bitmap I considered was 4 pixels on a side since it was at the limit of legibility; the largest was 16×16 due to implementation restrictions of the classifier[2]. An intermediate 8×8 bitmap was also evaluated. Examples of these are shown in Figure 3.3.

Converting a bitmap to a feature vector for classification was simply a matter of placing the pixels in an arbitrary but consistent order. Numeric values of 0 and 1 corresponded to white and black pixels respectively. Having computed the necessary feature vectors, the classifier may be trained and tested. The raw classification results consist of a vector of scores, one value per label, for each token in the test set.

Result Analysis

There are many ways to process classification results so that we may assess the system's performance. Since these are the first results presented, I have taken the opportunity to describe some of the possibilities.

Perhaps the most common performance metric is to compute *top-choice accuracy* by comparing each token's highest-ranked classifier label with its transcription label. The accuracy for each of the bitmap representations is shown in Figure 3.4. Of the possibilities tried, the 8×8 resolution yielded the best accuracy. The lower performance of the 4×4 resolution is probably due to its being too coarse to adequately differentiate between some characters. Similarly, the high dimensionality of

[2]For example, as the image resolution increases adjacent pixels tend to be more strongly correlated. This condition was poorly handled by the algorithms chosen for principal components analysis.

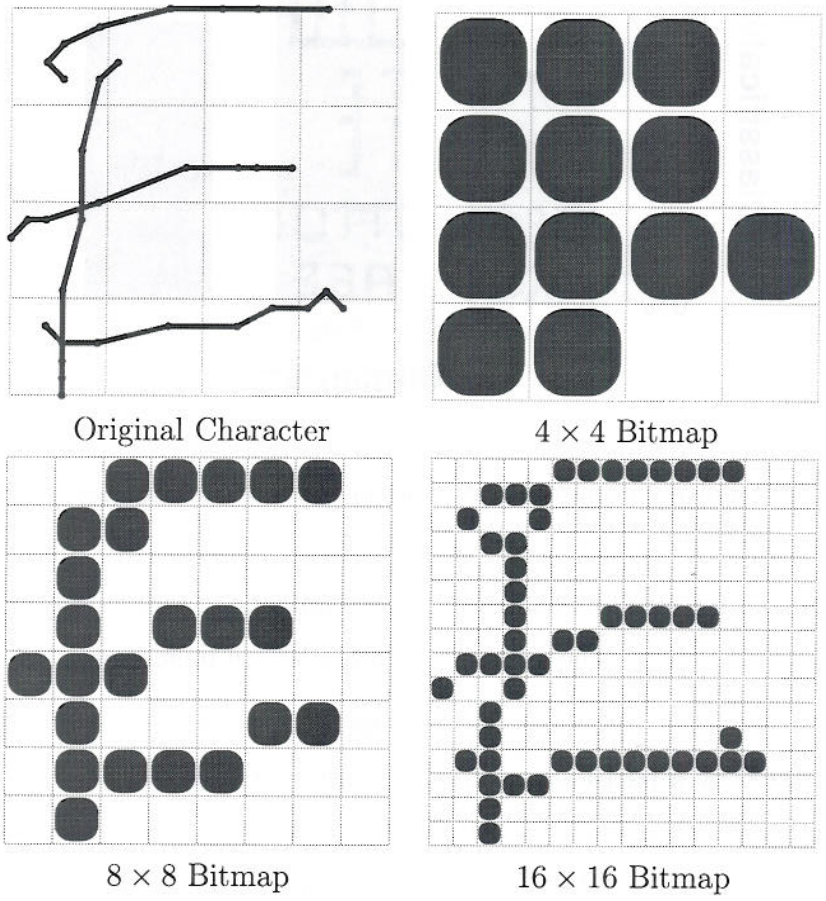


Figure 3.3: A character and its bitmap image representations.

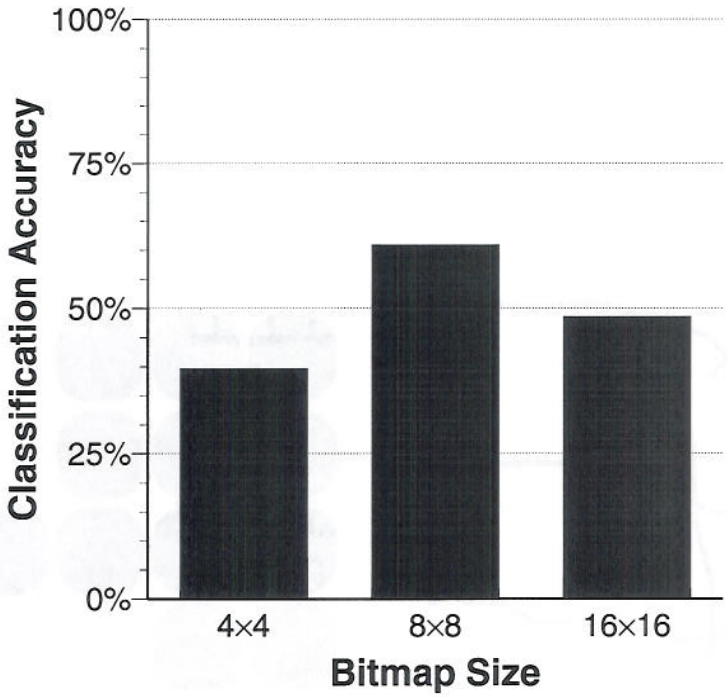


Figure 3.4: Top-choice character classification accuracy for bitmap representations.

the 16×16 resolution probably hinders robust modeling. One might employ significance testing [24] to ensure the accuracy differences are meaningful.

Top-choice accuracy measures the *average* system performance. However, the accuracy obtained by a particular writer depends on the degree to which their handwriting matches the classifier's prototypes. Accuracy on a particular subject's data can vary greatly, as shown in Figure 3.5. Among the testing subjects, accuracy ranged from 31.9% to 77.3%. Since deployed systems must serve a large fraction of the population, it may be desirable to evaluate approaches based on the lowest accuracy among some portion of the testing subjects. Thus one might say this representation achieved a median accuracy of 61.6%, but only 50.3% or better accuracy on 80% of the testing subjects. Note that approximately two-thirds of the testing subjects were associated with accuracies above the lowest-scoring training subject. Excluding data from such outlier training subjects may result in improved system performance.

Entropy can serve as a more comprehensive performance metric [47] because it

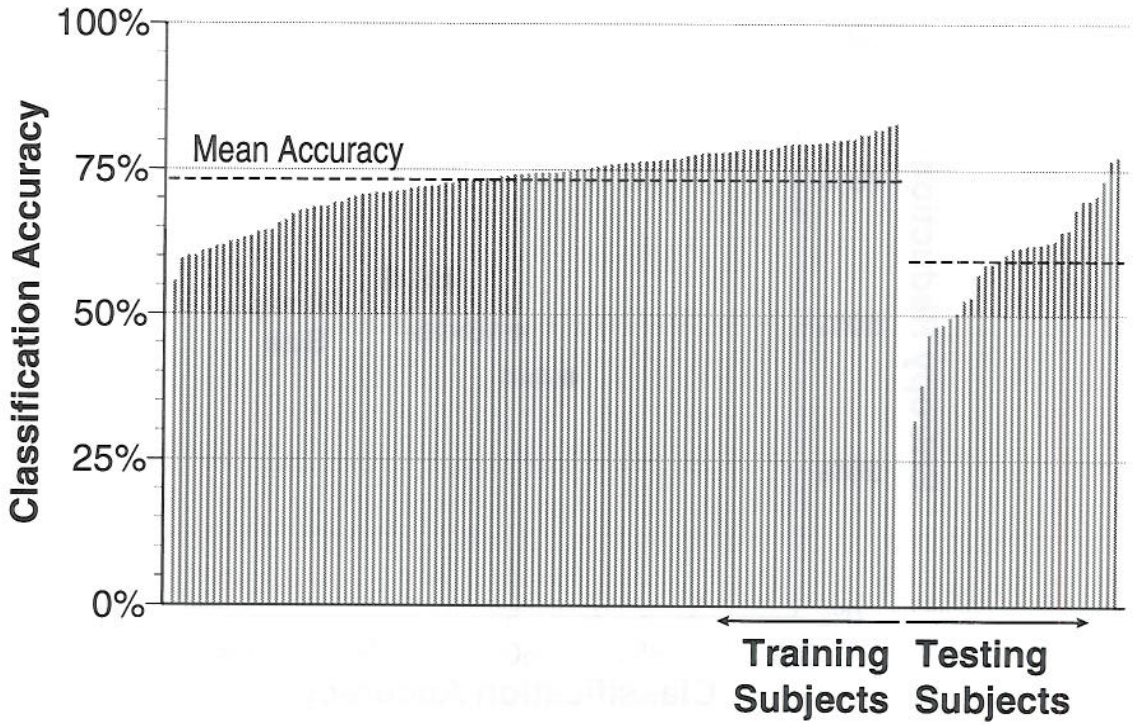


Figure 3.5: Top-choice accuracy by subject for the 8×8 bitmap representation.

accounts for the distribution of errors. For example, a character classifier which errs only in letter case may be preferable to an equally accurate system producing errors uniformly distributed over all symbols. To create a measure comparable with accuracy (ranging from a paltry 0% to a perfect 100%), I compute *entropy reduction*:

$$\frac{I(X;Y)}{H(X)}$$

where $I(X;Y)$ is the average mutual information [22] between correct and classifier labels and $H(X)$ is the entropy of the correct labels. Entropy reduction for the bitmap representations is plotted against top-choice accuracy in Figure 3.6. The correlation between the two measures is typical except for degenerate representations. Despite its theoretical advantages, in practice entropy is rarely more informative than accuracy.

Distilling classification results to a single number is noble, but it does not reveal the nature of symbol substitutions. This can be achieved using a *confusion matrix*.

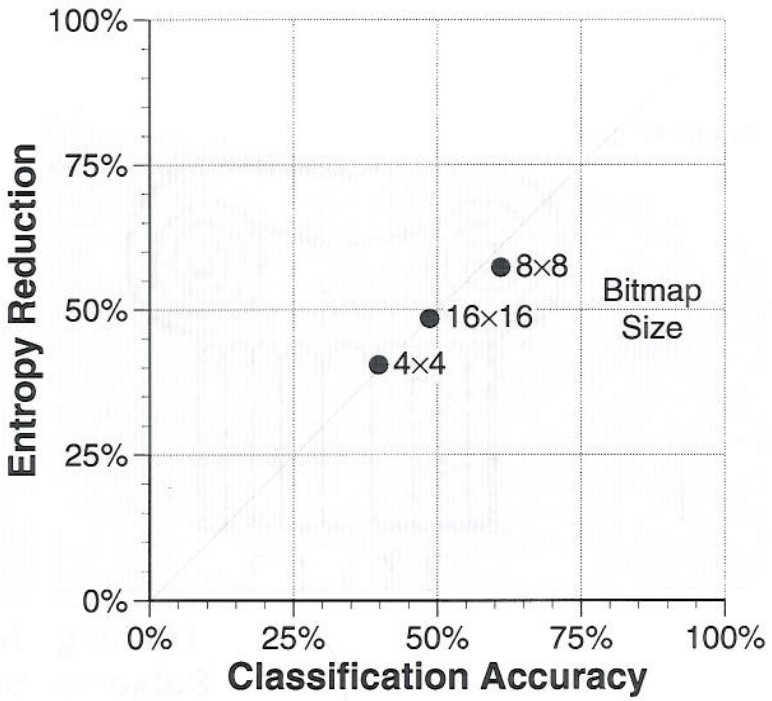


Figure 3.6: Comparing accuracy with entropy reduction for bitmap representations.

Each cell in the matrix represents an accumulator, indexed by the correct and classifier labels of each token. Top-choice accuracy can be computed from values along the diagonal. The wide dynamic range of bubble charts makes them particularly well-suited for presenting confusion matrices. Because of the wide disparity in label frequencies for handwriting, I chose to normalize the data within each transcription label. Figure 3.7 shows such a display for the 8×8 bitmap representation. I included guides to make the areas for upper-case letters, lower-case letters, and digits more distinct.

The errors made by the classifier are far from uniformly distributed. As might be expected, errors are biased toward more frequently occurring characters. Structure resulting from case substitution can be seen as two off-diagonal lines of confusions. Other significant sources of error are characters confusable because of their similar shapes, including "1" with "l," "0" with "o," "q" with "9," "S" with "5," and "v" with "u."

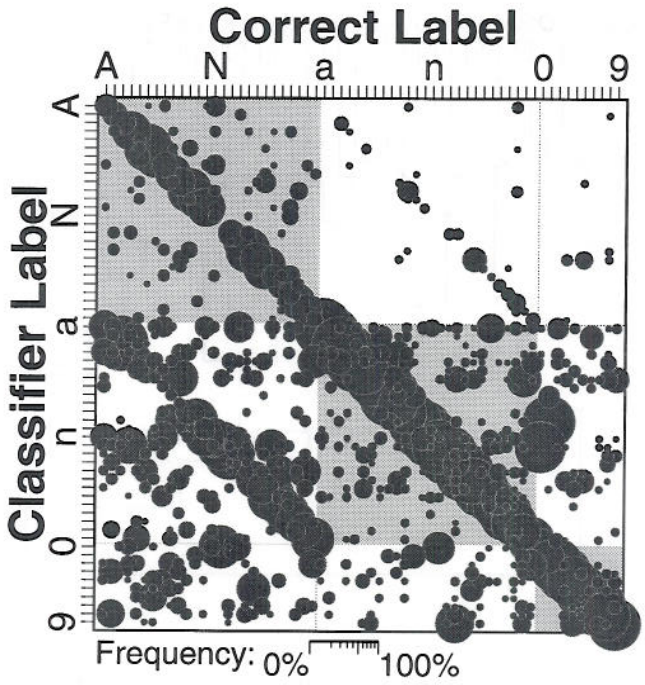


Figure 3.7: Character classification confusions for the 8×8 bitmap representation.

The similarity between characters, as defined by a particular representation, can be made apparent by restructuring the confusion matrix through an agglomerative clustering procedure [14]. Clusters are initialized so that they each contain a single character. The two most similar clusters are merged, iterating until only a single cluster remains. Both the cluster similarity metric and the merging procedure may be varied. I chose a simple merging procedure: the corresponding rows and columns in the confusion matrix are added to represent the conjoined confusions. For cluster similarity I measure the mutual information between all correct and classified token labels. The most similar pair of clusters retain the greatest amount of information when merged. A character dendrogram constructed in this manner is shown in Figure 3.8. This display shows both the clusters formed and their relative strengths. The earlier clusters generally make sense from a pictoral standpoint, perhaps best illustrated by the structuring of "I," "i," "L," "l," and "1" at the far right of the diagram.

All of these evaluation methods have examined only the top-choice classifier label.

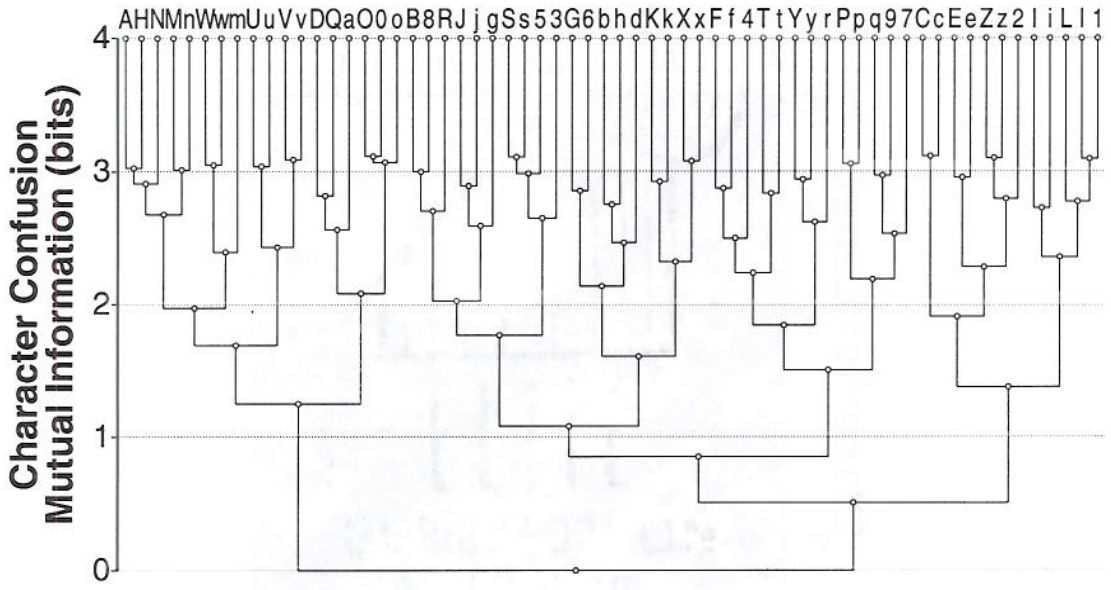


Figure 3.8: Clusters based on mutual information over 8×8 bitmap classifier confusions.

An alternate approach considers the remaining candidates by computing the mean depth of the correct label in the classifier result vectors. Again, reducing the performance to a single number has its place but obscures system behavior. Instead, one may plot the *cumulative accuracy* of observing the correct label in the top-n classifier labels. Such a display is shown in Figure 3.9. The first data point corresponds to top-choice performance. The last always corresponds to 100% accuracy, since the correct answer must be in the response vector. A steeply sloping curve rapidly reaching the final value indicates a better classifier.

While all these methods have been used in various stages of my research, I will depend primarily on classification accuracy in comparing representations. It is simple to compute, succinct, and meaningful. Also, it allows for a comparison between classification and recognition systems.

Pixmaps

In preparing bitmaps for classification the spatial relationship between pixels was not preserved. Bitmaps which are visually similar can have very dissimilar feature

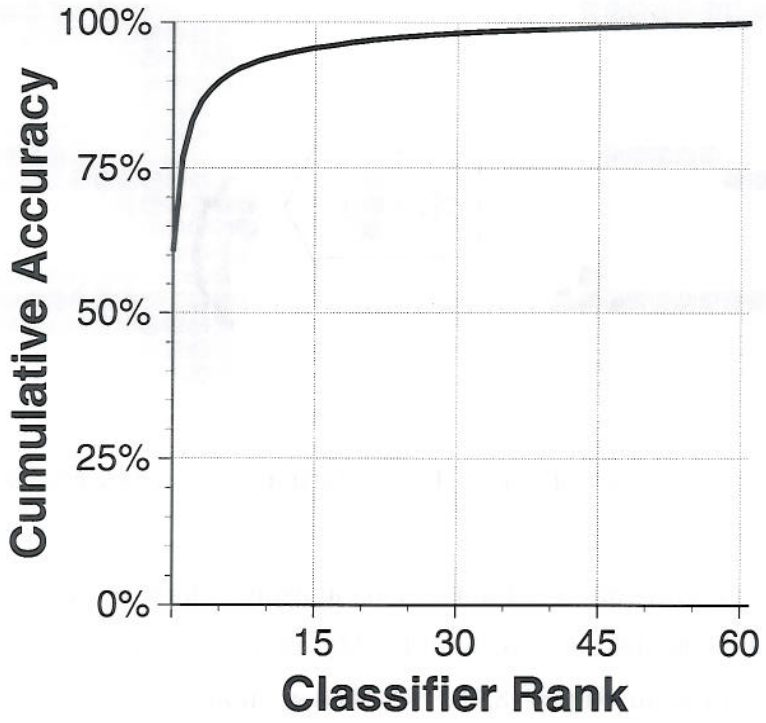


Figure 3.9: Cumulative accuracy for the 8×8 bitmap representation.

vectors. Small variations in pen coordinates can produce these differences due to quantization with hard thresholds. The principal components analysis should recover the correlation between pixels, but explicitly addressing this problem could potentially yield higher accuracy.

One approach is smoothing the bitmap so that a pixel's value is distributed across its neighbors. This can be accomplished by convolving a blurring kernel with the image to produce a pixmap (in which pixels can take on arbitrary floating-point values). An example of this processing is shown in Figure 3.10. The blurring kernel was specified to have unit volume, but its precise shape was arbitrary and in fact should be optimized. Blurring does not introduce additional information to the representation. The key to its potential success is purely in making the feature vector distance better reflect character distances.

An alternate was to soften the thresholds used in setting pixels to produce an anti-aliased image [17]. Because of spatial quantization, many sets of endpoints can

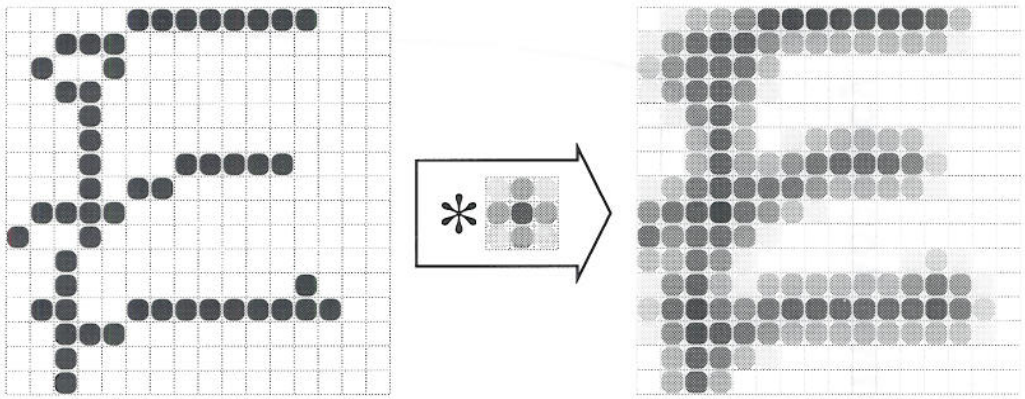


Figure 3.10: Convolving a 16×16 bitmap with a blurring kernel.

produce identical scan-converted lines. This aliasing can be reduced by setting pixel values based on their distance to the line. My algorithm for creating such character images is illustrated in Figure 3.11. A bitmap image is produced at double the requested resolution and blurred as described above. The final pixmap is constructed by averaging higher-resolution pixel values corresponding to each lower-resolution pixel. This type of image *does* introduce additional information, in effect encoding higher resolution as gray levels.

I examined both of these representations at the three resolutions used for bitmaps. The results are shown in Figure 3.12. Blurring the images did not affect the accuracy compared to that of the original bitmaps. The anti-aliased images showed gains at lower resolutions. Note that the 8×8 anti-aliased image performed substantially better than the 16×16 bitmap on which it is based.

Summary

The first class of representations I considered were images because of their natural relationship to writing. An 8×8 bitmap yielded an accuracy of 60.7%. A number of related performance measures were demonstrated. The accuracy of image classification could be increased to 63.8% by using an anti-aliasing algorithm.

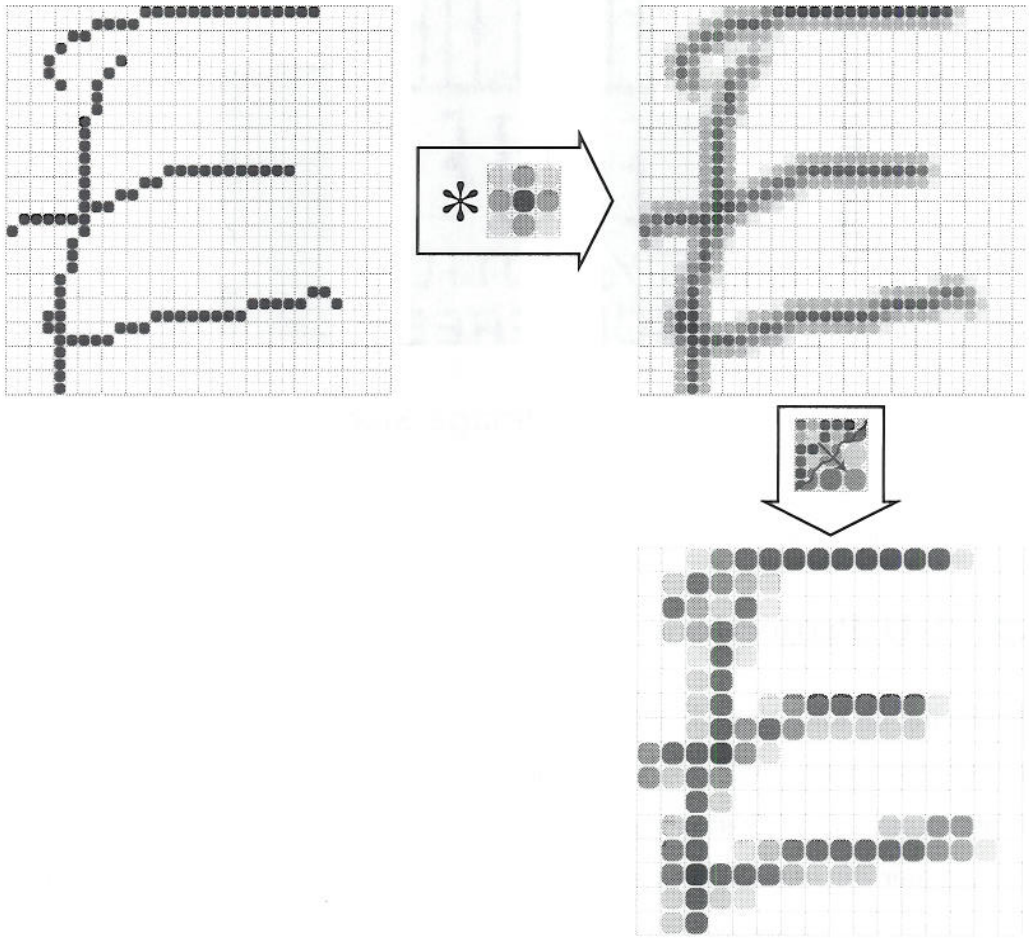


Figure 3.11: Constructing a 16×16 anti-aliased image from a blurred, higher-resolution bitmap.

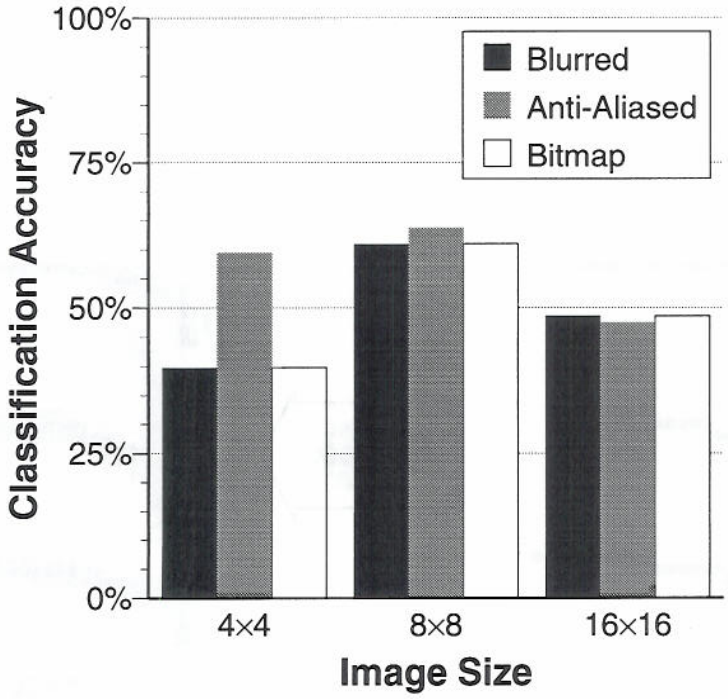


Figure 3.12: Top-choice character classification accuracy for pixmap representations.

3.3.2 1-D Projections

Producing an anti-aliased character may be viewed as projecting an image onto a lower resolution space, reducing the feature vector size by increasing the quantization levels per pixel. Since this operation improved classifier accuracy for low-resolution images, further dimensionality reduction might result in addition performance gains.

The notion of projection can be extended to producing 1-dimensional results. For example, each element in an accumulator array may sum pixel values from a single column in an image. An orthogonal projection could be produced by summing pixels from each row. A desirable property of these projections is that they can provide complementary representations of the character. Two shapes which are confusable when projected along one axis may be resolvable when projected along another. Projections along arbitrary axes may be produced by rotating the character before it is scan converted. Thus, this type of representation has two controlling parameters: the number of pixels in the accumulator array and the axis along which the image is

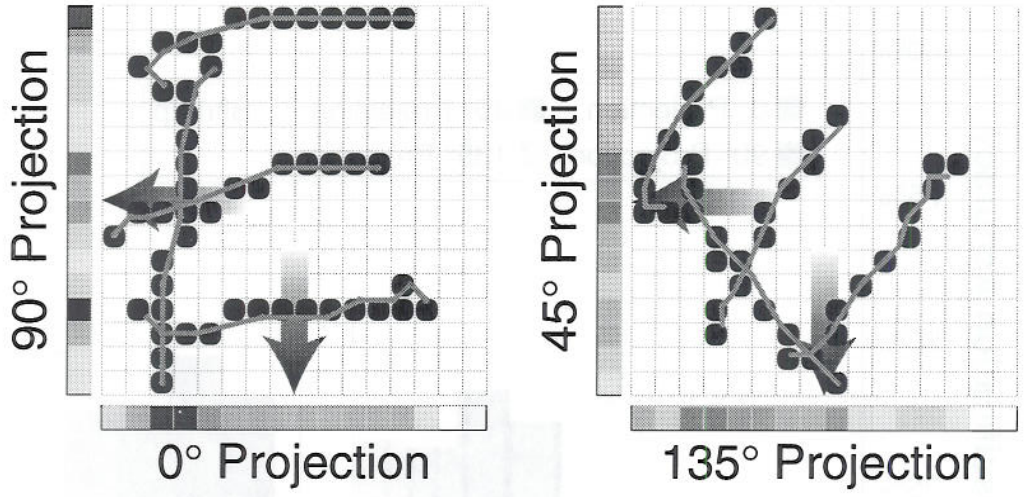


Figure 3.13: Constructing 16-pixel, 1-dimensional projections of a character from its bitmap images. Pixels are summed within each row or column.

projected.

For my experiments I considered four projection axes inclined at 0°, 90°, 45°, and 135°. Examples of these are shown in Figure 3.13. The first two projections are based on the same bitmap images already investigated and so allow for a meaningful comparison with them. The other two projections evenly divide the range of directions. These choices are somewhat arbitrary. Were this representation adopted for classification, the optimum angles should be identified.

In the baseline experiment on this representation, these four projections were considered individually. Each projection's accumulator array was used as a feature vector. As with the image representations, 3 resolutions are considered. The results of these experiments are shown in Figure 3.14. The 90° projection consistently yielded the best accuracy. However, even this representation performed worse than the bitmap from which it is derived.

Combining Projections

The complementary nature of these projects suggested constructing a feature vector by adjoining them. The performance of three cases, two combining orthogonal projections and one combining all considered projections, is shown in Figure 3.15.

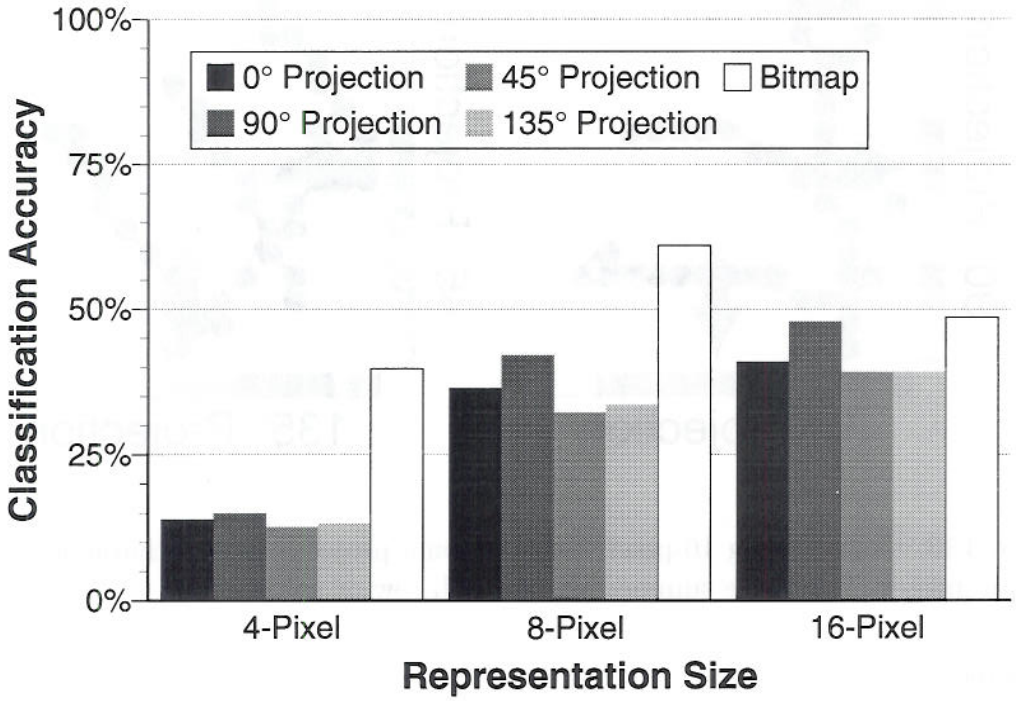


Figure 3.14: Top-choice character classification accuracy for projected bitmap representations.

The representation size reflects the image used for each projection. Thus, a 4-pixel representation containing two projections resulted in an 8-dimension feature vector. All of these combinations performed better than the individual projections from which they were composed. Representing characters using all four projections consistently worked better than an equivalently sized bitmap.

Two comments are in order. First, this experiment showed that complementary sources of information could be used to increase the accuracy of a character classifier. However, the marginal utility of additional representations diminished due to the limited information of the source. The accuracy for a combination of the two worst performing projections, 4 pixels at each of $45° + 135°$, exceeded the sum of its component's accuracies. Such substantial gains were not realized for combinations of the better projections, such as 8-pixel projections in the same directions. As more components were added to the feature vector, the accuracy of the system eventually decreased due to the additional parameters that had to be estimated in training the

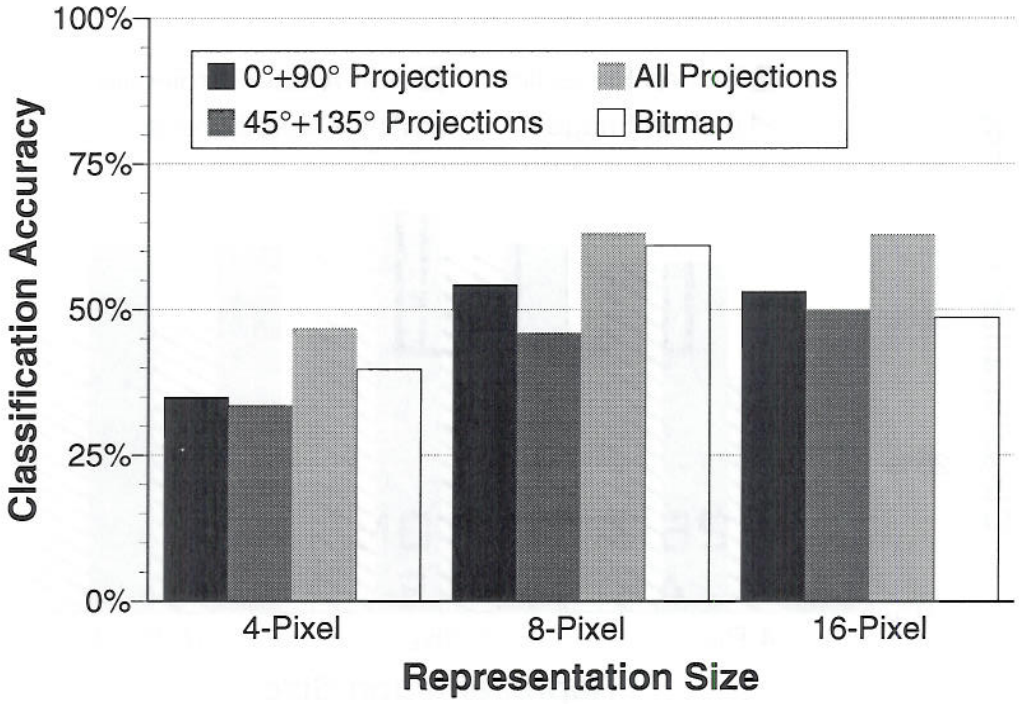


Figure 3.15: Top-choice character classification accuracy for combinations of projected bitmaps.

classifier. Second, this experiment showed how the organization of a representation could affect classification performance. The combination of 16-pixel projections at $0° + 90°$ did not contain sufficient information for an unambiguous reconstruction of the source image. Despite this, it yielded superior performance compared to the more informative bitmap. In addition to the information available to the classifier, we must pay attention to how that information is presented.

Pixmap Projections

I have shown that an anti-aliased image can yield higher classification accuracies than its corresponding bitmap. Would projections based on these grayscale images similarly perform better? For this experiment I examined only the best projection method observed: the combination of all four projections considered. This projection was computed for both the simple bitmap and its anti-aliased cousin. The results of this experiment are shown in Figure 3.16. In all cases the projected representation

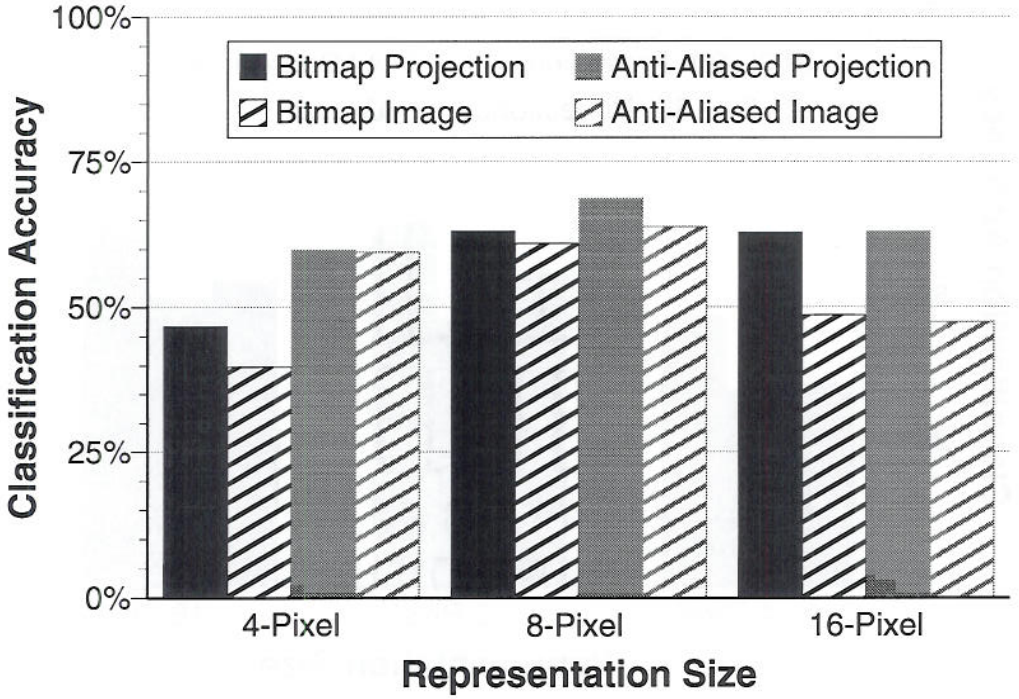


Figure 3.16: Comparing the top-choice character classification accuracies of 4-way projected images derived from bitmapped and anti-aliased sources.

outperformed the images. Additionally, the projection of anti-aliased images consistently outperformed the bitmap-based projections, perhaps because they incorporate higher resolution information.

Summary

I have shown how to combine rotation, scan conversion, and accumulation to project a character along any axis. By projecting along multiple axes, the resulting one-dimensional images could be combined to form a representation with superior performance. In fact, this experiment has yielded the best classification results described thus far: the combination of 0°, 90°, 45°, and 135° projections of an 8×8 anti-aliased image provides a character classification accuracy of 68.8%.

3.3.3 Image Transforms

We have already seen that transforming a character's image to an alternate representation can improve classification accuracy. In this section I examine some of the many other transforms possible. My selections were motivated by representations that attempt to explicitly capture distinctive symbol characteristics.

Hough Transforms

Characters are formed from lines and curves. In character images these structures exist only as correlations between pixels. However, the image may be transformed to better reflect the underlying geometric construction.

The first such transform examined assumes the image is composed of only straight lines. In general the problem of detecting colinear points is difficult. The Hough transform [3], more completely examined by Leavers [44], reduces this problem to a simpler task of locating intersecting curves. Planar lines may be uniquely specified using two parameters. Hough's formulation was based on the slope-intercept parameterization of a line, but both of these parameters are unbounded. An alternative parameterization suggested by Duda and Hart [12] avoids this difficulty by representing the normal to a line. Under this formulation the equation for a line is

$$x \cos\theta + y \sin\theta = r$$

where θ is the angle of the normal and r is the distance from the origin to the line. An example of two lines running through a single point is shown in Figure 3.17.

To compute the Hough transform of an image, the θ-r parameter space is quantized and represented using a set of accumulators. Parameters are computed for all possible lines passing through each point in the image by stepping through values of θ. The indicated accumulators are then incremented. This transforms each image point into a curve in parameter space. A trivial example of this transform is shown in Figure 3.18. The image, shown on the left, contains only two pixels. These have been shaded differently to show the correspondence between each point and its contributions to the transform, on the right. Two "curves" can be seen, with the taller curve due to the point more distant from the origin. A line connecting the two image points is

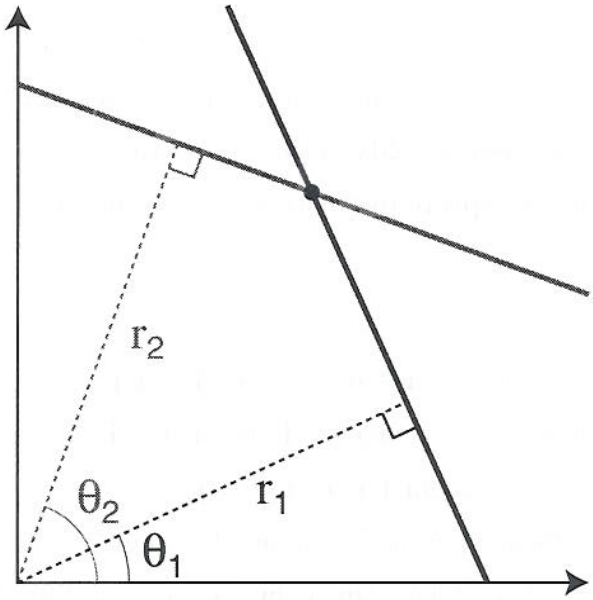


Figure 3.17: The normal parameterization of two lines through a single point.

represented by the intersection of the parameter space curves.

A careful inspection of Figure 3.18 reveals that the parameter space curves do *not* intersect in the sense that they have no common pixels. This problem stems from the procedure used to draw the transform: when the slope of the curve has magnitude greater than 1, stepping through angle values results in an inadequate sampling of radius. This may be remedied by connecting adjacent samples with a straight line, as shown in Figure 3.19. The utility of this connected Hough transform will be determined through classification experiments.

For my experiment I considered three transform sizes based on the bitmaps described earlier. At each size I compared plain and connected Hough transforms. The manner in which I applied the transform differs from common practice in two respects. First, to reduce the number of variations considered I restricted the resolution of the parameter space to match that of the image. These values are independent and could be optimized separately. Second, constructing the parameter space representation is typically followed by peak detection to identify the parameters of prominent lines. In general this is a difficult problem which introduces an additional source of error. Following the philosophy of avoiding hard decisions, I constructed the feature vector

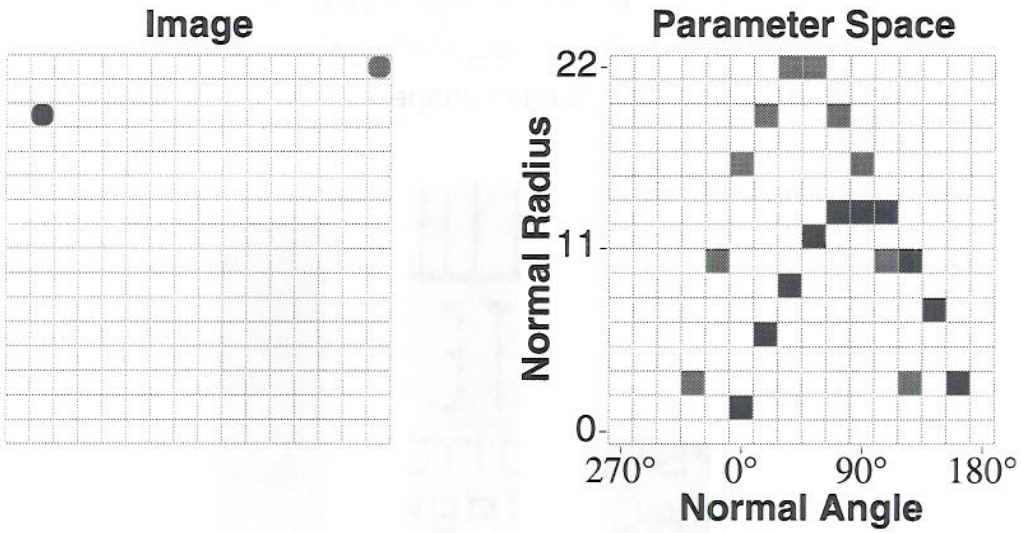


Figure 3.18: A simple bitmap image and its Hough transform.

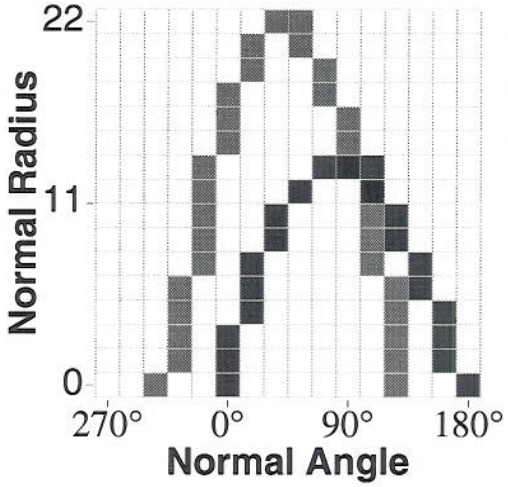


Figure 3.19: Connecting points in a Hough transform to better represent intersections.

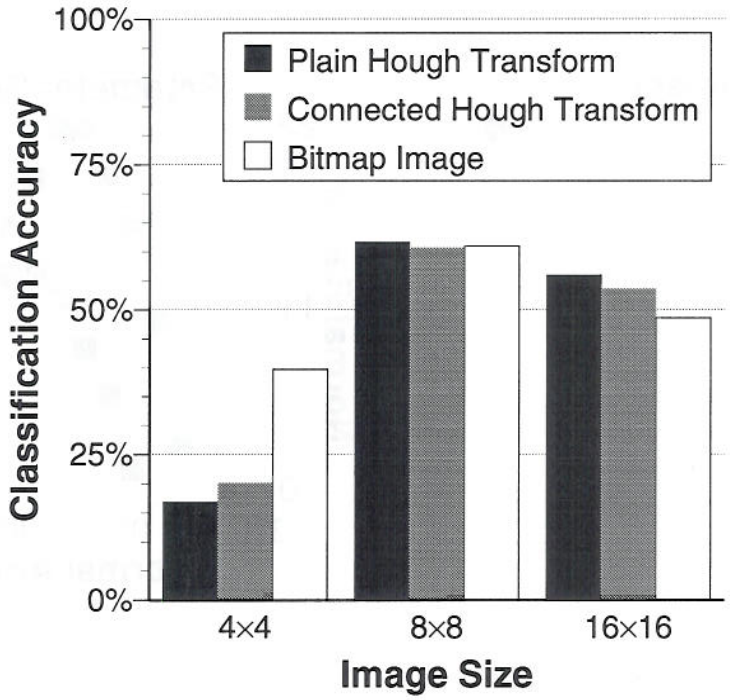


Figure 3.20: Top-choice character classification accuracy for plain and connected Hough transforms.

using the entire transform. Note that parts of the parameter space are unreachable and are excluded in assembling the feature vector. Retaining the whole transform has a secondary benefit when the data includes shapes other than lines. Curves, approximated by many straight lines, will generate a characteristic transform in parameter space which is not a single point. By avoiding peak selection these diffuse shapes can be detected. The results of this experiment are shown in Figure 3.20. Connecting points was beneficial only for the smallest transform. The best case was only slightly more accurate than the image from which it was constructed.

How well do Hough transforms represent the character data? One way to evaluate this is to reconstruct an image from its transform. Provided the correct technique is used for reconstruction, this demonstrates the information preserved in the transform. The results can be analyzed objectively, by comparing the original image to its reconstruction with some distortion measure, but even a subjective visual inspection can be informative. Performing an inverse Hough transform is relatively straightfor-

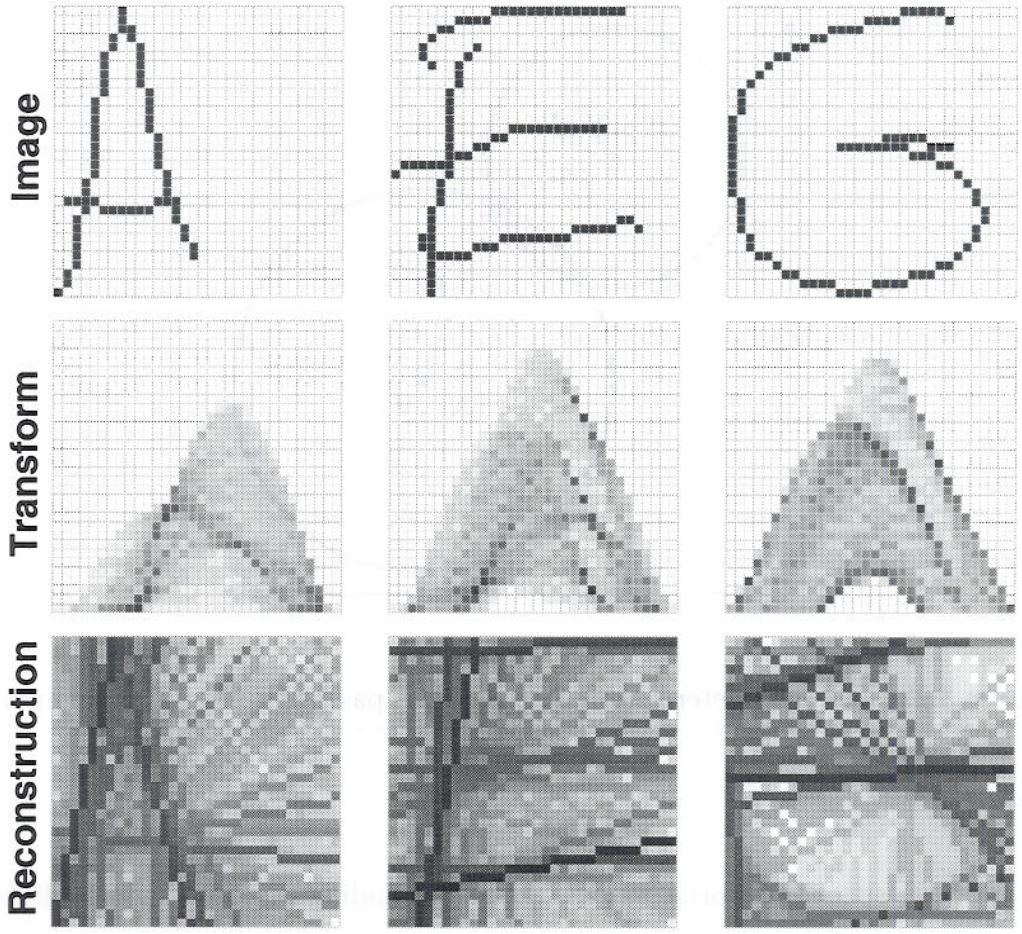


Figure 3.21: Images, their Hough transforms, and reconstructions.

ward. Each point in the parameter space corresponds to a line in the image at a particular intensity. To reconstruct the image, these lines are drawn retaining the strongest intensity for each pixel. Elements of the original images can be seen in the reconstructions of Figure 3.21, but the results are far from perfect. Straight lines are better represented than curves. However, the endpoints of each line are not retained. Depending on the shapes to be classified, this could be a desirable property or disastrous.

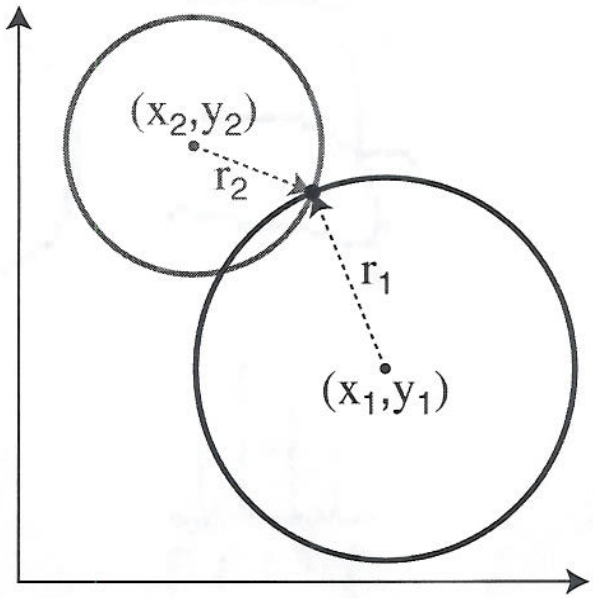


Figure 3.22: The parameterization of two circles passing through a single point.

Circle Transform

Parameter space transforms may be applied to finding shapes other than lines. As described by Kierkegaard [38], I have examined representing character images using a circular arc parameterization. Circles in a plane may be uniquely specified using three parameters, two coordinates for its center and one for its radius. An example of two circles passing through a particular point is shown in Figure 3.22. At a given radius, an image point is transformed into a parameter-space circle; as the radius is varied, the transform of a point becomes a cone. The point at the intersection of three such cones corresponds to the image of a particular circle.

In order to keep the feature vector size commensurate with earlier experiments, I chose to retain only the center parameters from the transformation. Each center was valued at the maximum for all of its radii and constrained to appear within the bounds of the source image. Examples of this representation are shown in Figure 3.23. As might be expected, a strong center is identified within the more circular "G," but a characteristic transform is associated with other shapes as well. For example, a circular arc approximates a line as its radius approaches infinity. Within the limitations

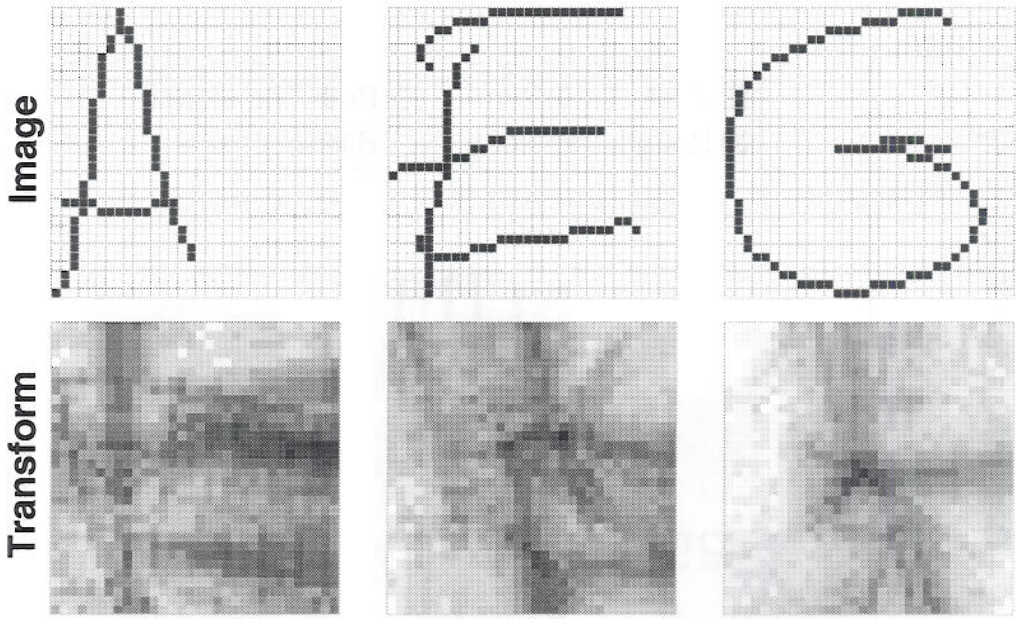


Figure 3.23: Examples of circle centers detected by a parameter-space transform.

of this particular implementation, lines at the edge of the image field can be better represented since they can obtain the largest radii.

As with other experiments, I considered three sizes of circle transforms. Detecting circle centers could potentially provide complementary information to that given by the Hough transform. Accordingly I also evaluated a combination of these two representations. The results are shown in Figure 3.24. The combination of circle and Hough transforms consistently outperformed the circle transform alone. However, for the 8×8 image the combination did not perform as well as the Hough transform alone. At that resolution the two transforms do not provide sufficient complementary information to overcome the burden of additional model dimensions.

Two-Point Transforms

The parameter space transforms described so far are based on individual points, with all potential parameterizations passing through each point considered. For typical images, the majority of these parameterizations will prove uncorroborated by other points. The resulting transforms are cluttered with many potentially insignifi-

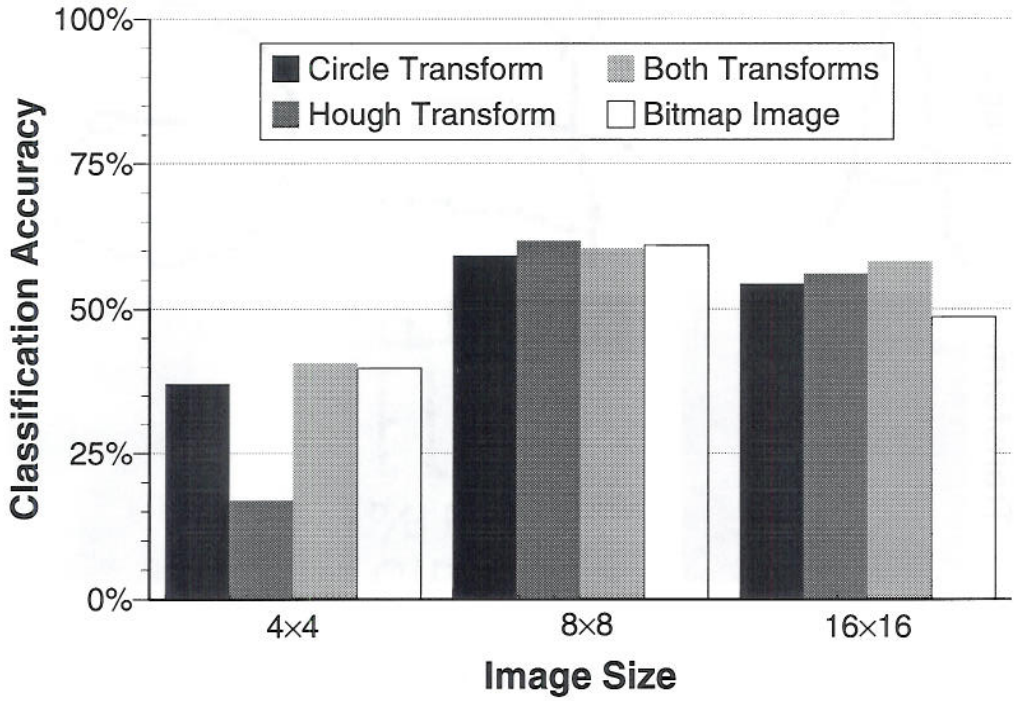


Figure 3.24: Top-choice character classification accuracy for circle parameter transforms.

cant components, possibly degrading classifier accuracy. This situation may be remedied by reducing the number of parameterizations considered for each point. A shape characterized by n parameters may be *uniquely* specified using n of its points. Thus, a transform may be pruned of impossible candidates by processing points in sufficiently large groups.

I have examined three parameter space transforms based on pairs of points. These are illustrated in Figure 3.25. The colinear transform is comparable to a Hough transform. Rather than consider all lines passing through a single point, it is based on the parameters of a single line passing through two points. This line is encoded using the angle and radius of the normal, just as before. The midpoint transform is comparable to the centers extracted from the circle transform. For each pair of points, the location of their midpoint is accumulated. The symmetry transform extends this notion from points of symmetry to lines of symmetry. The midpoint of two points lies on a potential line of symmetry. These lines can be identified by taking a Hough

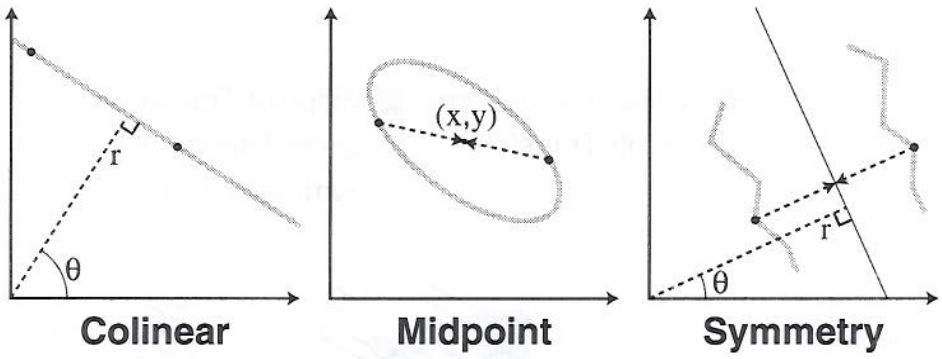


Figure 3.25: Three parameter space transforms based on pairs of points.

transform of all the midpoints.

Again I conducted classification experiments based on three sizes of bitmaps and present the results in Figure 3.26. In general the performance of these transforms was disappointing. However, these results suggest that making the entire transform available to the classifier was a prudent decision. It seems that transform components attributable to single image pixels help to characterize the shape.

Fourier Transform

All of the images treated so far have been in the spatial domain. A 2-Dimensional Discrete Fourier transform (2D-DFT) converts an image to the spatial frequency domain. In this domain, pixels may take on complex values. Unlike other transforms examined, a 2D-DFT preserves all information present in the image. This can be demonstrated by perfectly reconstructing the original image from its transform pair. The transform has two potential advantages over the original image. First, the basic shape of the image is contained in the lower frequencies while details reside in the high frequencies. This organization could assist the classifier by concentrating the more salient information. Second, the shape of graphical elements may be treated independently from their position. This is accomplished by encoding the transform's complex values as magnitude and phase.

I have only examined square images with 2^n pixels per side because this restriction simplifies construction of the 2D-DFT. The resulting transform is also square with 2^n

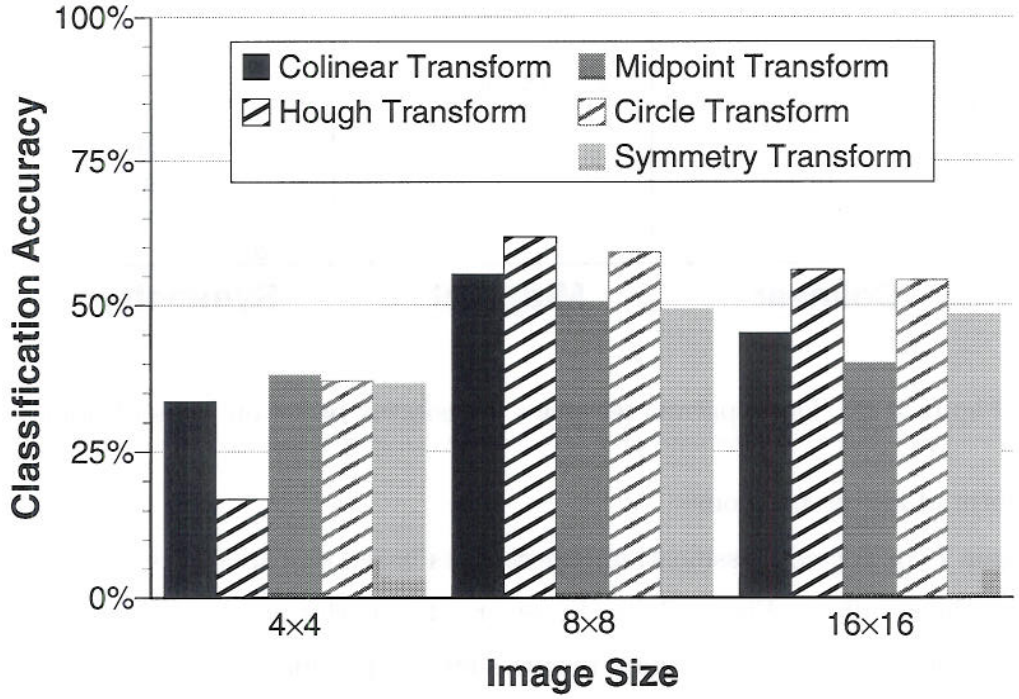


Figure 3.26: Top-choice character classification accuracy for two-point parameter transforms.

pixels per side. However, redundancy in the transform of real inputs permits half of this plane to be discarded. I compared feature vectors constructed from the magnitude and phase alone as well as the two combined. The results of this experiment are shown in Figure 3.27. It is interesting that the accuracy of the phase representation decreased as the transform size increased. I hypothesize that this is due to the higher resolution images permitting greater variability in the position of character elements. Since position is encoded by the phase, this translated into greater variability for the phase component. None of the 2D-DFT representations performed better than the original image, perhaps because both shapes and their positions are important in distinguishing between characters.

Summary

In this section I have examined a number of image transforms. They do not add information to the original image and in fact may add noise. Rather, their

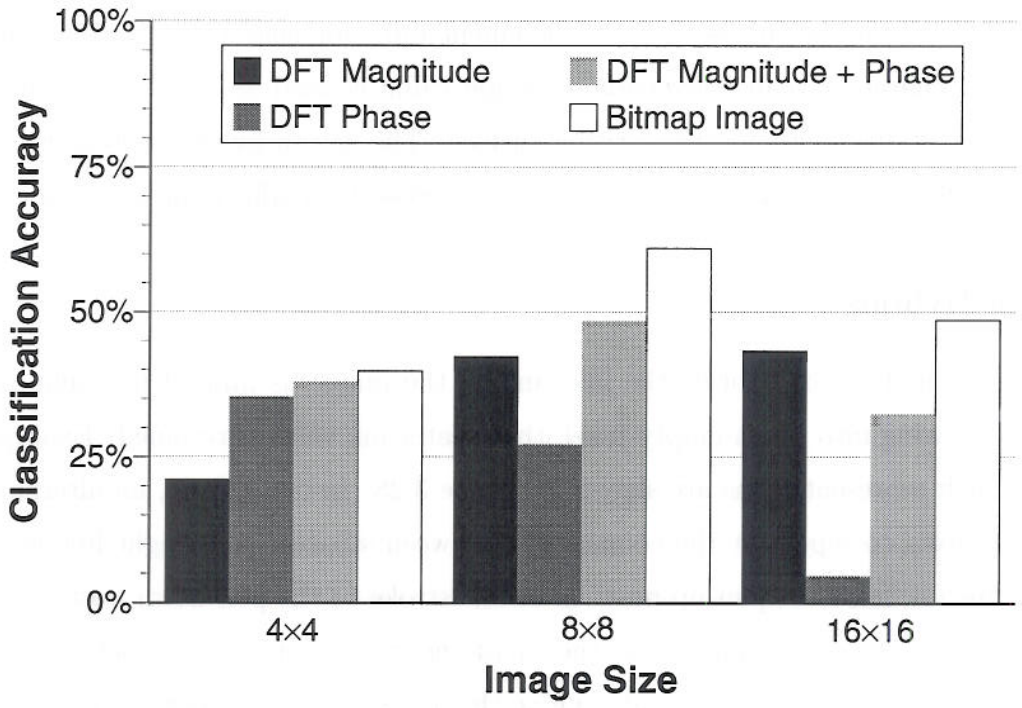


Figure 3.27: Top-choice character classification accuracy for two-dimensional discrete Fourier transforms.

potential power lies in restructuring the pixels to make distinguishing characteristics more apparent. Regardless of the theoretical benefits, my character classification experiments showed no compelling reason to adopt any of these representations.

3.4 Dynamic Representations

Unlike static representations, dynamic representations depend on the pen's trajectory. This results in additional variability which must be modeled, but also additional information which may aid distinguishing characters.

3.4.1 Hybrid Representations

I have relied on images as a proven and non-controversial representation of characters for classification. Indeed, experiments have proven its performance to be satisfactory compared to other static representations. In this section I suggest several ap-

proaches to incorporating dynamic information within images. These provide means of evaluating the usefulness of pen movement data for character classification. In standard character scan conversion, a single value is used as "ink" to fill-in pixels where the stylus contacted the writing surface. The key to hybrid representations is varying the ink throughout the character to represent an additional variable.

Scalar Hybrids

In the first set of hybrids the ink, and so the image, is limited to scalar values. When drawing into a non-empty pixel, the greater ink value is retained. Examples of three such representations are shown in Figure 3.28. In one hybrid, an alternate ink value is used to represent the connections between strokes. A straight line is drawn with this ink from the pen-up position of one stroke to the pen-down position of the following stroke. The remainder of the character is scan converted as before. A lesser value is chosen for the pen-up ink to give the actual writing priority. The other two hybrid images "color" the pixels of a bitmap. In one case, pixels within each stroke are assigned a unique identity. Later strokes overwrite earlier ones. This permits the number and order of strokes to be determined, but not the direction of each stroke. In the other case, the ink values express the time of writing (recall that this has been normalized within each character). The line drawn between two samples is valued at the average time for that segment. This representation allows the stroke direction to be deduced in addition to stroke order.

The results of this experiment, conducted on three image sizes, are shown in Figure 3.29. The dynamic information proved useful only for the smallest representation. In that case the normalized time ink provided considerable improvement, exceeding even the performance of an image with 4 times as many pixels. This suggests that coarsely quantized dynamic information is more useful. However, normalized time ink consistently yielded superior performance over the less detailed stroke order ink.

Vector Hybrids

In the second set of hybrids the ink takes on vector values. Vector addition is used to combine old pixels with new. Examples of such representations for instantaneous

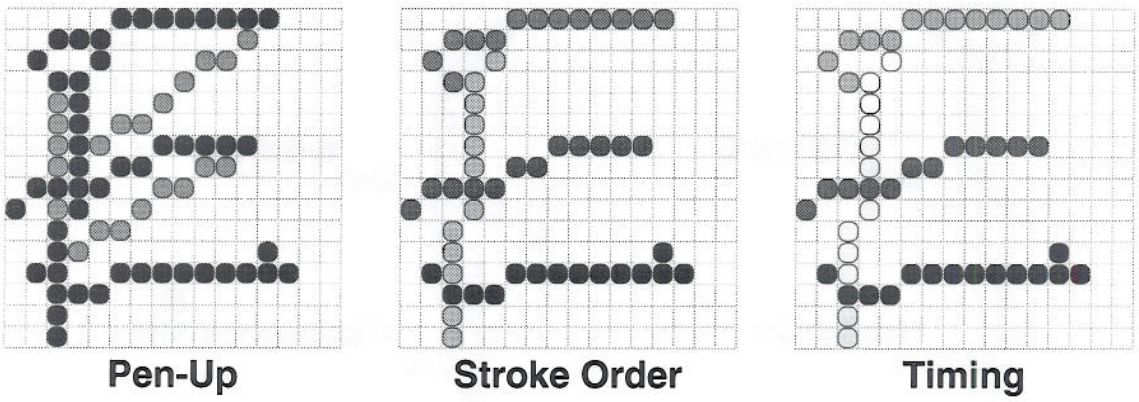


Figure 3.28: Representing scalar variables within hybrid images.

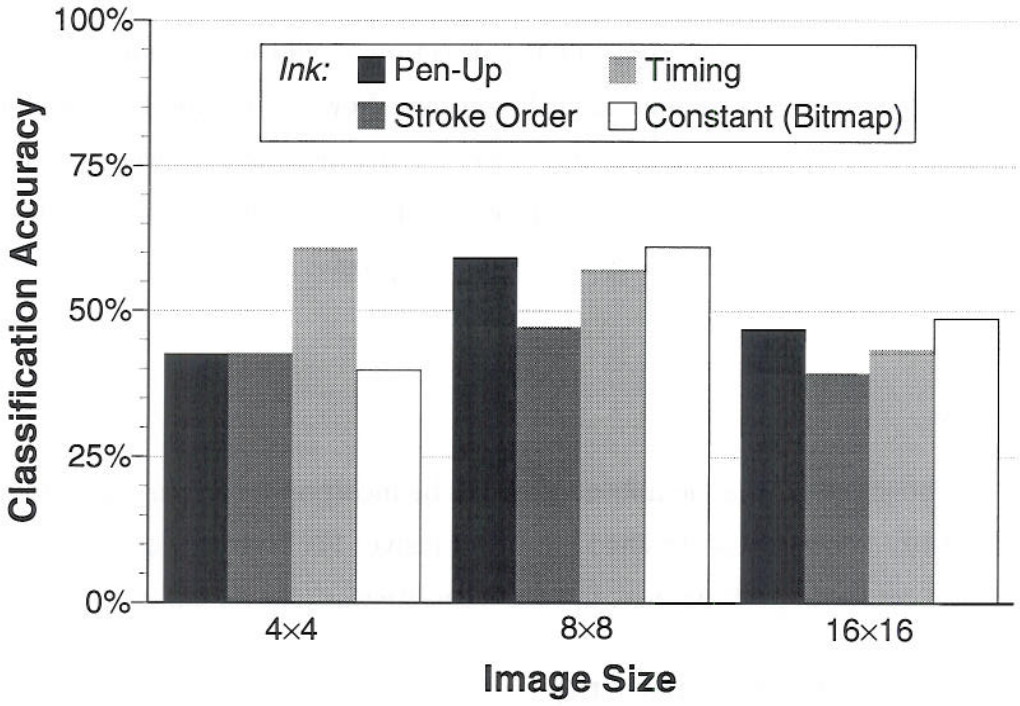


Figure 3.29: Top-choice character classification accuracy for scalar hybrid images.

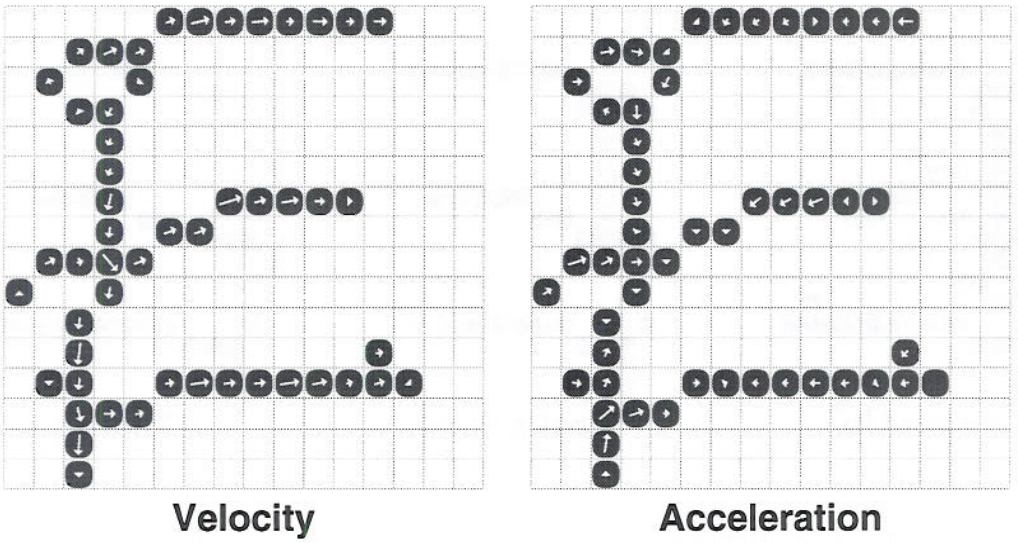


Figure 3.30: Representing vector variables within hybrid images.

velocity and acceleration, computed from first and second differences between points, are shown in Figure 3.30. The arrows indicate the direction and magnitude of the dynamic parameter for each pixel, but the direction should prove more useful. The results of using these representations at 3 sizes are shown in Figure 3.31. Not shown are the results for using the magnitude alone, for this did not fare well. In all cases using the direction of the vector alone proved better than including both direction and magnitude. Once again, dynamic information proved useful only for the smallest image.

Summary

I have shown how dynamic information can be incorporated in an image to determine its utility. The results obtained are inconclusive. For larger images the dynamic information degraded performance, but the opposite was true for the smallest images.

3.4.2 Trajectory Sampling

Dynamic character representations are based directly on the writing data samples without the spatial quantization and temporal aliasing associated with images. Knowing that pen position corresponds to ink in the image, one might select the

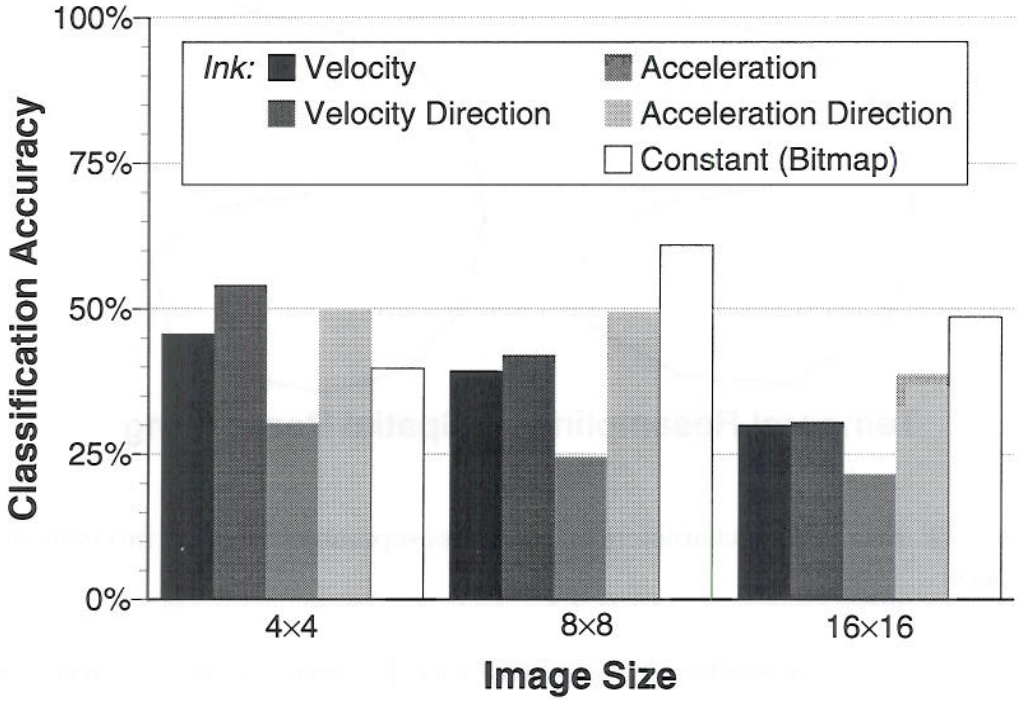


Figure 3.31: Top-choice character classification accuracy for vector hybrid images.

Cartesian coordinates of data samples as the simplest, least controversial dynamic representation. Since characters contain different numbers of samples, whereas feature vectors must all have the same length, some normalization procedure must be applied.

Uniform Resampling

Simply truncating and padding the feature vectors starting with the first sample assigns the final sample to varying dimensions, if it is included at all. A more desirable approach would produce similar feature vectors for isomorphic characters. Accordingly, I have chosen to resample each character to an identical number of data points along the pen's trajectory.

I have examined two methods for uniformly resampling each character. For samples evenly distributed in time, the total duration of inking is computed and divided by the number of samples desired. Note that this interval excludes time during which

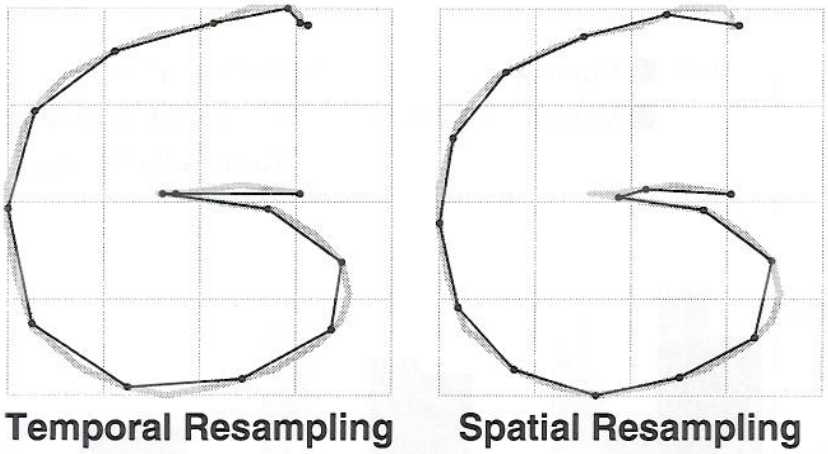


Figure 3.32: Resampling characters at 16 points separated by equal intervals of time and space.

the pen is raised. For samples evenly distributed in space, the pen's travel distance is summed and divided, again ignoring movement between strokes. In both cases the data is resampled at the appropriate interval using linear interpolation between original data samples. Examples of these techniques are shown in Figure 3.32. The difference between the two is most apparent at the top of the "G" when the pen is moving slowly.

The number of samples taken should be determined empirically. Too few samples cannot adequately capture the character shapes. Too many will provide only linear combinations of other samples. Hoping to cover the range of reasonable values, I tested representations based on 2, 4, 8, and 16 samples. With only two samples, each character's initial pen-down and final pen-up were chosen. Samples each supplied two values to the feature vector corresponding to the point's Cartesian coordinates. As a control I also considered randomly resampling the input. The writing was first resampled at equally spaced intervals to produce four times as many points as ultimately desired. An appropriate sized subset of these points, in temporal order, was chosen at random. Random approaches should not be ignored for they can succeed in unusual cases which confound the most carefully crafted heuristics. The results of this experiment are shown in Figure 3.33. Random sampling did not work as well as the uniform sampling techniques. Equally spaced sampling outperformed equally timed

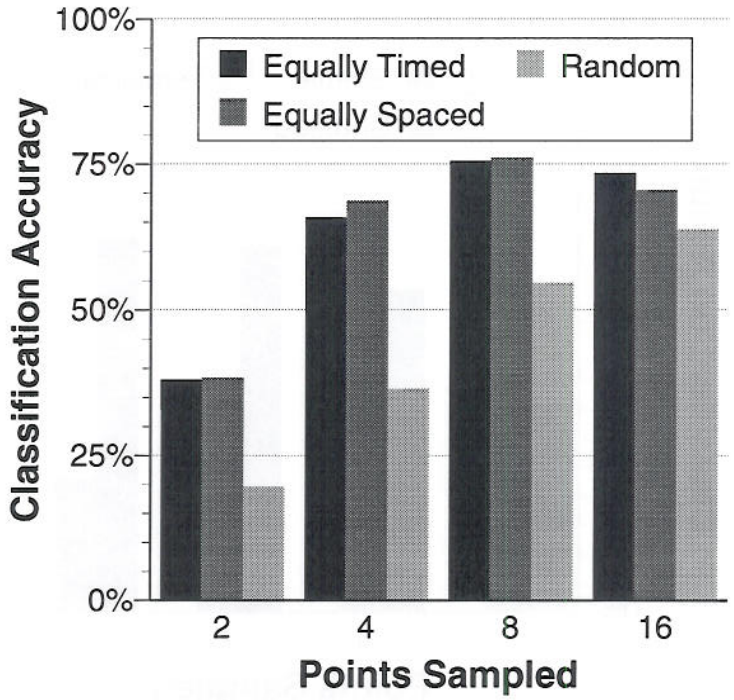


Figure 3.33: Top-choice character classification accuracy for Cartesian coordinates of uniformly and randomly resampled pen trajectories.

samples for three of the four sizes. The best case of eight equally spaced samples yielded a character classification accuracy of 76.1%. This is better than that achieved with an 8×8 bitmap while using the same number of dimensions as a 4×4 image.

Sample Reordering

In constructing the feature vector for classification, the resampled representations described maintain the order in which the ink was written. As seen in the allographic clustering study, otherwise similar characters can differ in the order and direction of their strokes. These differences can be eliminated by placing samples in an order independent of their timing.

I examined only the equally spaced resampling since its performance was superior. Characters were resampled at one of four resolutions already listed. These samples were sorted spatially by increasing coordinates, arbitrarily giving priority to the ordinate. The feature vector was then constructed as before. As a control, shuffling

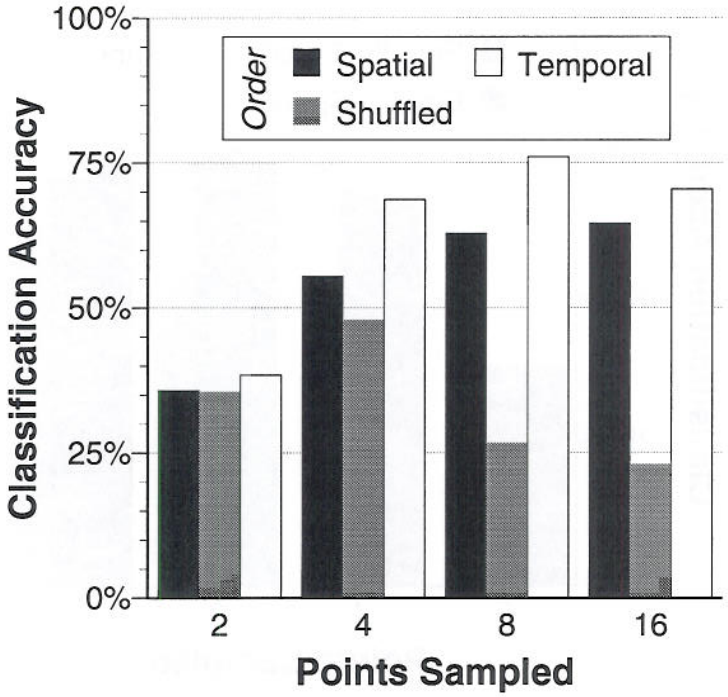


Figure 3.34: Top-choice character classification accuracy for Cartesian coordinates of equally spaced samples in several orderings.

the samples was also considered. Both of these representations were compared to the temporal-ordered samples. Because the feature vectors contain the very same data but in different orders, this provided another opportunity to evaluate the utility of dynamic information. The results of this experiment are shown in Figure 3.34. Spatial sample ordering was always inferior to temporal ordering, suggesting that the dynamic information was indeed beneficial for this representation. Most differences in stroke direction and order must be successfully assimilated by the mixture models. Shuffling the points was inferior to maintaining some canonical order.

Nonuniform Sampling

Resampling the pen trajectory treats all points within a symbol uniformly. However, some points may play a greater role in establishing a character's shape than others. If a property can be defined to be coincident with these points, the pen trajectory can be sampled unevenly to favor them. For this selection to be robust, the

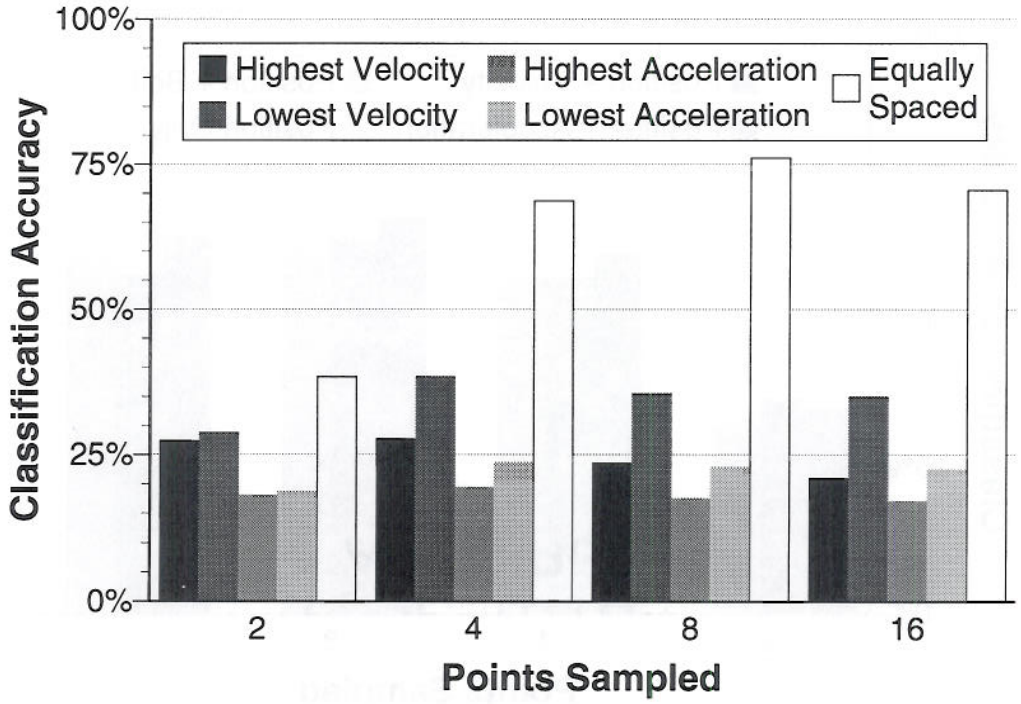


Figure 3.35: Top-choice character classification accuracy for Cartesian coordinates of nonuniformly resampled pen trajectories.

property should be defined in relative terms rather than absolute thresholds. One possibility is to sample points based on their instantaneous velocity or acceleration.

To compute these representations I began by equally-spaced resampling the characters at 4 times the desired resolution. The samples were placed in ascending or descending order according to the magnitude of their velocity or acceleration. The feature vector was constructed using only the first quarter of the points. The results of this experiment are shown in Figure 3.35. None of these techniques were superior to equally spaced sampling. Sampling based on velocity consistently outperformed sampling based on acceleration. The best of the four representations favored points with low velocity, often associated with the beginning and end of strokes as well as changes in directions.

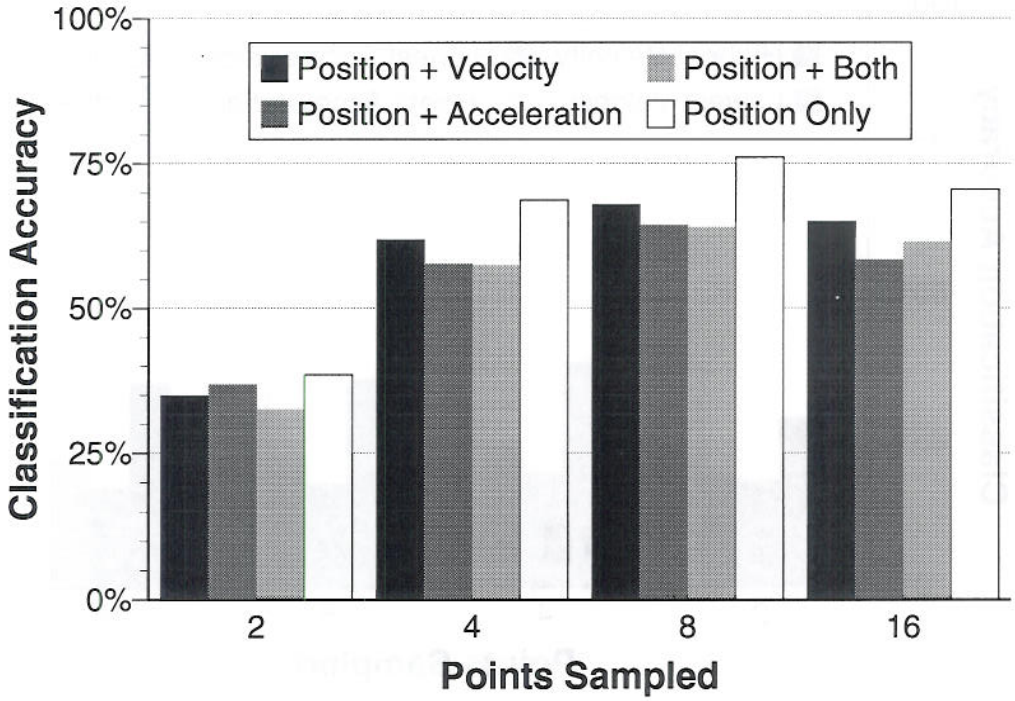


Figure 3.36: Top-choice character classification accuracy for equally spaced samples encoded as Cartesian coordinates with other properties.

Alternate Encodings

Thus far I have constructed feature vectors from the resampled characters using each point's Cartesian coordinates. Many other properties can form the feature vector instead of, or in addition to, position. I considered including velocity and acceleration, alone or in combination, with the coordinates of each sample. Both the magnitude and direction of these properties were used. The results from this experiment are shown in Figure 3.36. Incorporating the additional information proved futile compared to encoding position alone. Except for the smallest case, velocity again proved more useful than acceleration.

If the feature vector is to be composed of point positions alone, there are still many ways this can be formulated. I considered one alternative: representing equally spaced points in polar coordinates relative to a character's center of mass. The result of this study is shown in Figure 3.37. This method performed well at the extreme

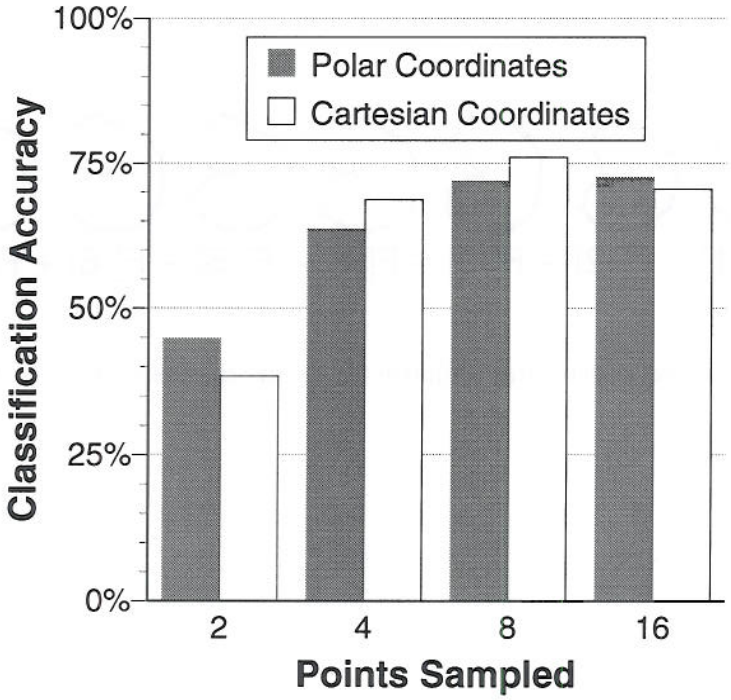


Figure 3.37: Top-choice character classification accuracy for equally spaced samples encoded as Cartesian and polar coordinates.

sizes, but it never attained the best accuracy of Cartesian coordinates.

A final possibility considered was encoding the coordinates in the frequency domain. This was done using a 1-dimensional, discrete Fourier transform along the trajectory (TDFT). The character was first resampled at equally spaced points. The coordinates of each point were converted to a complex number, according to the formula $x + iy$. These numbers were placed in a vector which is transformed to the frequency domain. This processing can be understood best by examining a low-pass filtered reconstruction of the character. In Figure 3.38 I have done this by inverse transforming successive frequency domain coefficients. The DC component encodes the center of mass for the character. Low frequency components define the overall shape while higher components provide detail. In classifying the TDFT, I considered the magnitude and phase of the coefficients independently. The results, shown in Figure 3.39, indicate that this representation is no better than the original resampled character.

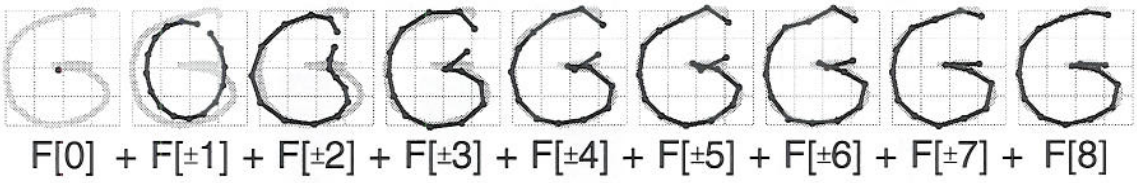


Figure 3.38: Reconstructing a character from successive Fourier coefficients.

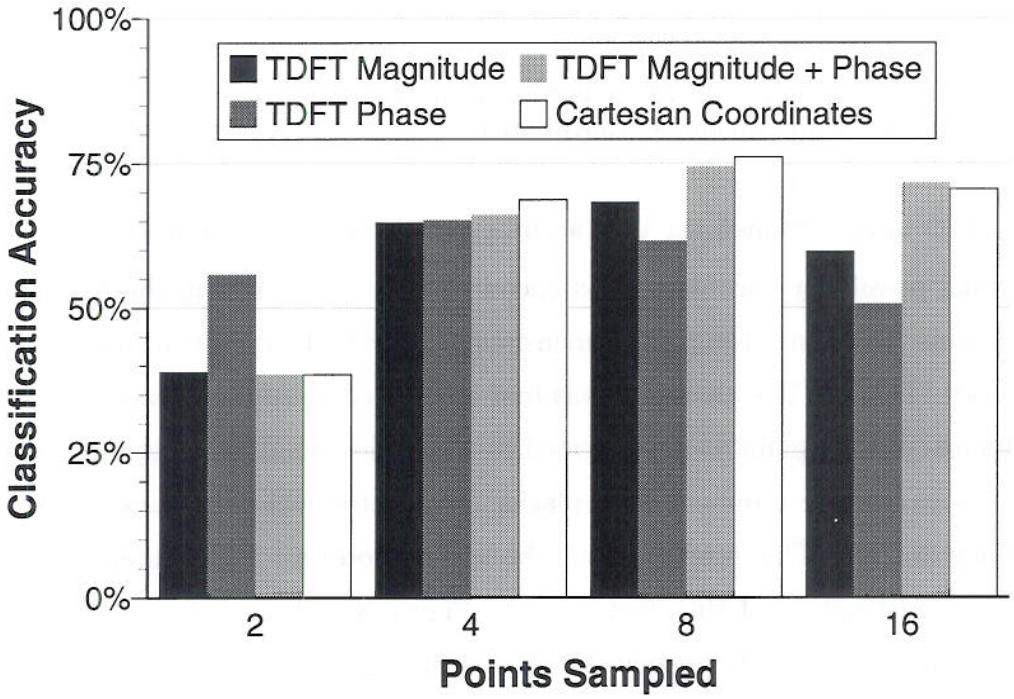


Figure 3.39: Top-choice character classification accuracy for a 1-dimensional frequency domain encoding of the pen trajectory.

Summary

In this section I have examined a number of representations based on resampling the pen trajectory of a character. The original motivation was to equalize the number of points forming each character to establish a uniform set of classification vectors. Additionally, I considered how to order and encode the points in each vector. The best representation used the Cartesian coordinates of eight equally spaced points to obtain an accuracy of 76.1%.

3.4.3 Trajectory Coding

A popular method of representing shapes for classification is the Freeman or chain code [2]. In its classic implementation, the pen's movement between samples is quantized into one of eight directions. The quantization boundaries are equally spaced with one bin centered on 0° to avoid jitter in encoding horizontal and vertical lines. Any pen trajectory can be represented as a string over an eight-symbol alphabet. Straight lines are represented as runs of a single symbol while curves are approximated by line segments. Characters with similar shapes will have similar chain codes. The similarity can be made scale-independent by reducing repeated symbols to single examples.

A particular shape may be identified from its chain code using syntactic pattern matching [19], parsing the string of direction symbols to form less primitive constructs. However, in order to make this approach comparable with other representations studied I did not introduce an additional pattern classification technique. Instead, the coded trajectory was used as a feature vector to the Gaussian mixture classifier. A cap was placed on the number of codes allowed in a given representation. When this limit was exceeded, codes representing the shortest segments were merged with their neighbors as needed. A reserved symbol is used to pad strings which are too short. This chain code representation has two parameters: the number of levels used to quantize the direction and the maximum number of symbols permitted in a descriptor string. The results of an experiment considering three values for each parameter are shown in Figure 3.40. For all quantization choices, permitting a maximum of eight codes per character gave the best performance. The eight-level quantization was con-

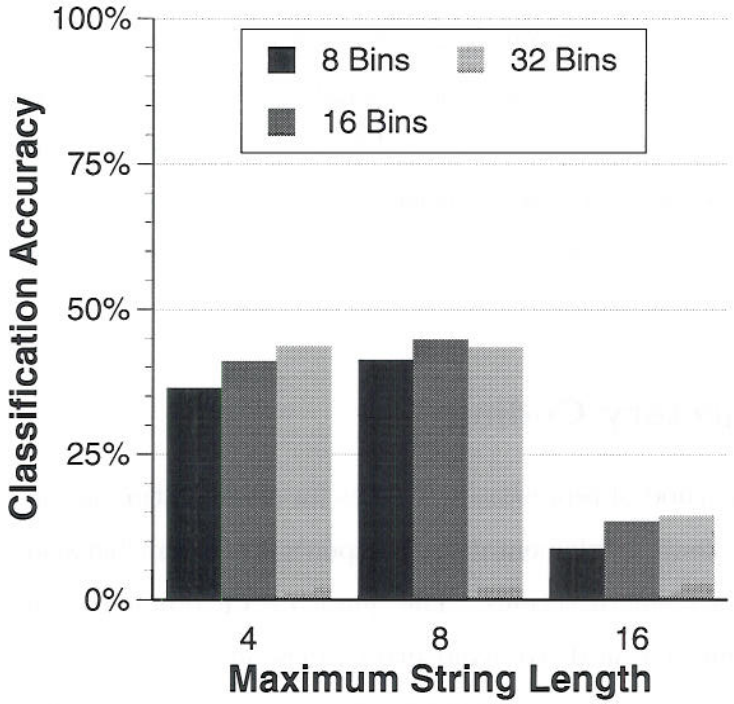


Figure 3.40: Top-choice character classification accuracy for chain codes.

sistently inferior to making finer distinctions. The extremely poor performance of the longest strings may be due to the same misalignment problem described for characters containing different numbers of samples. Resampling characters uniformly before applying the chain code might have proved beneficial.

I considered two variations of chain coding. A given line can be produced by either of two pen trajectories with directions separated by 180°. I postulated that it might be worthwhile to alias angles when encoding pen motion so that this distinction is hidden. Because this halves the range of values permitted, the resolution for a given number of bins is doubled. Alternatively, the relative lengths of each coded run could provide additional information about the character. After constructing the chain code string its elements are sorted by run length. This implicitly represents the relative length of lines within each character while maintaining the size-independence of the encoding. Unfortunately, both of these approaches reduced classification accuracy significantly.

I also considered an adaptive quantization of the trajectory angle. To construct

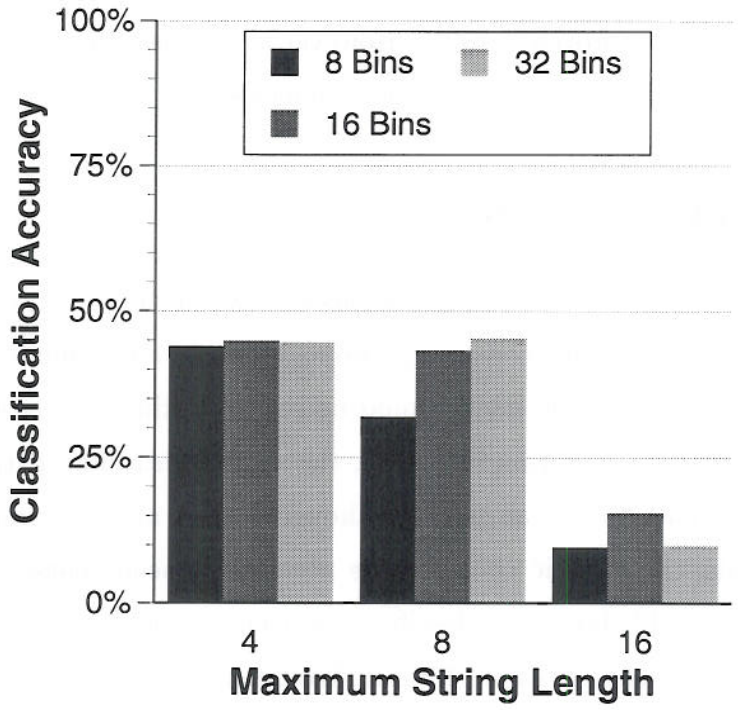


Figure 3.41: Top-choice character classification accuracy for cluster codes.

the representation, each segment of the trajectory is first represented by its angle to the horizontal. Adjacent segments which are most similar in direction are merged through a weighted average. This process is repeated until the angle between all segments and their neighbors exceeds a threshold. If needed, the resulting string is brought below a length limit as was done for chain coding. Because this approach clusters segments to quantize their direction, I call this a "cluster code." I tried this technique using thresholds matching the chain code bin widths described above. The results are shown in Figure 3.41. The performance was comparable to that of chain codes and in the best case slightly improved. None of the coded representations worked as well as classifying the trajectory's coordinates directly.

3.5 Improving Performance

Of the many representations examined, the best character classification accuracy was given by the Cartesian coordinates of equally-spaced samples. None of the vari-

ations on this technique – adding information, reordering the points, or alternate encodings – yielded improved results. However, this does not mean the maximum performance of this representation has been achieved.

3.5.1 Tuning Parameters

Ideally the parameters controlling the classifier would be tuned to the representation chosen, but this is a time consuming proposition. As a minimum, the parameters controlling the representation itself should be set to produce the most accurate system. For the best representation identified, the single controlling parameter specifies the number of points to be sampled. Although I have already considered a range of values for this, the specific choices have necessarily been sparse but will now be expanded. Figure 3.42 shows the classification accuracy of this representation as the number of points selected is varied from 2 through 16. The result is a broad plateau which peaks slightly at ten points, though nearly the same performance could be achieved with as few as six points. For brevity I will refer to the best parameterization of the best representation as the "champion representation," rather than "the cartesian coordinates of ten points equally-spaced along the pen trajectory."

Since this was the best representation identified it deserves further analysis. Its top-choice character classification accuracy was 77.2% for test data. Recall that this was accomplished without relative size or position information. The comparable human performance, albeit on a static representation, was 81.7%. A confusion matrix for the champion representation is shown in Figure 3.43. The errors are structured similarly to those already described, with case substitution quite prominent. Other highly confusable pairs, such as "o" with "0" and "h" with "n," are generally reasonable based on shape similarity. In some cases only particular allographs would be confusable, such as dotless "i" confused with "l" or North American "z" confused with "2." In other cases additional graphic information would be required to resolve the error. For example, "9" and "g" are confusable without knowing how the character is positioned relative to the baseline.

The cumulative accuracy for the champion representation, shown in Figure 3.44, is also encouraging. Not only does this representation achieve a relatively high top-

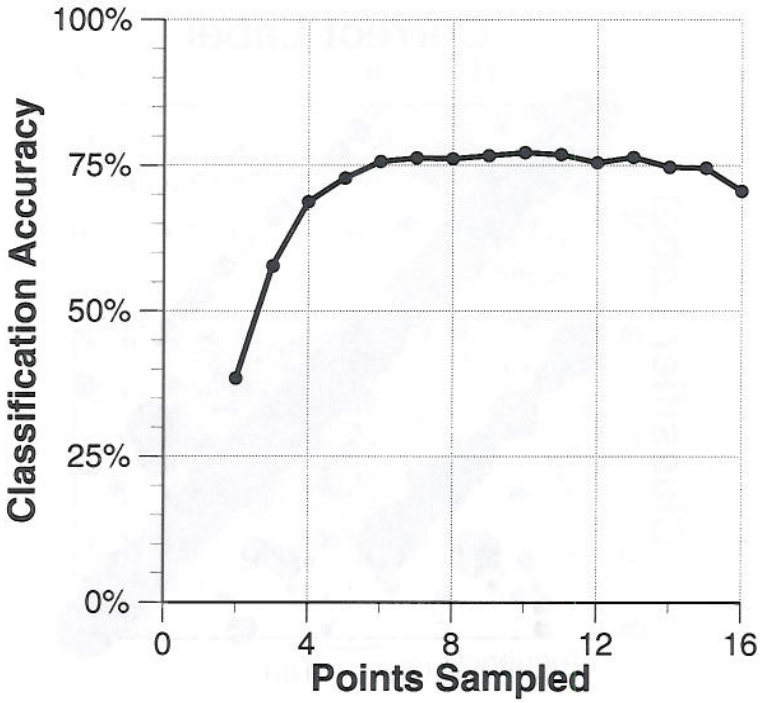


Figure 3.42: Optimizing the number of points sampled by the best representation.

choice accuracy, alternate candidates quickly cover the answer. The correct label was among the top 2 choices 90.1% of the time (compared to 89.7% for the untuned representation). This is not surprising given the large number of case errors and suggests that resolving such confusions could result in dramatically higher top-choice accuracy.

3.5.2 Subject Cohorts

Removing inter-subject variability by focusing on writer-dependent classification is likely to improve system accuracy provided that adequate training material is available. However, it is possible to realize a reduction in inter-subject variability, without incurring the disadvantages of writer-dependency, by considering subject cohorts. Subjects within each cohort should have similar writing characteristics. These cohorts might be created automatically though a clustering procedure, or they may be based on demographic variables. Independent classifiers are each trained on data from sub-

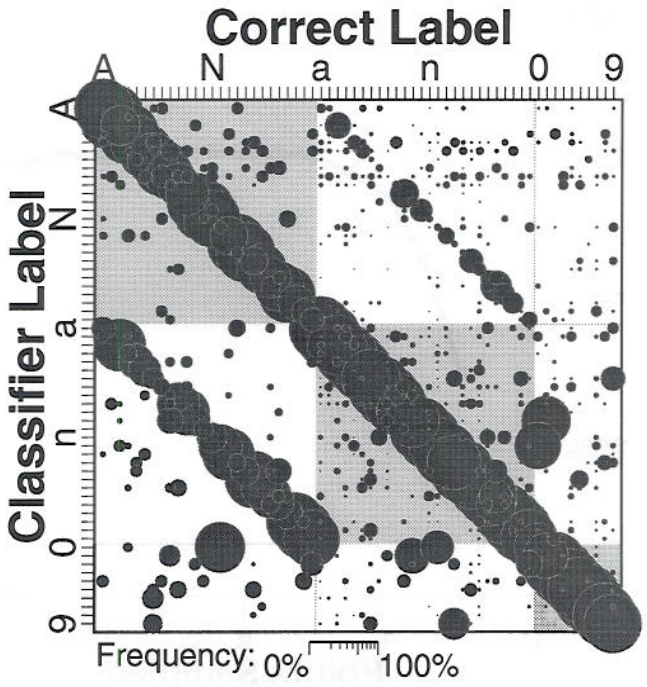


Figure 3.43: Confusion matrix for the champion representation.

jects in a single cohort. Material to be classified can be identified as belonging to a subject in a particular cohort and passed to the appropriate classifier. Alternatively, all classifiers can be run in parallel and the results combined through a decision rule.

I have examined subject cohort distinctions based on gender or writing hand. In both cases, subjects were assigned to one of two cohorts dependent on their responses to questions posed as part of data collection. The classifiers were evaluated under matched (training and testing cohorts agree) and crossed (training and testing cohorts disagree) conditions. The results of these studies are shown in Table 3.2. In both cases, relying on a larger but less homogeneous pool of training data provided the highest accuracy. Writing from females consistently scored higher than did writing from males, even in the cross-testing case.

3.5.3 Perturbation Training

The accuracy of the champion representation tested on the training set is 88.7%. The large disparity between the training and testing set results suggests that the

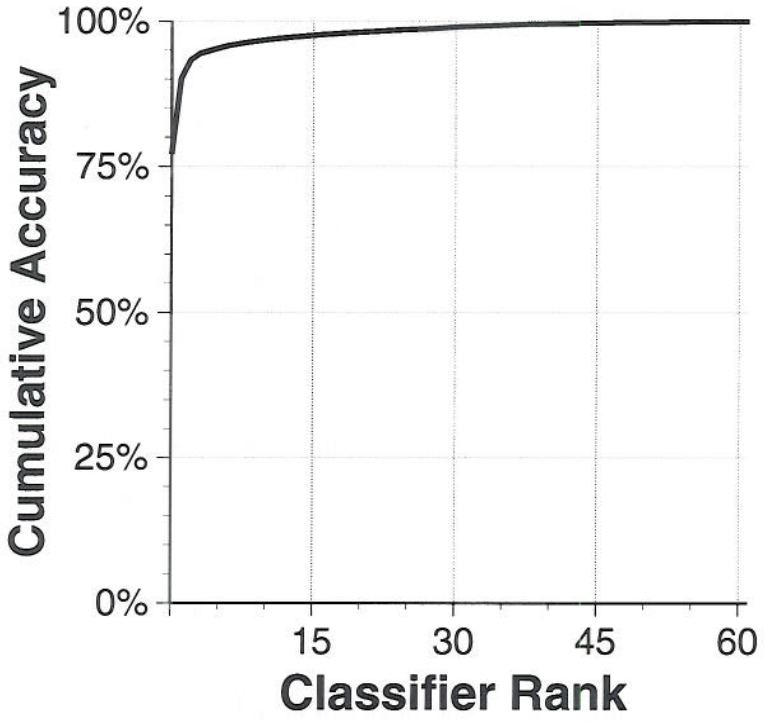


Figure 3.44: Cumulative accuracy for the champion representation.

Training Cohort	Testing Cohort		
	Male	Female	Both
Male	71.7%	77.7%	74.5%
Female	68.9%	78.8%	73.5%
Both	73.8%	81.1%	77.2%

Training Cohort	Testing Cohort		
	Right	Left	Both
Right	75.8%	74.9%	75.7%
Left	64.7%	74.1%	65.9%
Both	76.6%	81.5%	77.2%

Table 3.2: Classification based on subject cohorts dependent on gender (left) and writing hand (right).

classifier is not sufficiently generalized to cover unseen phenomena. One technique to improve this situation is to train the system on additional data. Unfortunately, more data may be difficult or impossible to obtain. Is there a way to extract greater utility from the existing training set?

The purpose of additional training material is to capture more of the variability observed in a test set. In some cases, this variability can be characterized analytically. For example, the size of a character can be represented as a scale factor. When this is possible, existing training material can be manipulated to approximate the unseen variability [5, 11]. In general this technique will give the best approximation when the manipulations are restricted to minor perturbations. However, one can imagine an extreme case of a classifier trained on a single token for each label, but with those exemplars manipulated along many dimensions to mimic test data.

I have chosen to explore character rotation as a potential application of perturbation training, but deformations varying character aspect ratio or sample point locations are also promising. Since the data collected was all at the same orientation, rotational variability was introduced primarily by writing slant. However, the same technique could be applied over a wider range of rotations to produce an orientation-independent classifier.

Each token in the training set was rotated clockwise and counterclockwise by a fixed step size to produce new training tokens. This procedure was repeated until a maximum rotation was reached. The original training token was retained. The experiment was conducted under different conditions of step size and rotation limit. In all but one case, as seen in Table 3.3, the character classification improved. In the best case the classification rate reached 79.1%. For those perturbation conditions, the performance on training data (including the rotated tokens) also increased to 91.4%. Perturbation of mainstream data has helped to cover outlier tokens. But changes to the outliers have pushed them even farther afield. Apparently, errors on newly formed outliers were spread over a wider pool of data, yielding the improved performance seen.

Maximum Rotation	Rotation Step Size: 2.5°	5°	10°	20°
5°	77.6%	76.4%		
10°	78.4%	79.1%	77.3%	
20°		79.0%	78.5%	78.4%

Table 3.3: Character classification accuracy for the champion representation based on various perturbations of the training set.

3.6 Summary

In this chapter I have examined many handwriting representations within the framework of character classification. I began by establishing human character classification performance for the test data. Next I examined static representations based on character images. A bitmap of each character, a representation akin to writing on a page, gave reasonable classification accuracy. However, representing the image with multiple projections yielded better performance. I showed how dynamic information could be incorporated in character images, but found that this improved performance only for the lowest resolution, most confusable images. In considering dynamic representations, the best performance was given by a simple representation based on the Cartesian coordinates of ten equally spaced points along the pen's trajectory, yielding a 20-dimensional feature vector. Adding to or manipulating this representation failed to improve its performance. The results of these experiments are summarized in Figure 3.45. The best representations incorporated dynamic information, but some static representations were close behind. None of the representations could achieve the accuracy of human authenticators using a static representation.

Having identified the highest-performing character representation, I showed how its performance could be improved further through tuning its parameters. Using the

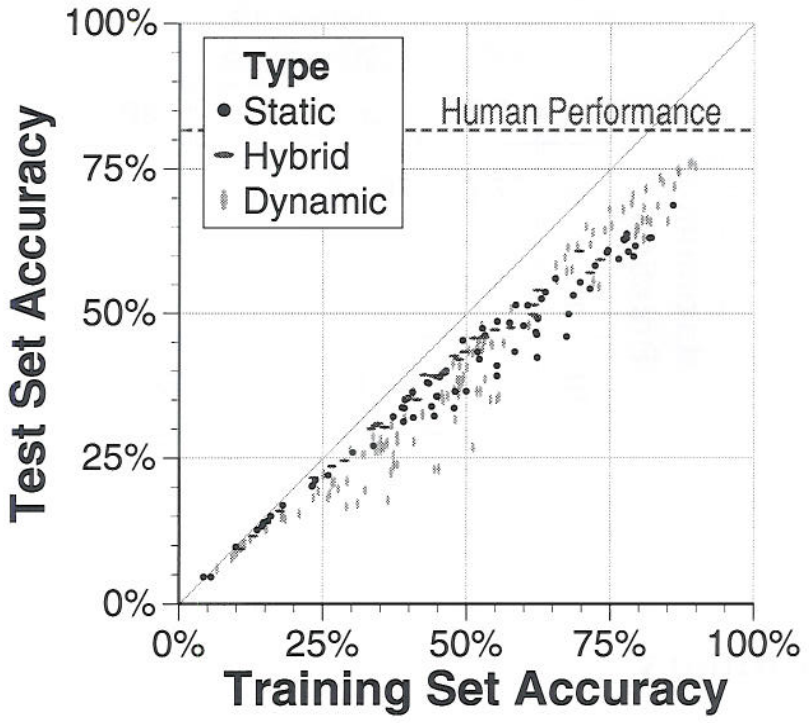


Figure 3.45: Character classification accuracies for the representations examined in this study.

best parameters for the best representation yielded a classifier with 77.2% top-choice accuracy and 90.1% top-2 accuracy. Finally, I showed how additional gains could be realized by perturbing the training data to better account for test set variability. The best top-choice accuracy achieved with this technique was 79.1%.

Chapter 4

Recognizer Development and Evaluation

It would be wrong to assume that the character classification results reported in the previous chapter accurately represent the performance expected of a handwriting recognition system. For classification, the writing corresponding to each character was identified by a near error-free hand transcription. This contrasts with a recognition system, in which the input is divided into regions by an automatic segmentation algorithm. The segmenter can make errors, inserting boundaries where none are necessary and omitting them where they are required. Even when boundaries are placed only at plausible positions, the correct segmentation can be ambiguous in the absence of higher-level knowledge, as shown in Figure 4.1. The classification component of the recognizer must also contend with partial and merged characters. These effects can introduce confusions unseen in classification experiments.

In this chapter I describe the construction of a recognition framework to test the best handwriting representation found for classification. The bulk of this work was in creating an automatic segmentation algorithm appropriate for the handprinted data of the corpus. To do so required characterizing the properties of the data which may indicate symbol boundaries.

4.1 Experimental Procedure

There are many approaches that could be pursued in creating a handwriting recognizer; again there is much optimization to be performed to obtain the highest

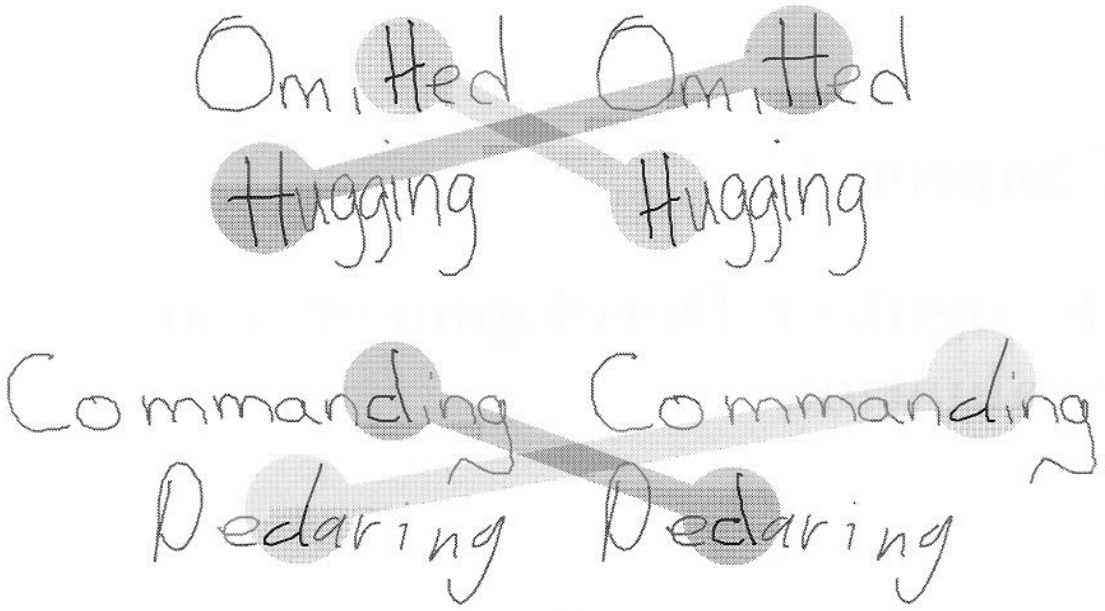


Figure 4.1: Context can determine if a shape is more likely to be interpreted as one or two characters, illustrated by exchanging handwriting between examples from the corpus.

performance possible. I have foregone most of this tuning and instead created a reasonable yet easily reproducible system. Because I view this work as only a starting point, it is important that others be able to closely duplicate my experiments to ensure their results are commensurable.

4.1.1 Ensuring Comparability

As with my classification experiments, I depended on existing technology to carry out these studies. The segmentation and feature extraction were particular to the handwriting problem, but the classification and search components were taken from the SUMMIT [93] speech recognizer. This classifier, with the appropriate controlling parameters, was identical to the one used for the classification experiments. The search phase allowed for probabilities assigned to each boundary to influence the selection among alternate segmentation paths.

I made other decisions to facilitate meaningful comparisons between classification and recognition experiments. This included using identical training and development

sets. In some cases maintaining comparability compromised the performance of the recognizer. Although neighboring symbols can influence a character's shape, context-independent models were used to better match the classification experiments' conditions. Similarly, language modeling can boost the accuracy of a system by penalizing responses with unlikely character strings. Character frequencies were captured when training the classifier, but higher-order statistics were not used for the bulk of my experiments. When they were incorporated it was only to determine the additional constraints they provided.

In classification studies based on hand-marked character boundaries one is free to ignore selected segments. However, a recognizer must contend with the entire handwriting stream. In my classification studies I discarded ink corresponding to pen skips and ligatures. For recognition I generally modeled these phenomena explicitly, treating them as additional characters in the alphabet. These symbols would not be seen by the writer and so were ignored in scoring the results.

4.1.2 Recognizer Construction

To accurately determine the effects of algorithmic differences, a single character representation should be selected for all studies. Because the recognition experiments were concurrent with the classification studies, the best representation was not identified in time to be used for system development. Instead, I selected 8×8 bitmaps as simple, competitive, and uncontroversial.

Ideally a recognizer could combine probabilities generated by its components to determine the relative worth of alternate results. The simplifying assumptions made in building actual recognizers distort probability estimates. For example, sequential tokens are often treated as statistically independent despite our knowledge to the contrary. Handling the component scores as if they were true probabilities is not guaranteed to identify the most likely result. Although they are unnecessary in theory, adjustment factors are required for a system to achieve the best performance. These controls weight the various sources of information and bias the results to contain more or fewer segments. An initial set of parameter values was selected to obtain the highest accuracy for a recognizer trained and tested on only a subset of the available data.

For each experiment, these parameters were reestimated iteratively to balance the number of insertion and deletion errors. At each iteration, the character models were retrained with data from the most likely correct segments to account for differences from the hand transcription.

The result of recognizing a handwriting sample is the highest-scoring character string. This string is evaluated by checking it against the transcription of the data. For classification this comparison is easy because each string contains exactly one character. However, with recognition the correct and hypothesized label strings may differ in length; a correlation between the two must be constructed before they can be compared. This was accomplished with a public domain program developed by NIST [57] for evaluating speech recognizers. It uses dynamic programming [64] to determine the best alignment between word strings and reports on insertion, deletion, and substitution errors. To adapt this to my recognition task, each character was treated as a separate word.

4.1.3 Segmentation Approach

There are many ways to automatically segment handwriting. At one extreme an exhaustive list of all possible segmentations may be proposed. Alternatively, one can avoid segmentation altogether by choosing a recognition strategy which implicitly divides the input. I chose an intermediate position, inserting boundaries to create character-sized segments.

The correctness of a segmentation can be evaluated by comparing its boundaries with those of a hand transcription. Some latitude in boundary position may be acceptable in lieu of an exact match. Because the speech signal is 1-dimensional, a time difference between corresponding boundaries is an effective measure of segmentation accuracy. The 2-dimensional nature of handwriting makes for a more difficult comparison. Characters may touch or overlap, situations inapplicable to speech. The third dimension added for on-line handwriting further complicates matters. In some cases, such as latent "t" crossing, small spatial differences can be widely separated in time while small temporal differences can be widely separated in space.

While a segmentation similar to the transcription should be adequate for recog-

nition, other divisions of the input can result in equally high performance. For this reason I elected to avoid the difficulties of comparing boundary positions by evaluating segmentation schemes in the context of recognition. That is, different segmentation algorithms were applied to otherwise identical recognizers.

Handwriting is an inherently multi-dimensional process, but the text it encodes is 1-dimensional. At some point in any handwriting recognizer the temporal signal must be mapped into a symbol stream. As with other operations, it is desirable to delay firm segmentation decisions as long as possible to bring as much information to bear on the problem. This suggests waiting until the search phase to find a linear path through the 2-dimensional image space. However, this would preclude using the search procedures developed for speech recognition.

The alternative is to treat handwriting as a 1-dimensional signal. There are several ways this can be accomplished. The signal could be recognized in temporal order, perhaps splitting characters into multiple symbols to allow for retrograde crossing and dotting. This output could be converted to words using an appropriate lexicon, or arbitrary character strings could be accepted by reordering the output symbols. The signal could also be recognized in temporal order after retrograde strokes are noted and removed. For example, the data associated with the stem of a "t" could be marked with a special designation indicating spatial overlap with data from a retrograde cross stroke. Even with that cross discarded, sufficient information is available to classify the vertical bar correctly rather than as the otherwise confusable "l."

Both of these techniques require special processing which makes character recognition less like the classification problem. The approach I selected better retained the comparability, permitted a linear segmentation, and was readily implemented. Strokes in the input were sorted based on the horizontal position of their centers. The motivation for this was derived from the 3 strokes used to form a "tt" sequence with a common cross stroke. To best match the prototype shape of each "t," the cross needs to be split between the two symbols and so should be ordered between the two stems.

All of my segmentation algorithms depended on having access to the writing in its entirety. Prior to segmentation the handwriting strokes were sorted according to

their position. Spatially adjacent strokes could then be grouped and divided into character-sized segments, forming a directed acyclic graph.

4.1.4 Summary

The handwriting recognition experiments I performed were based on a segmental speech recognizer. The bulk of the work required to construct the system was in developing the segmentation algorithm. To avoid the difficult and perhaps inappropriate problem of evaluating a segmenter's accuracy, the performance of recognizers incorporating alternate segmentation algorithms were compared. For this comparison to be meaningful, other factors affecting recognition had to be held constant. These experiments were based on an 8×8 bitmap image representation with other options selected to ensure comparability with the classification experiments.

4.2 Segmenting at Pen-Lifts

Segmentation is the first step in recognizing handwriting. In the formalism I have adopted, additional segments cannot be proposed in later stages of processing. Accordingly, an algorithm which over-generates, producing spurious segments in addition to the "correct" ones, is favored over one which under-generates and misses needed regions.

A distinguishing characteristic of hand printing is that symbols are generally isolated from one another. It makes sense to exploit this property in creating a segmentation algorithm. For on-line data, this is best manifested as a pen-up between characters. A benefit of this cue is that it allows for spatial overlap between symbols.

How often were character boundaries coincident with pen-lifts? This was determined using the hand-aligned transcriptions. Each pair of neighboring characters was extracted from the training data. If the characters had no strokes in common, a pen-lift must have been present between them. Using this criterion, 98.4% of the character boundaries corresponded to pen-lifts, confirming the prevalence of this attribute.

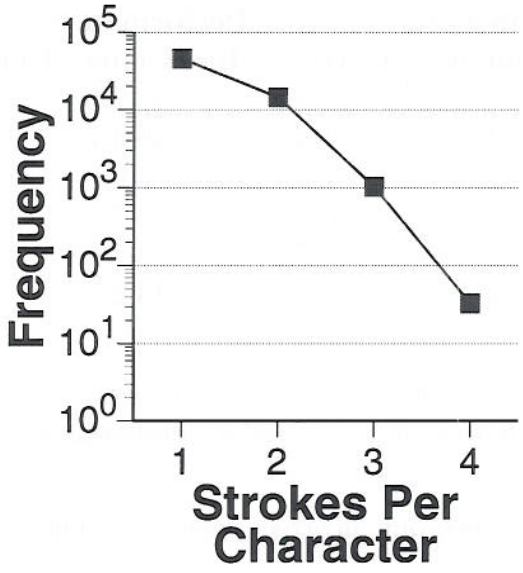


Figure 4.2: Number of strokes per character for the training set data.

4.2.1 Uniform Boundary Probability

While pen-lifts served as a powerful indication of character boundaries, they were also present internal to characters constructed from multiple strokes. Simply producing one segment per stroke was insufficient for matching character boundaries; segments including multiple strokes had to be hypothesized. Each stroke could be assigned to more than one segment and any overlap would be dealt with in the search phase.

Examining transcription tokens from the training data, statistics were collected on the number of strokes per character. These are shown in Figure 4.2. No character observed is constructed from more than four strokes. Single stroke characters are the most common and are three orders of magnitude more frequent than four stroke characters.

A straightforward way to account for the range in strokes per character is to propose segments for all combinations of 1 to n neighboring strokes. The likely limit is $n = 4$, but I considered values of n from 1 to 6. A uniform probability of 75.8% was assigned to each boundary created. This value is the fraction of pen-lifts that occur between, rather than within, characters. The approach depended solely on

Strokes per Segment	Performance		
	Correct	Insertions	Deletions
1	44.6%	17.2%	7.6%
1, 2	51.3%	16.3%	3.6%
1–3	52.2%	14.6%	4.0%
1–4	50.9%	13.7%	4.2%
1–5	50.7%	13.7%	4.6%
1–6	50.8%	13.2%	4.7%

Table 4.1: Character recognition performance for segments constructed from varying number of strokes and with uniform boundary probability.

poor classifier scores to weed-out incorrect segments. The results of this experiment are shown in Table 4.1. As might be expected, creating segments from more than four strokes did not provide the best performance. Interestingly, the most correct characters were recognized when segments were limited to only three strokes. Some insight into why this was so can be gained from the surprisingly high effectiveness of allowing only a single stroke per segment. Not only did this model the bulk of the data, multi-stroke characters could often be identified from less than their full complement of strokes. For comparison purposes, character classification based on an 8×8 bitmap representation correctly identified 60.7% of the symbols.

4.2.2 Classifying Pen-Lifts

The uniform boundary probability incorporated in these experiments was simple to compute but not particularly helpful in determining the best segmentation. If there is a property which can better predict the likelihood of a boundary it can be exploited to improve the system's accuracy.

Note that at character boundaries one expects the pen to move horizontally since English is written from left to right. At each pen-lift the direction traveled from pen-up to pen-down was computed. A histogram of these directions, extracted from the training data, is shown in Figure 4.3. Indeed, the pen did tend toward the right between characters, typically with an upward or downward component as well. Within characters the pen-lifts tended toward up and to the left, with some traveling

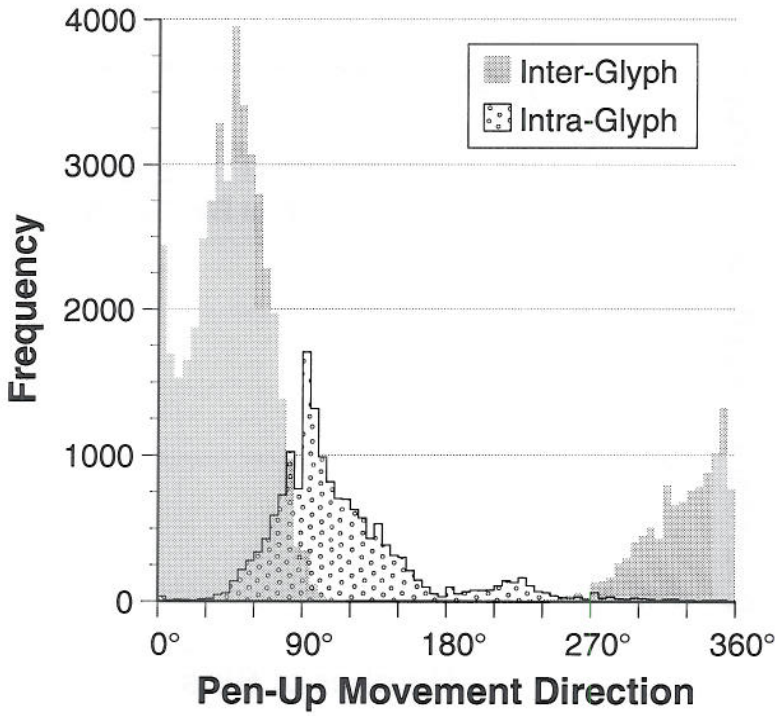


Figure 4.3: Histogram of pen travel directions at potential boundaries.

rightward and even fewer downward. Although there was some overlap between these two categories, this simple property provided remarkably clean separation between the distributions.

In order to incorporate this source of information in the segmenter, I constructed a non-parametric model to estimate the probability of a character boundary given the pen-lift's travel direction. The non-parametric model divided the training data by angle into equally spaced bins. The frequency of data within each bin was used to estimate the probabilities. The number of bins in the model was selected to provide the best classification performance on development data. In addition, a half bin width offset was considered to determine the sensitivity to angle quantization. The results of this optimization are shown in Figure 4.4. Using 18 bins and no offset, 92.5% of the pen-lifts were correctly classified as between- or within-characters.

To test the effects of this information source, segments were constructed as before from all possible combinations of one through four neighboring strokes. However,

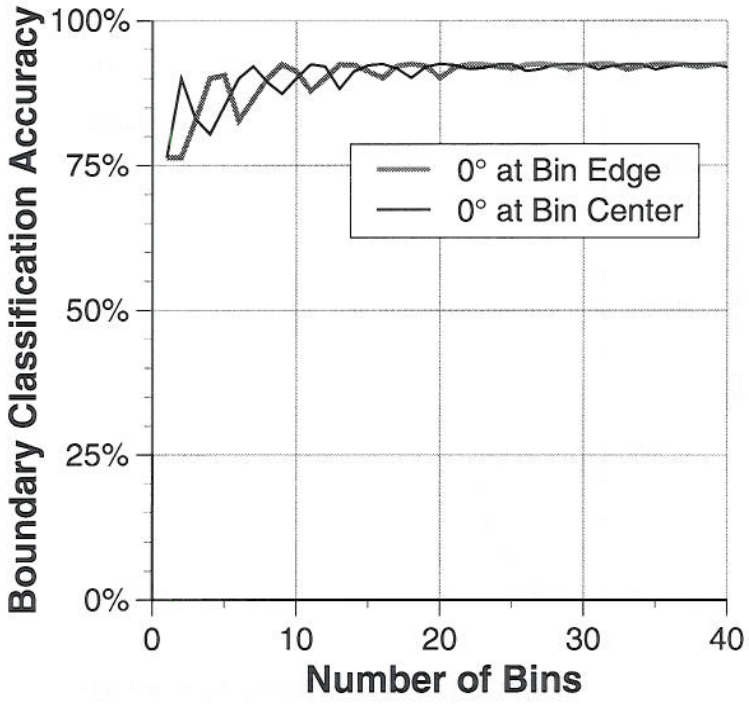


Figure 4.4: Optimizing bin count and offset for non-parametric modeling of pen travel directions at potential boundaries.

in this experiment the probability of each boundary was given by the classifier. In Table 4.2 this approach is compared to using a uniform boundary probability. Not only did the fraction of characters correctly identified increase, both insertion and deletion errors were reduced. In fact, these results were better than for any uniform probability case examined.

4.3 Segmenting Connected Characters

I next turned to segmenting connected characters. Since these are characterized by having some stroke in common, this problem can be decomposed into identifying shared strokes followed by selecting boundaries within these strokes. A viable alternative to segmenting joined characters is modeling the more frequent connected strings as symbol-like units of their own.

An important observation about connected characters is that they are rare. Fig-

Boundary Probability	Performance		
	Correct	Insertions	Deletions
Uniform	50.9%	13.7%	4.2%
Predicted	54.3%	11.7%	3.6%

Table 4.2: Character recognition performance using uniform and predicted boundary probability.

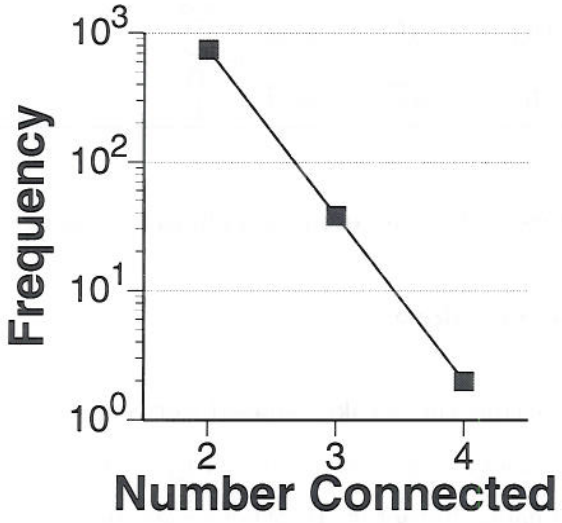


Figure 4.5: Frequency of connected character strings in the training set.

ure 4.5 shows the frequency of connected character strings as a function of their length. Not only are connected characters rare in handprinting, sequences of more than two connected characters are *extremely* rare.[1]

Accordingly, I decided to focus on connected pairs of characters only. These are not distributed uniformly. The most common connected character pairs are shown in Table 4.3, listing both their count in the training set and the fraction of those characters which are connected. A joined "ti" forms the most frequent connected pair, but less than one-fifth of the potential connections are realized. Although less frequent, over two-fifths of the "tt" sequences were connected. As can be seen in the examples, a particular pair can have several different written forms.

[1]This statistic is biased by the fact that data were rejected from subjects who seemed to rely on cursive writing.

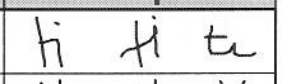
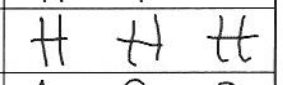
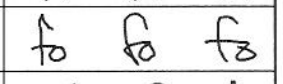
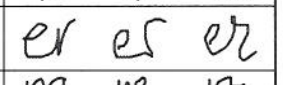
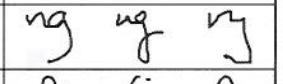
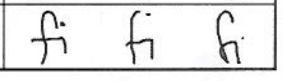
Pair	Frequency		Examples
	#	%	
ti	227	19.5	
tt	65	44.8	
fo	62	24.1	
er	54	24.1	
ng	51	2.6	
fi	47	14.9	

Table 4.3: The most common connected character pairs in the training set.

4.3.1 Splitting Strokes

Identifying and splitting the strokes shared between character pairs is a difficult problem due to their variety, demonstrated in Figure 4.6. In some cases the ligature includes only a part of each character. In other cases, both characters are constructed in their entirety from a single stroke. The connecting stroke can be straight, gently curved, or sharply bent.

Characters are generally written in sequence, left to right, regardless of their being connected or not. As a result, strokes which connect characters should finish to the right of where they started. A histogram showing stroke direction, computed from the starting and ending points, is shown in Figure 4.7. Strokes connecting characters do tend to progress down and toward the right. Unfortunately, so do many strokes within single characters. Worse, connecting strokes are two orders of magnitude less frequent than the others. A parametric model was constructed and optimized just as described before, but the wide difference in frequency precluded *any* strokes from being identified as connecting. Rather than correct classification, the optimization criteria was changed to maximizing the probability assigned to connecting strokes.

The variety in shared strokes also poses a problem for identifying segmentation points. I decided on a simple approach, segmenting strokes at two fixed positions.

Figure 4.6: A variety of strokes (darker lines) shared between two characters.

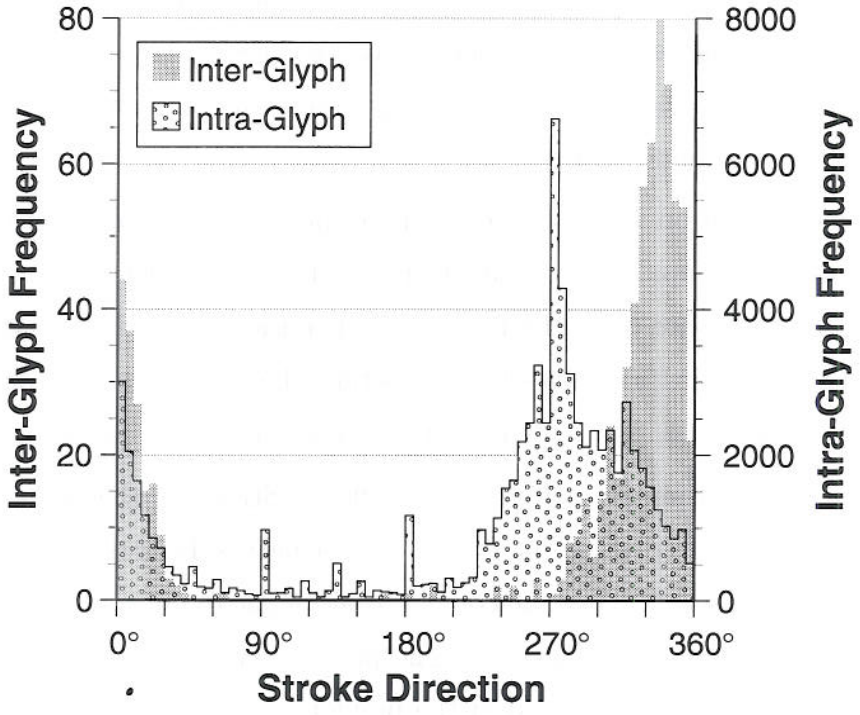


Figure 4.7: Histogram of stroke directions for potential connections.

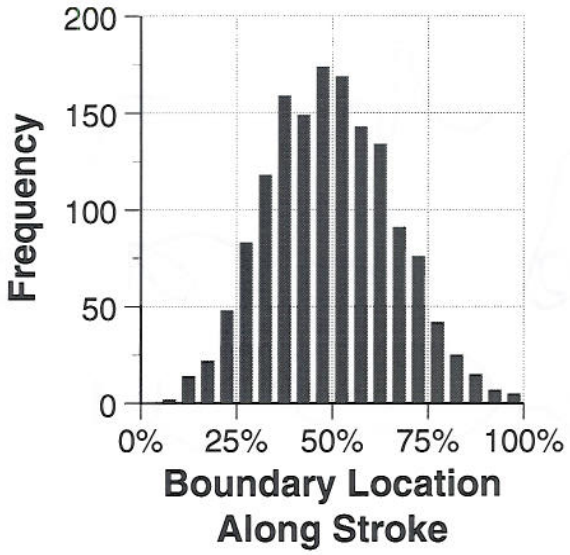


Figure 4.8: Histogram of boundary locations along connecting strokes.

Providing two boundaries gave added flexibility to the segmentation: one or both could be ignored in the search phase. This permitted the central region to serve as a ligature if needed and allowed the better single boundary to prevail when all of the stroke was used in forming characters.

A boundary's location within a stroke can be specified as a fraction of the distance from pen-down to pen-up. A histogram of these distances, extracted from the training data, is plotted in Figure 4.8. The mean for this data is very close to the half-way position. The one-third and two-thirds positions lie approximately one standard deviation from the mean, providing convenient points for inserting boundaries.

The final segmentation procedure was as follows. Strokes with non-zero probability of being shared were divided into three equal-length sub-strokes. The undivided stroke was retained as well. Then, all combinations of one to four adjacent strokes and sub-strokes were proposed as segments. The probability of a boundary at a pen-lift was given by the classifier described in the previous experiment. The probability of boundaries within a stroke was one-half the probability that the stroke was shared. Table 4.4 shows the performance of a recognizer incorporating this segmentation. At best, limited improvement over the previous system can be expected because con-

Segmentation	Performance		
	Correct	Insertions	Deletions
Pen-lifts Only	54.3%	11.7%	3.6%
Add Stroke Splitting	54.7%	7.8%	4.4%

Table 4.4: Character recognition performance when segmenting at pen-lifts only and when including boundaries within possible connecting strokes.

Representation	Performance		
	Correct	Insertions	Deletions
8×8 Bitmap	54.7%	7.8%	4.4%
Champion	65.1%	9.1%	3.7%

Table 4.5: Character recognition performance using two writing representations.

nected characters are so rare. Splitting strokes gave a modest gain in the fraction of characters recognized correctly. The number of insertions was reduced while the number of deletions increased.

4.4 Variations

The recognition experiments described so far were all based on an 8×8 bitmap representation. The champion representation, Cartesian coordinates of ten equally spaced samples within each character, yielded the best classification performance of any representation tested. Using this representation for recognition should provide similar performance gains. The results of doing so are shown in Table 4.5. Changing representations reduced the error rate by 23% while maintaining comparable insertion and deletion rates. For comparison, classification using this representation correctly identified 77.2% of the symbols. The remaining recognition experiments were all based on the champion representation.

The recognition alphabet included symbols for pen skips and ligatures. These symbols were not counted in evaluating recognition performance, but they undoubtably had an indirect influence. An alternative was to omit these regions altogether. Character models were seeded from the hand transcription as before. In retraining the

Character Set	Performance		
	Correct	Insertions	Deletions
Alphanumeric + Specials	65.1%	9.1%	3.7%
Alphanumeric	67.4%	8.3%	2.7%

Table 4.6: Character recognition performance using two alphabets. Special characters represent pen skips and ligatures.

recognizer, the extraneous ink will be assigned to one or both of the neighboring symbols and incorporated in those character's models. The results of this experiment are shown in Table 4.6. This simple change gave a small increase in accuracy. Either the pen skips and ligatures could not be classified reliably as independent units or their incorporation in character models helps clarify symbol variability.

Explicit contextual constraints, in the form of a language model, can also improve recognition performance. The classifier used in these experiments models the frequency of each character, and even each character's allographs. To determine the potential power of higher-order statistics, a bigram grammar can be applied to the search phase of recognition. The constraining power of this grammar is derived from restrictions on spelling patterns, the structure of upper- and lower-case letters within words, and the high mutual exclusivity of letters and digits within a particular string.

The grammar was constructed from over 14-million words and numbers containing over 65-million characters from the New York Times newswire data described on page 44. All punctuation was discarded, but capitalization was retained. Special symbols were used to indicate string start and end. Character pairs which were not observed were assigned a frequency of 1 to make all letter sequences permissible. Because the text analyzed is large and quite general, one would expect the resulting grammar to provide relatively weak constraints.

One measure of a grammar's constraining power is its perplexity [34], roughly defined as the average branching factor. The perplexity of a bigram character grammar may be computed as

$$2^{\left(-\sum_{c_1,c_2} P(c_1,c_2)\log_2 P(c_2|c_1)\right)}$$

Character Grammar	Perplexity	Performance		
		Correct	Insertions	Deletions
Unigram	36.0	67.4%	8.3%	2.7%
Bigram	11.3	76.4%	7.6%	2.1%

Table 4.7: Character recognition performance incorporating a bigram character grammar.

where c_1 and c_2 are characters from the alphabet in consecutive positions. Even this loose grammar has a perplexity of only 11.3, slightly more than half the perplexity considering only unigram statistics (20.0) and far less than the perplexity of uniformly distributed characters (63.0). The unigram perplexity is so low because lower-case symbols are much more common than other characters. For comparison, the bigram perplexity computed over training set transcriptions is 8.6 and the corresponding unigram perplexity is 36.0. The results of applying the bigram to the recognition process are shown in Table 4.7. Despite the broad task modeled, a 27% reduction in error rate was achieved. More limiting grammars (for example, constraining results to words in a lexicon) should yield even better performance.

4.5 Summary

In this chapter I have developed a handwriting recognizer using classification and search components from a segment-based speech recognition system. The classification component was identical to that used in my classification experiments, allowing the effects of automatic segmentation to be observed.

Most of the experiments I conducted were related to segmenting the handwriting into character-like regions. In all cases, multiple segmentations were proposed for each input. I have shown the effectiveness of segmenting handprinted data at pen-lifts. This technique was improved by estimating the probability of each boundary based on the direction of pen travel between strokes. I have shown that connected characters are relatively rare yet pose difficult challenges for segmentation. Still, a simple approach of identifying potential shared strokes and splitting them at fixed locations along their trajectory was able to improve system performance.

Having selected a segmentation scheme, the performance of the recognizer could be increased by using an improved representation. Additional gains were realized by allowing character models to incorporate neighboring pen skips and ligatures. Finally, I showed how incorporating even a loose bigram grammar can substantially improve recognition results.

Chapter 5

Summary and Future Directions

5.1 Summary

In this thesis I have presented a comprehensive series of experiments aimed at a better understanding of automatic handwriting recognition for on-line printed text. This included designing, collecting, and transcribing a suitable corpus of handwriting data; running human authentication experiments to determine the difficulty of character classification; comparing handwriting representations through automatic classification; and constructing a recognition system to observe the effects of automatic segmentation. Rather than searching for a high-accuracy system, my goal was to report on the performance of incremental experiments as a basis for additional studies.

The handwriting corpus developed for my studies incorporates several novel features. Its design is based on a set of variable length character sequences identified by an information theoretic metric. These sequences, and others chosen for their research interest, were covered by words selected automatically to achieve compact coverage. Handwriting was recorded from many subjects to validate my results for writer-independent systems. A minimum of influences on this writing was achieved by a large writing area, few instructions, and aural prompting. The handwriting corpus was transcribed and the data aligned with the transcriptions.

Handwriting data can have a wide range of consistency and confusability. To better understand the difficulty of the task at hand, I ran an authentication experiment on the training portion of my handwriting corpus. Mimicking the conditions of char-

acter classification, individual characters were excised using the aligned transcriptions and presented to authenticators in random order. Nearly 1 out of 5 characters were misidentified under these circumstances.

The classification studies I conducted were based on a single, flexible classifier. Experiments were designed to provide a meaningful comparison between various representations and all characters were normalized to ensure position and scale independence. In theory, making additional information available to the classifier should improve accuracy. However, I found that these gains were often offset by the performance reduction of additional classifier complexity. Incorporating dynamic information available from on-line data was not always beneficial. Nonetheless, some of the better representations observed did profit from this information. The best representation found was simple: a 20-dimensional vector containing the Cartesian coordinates of 10 equally spaced points along the pen trajectory. By making small perturbations to the training data, the classification accuracy could be improved further.

To construct a complete handwriting recognizer, I relied on the classification and search components of a speech recognition system. In segmenting the data, I showed how a boundary classifier based on pen travel could improve performance. I demonstrated that connected characters pose a significant challenge to segmentation and proposed a simple solution which improved recognition accuracy further.

5.1.1 Evaluation Results

I have saved reporting on a final set of experiments for the very end of this document. All of the experiments I have described so far were based on the training and development sets. The evaluation set has been excluded from all studies to prevent any tuning to its data. Maintaining an unseen data set for testing is required to truly evaluate system performance. Purely for comparison purposes with future studies, I now present selected systems tested on the evaluation data.

For each of these experiments I consider two training conditions. In one approach, the designated training set is used just as it has been in previously described experiments. Only the test data is different, allowing for a fair comparison between the two testing conditions. In the other approach the development set is used for train-

Training Data	Testing Data	Tokens Correct
Training Set	Development Set	77.2%
Training Set	Evaluation Set	81.0%
Training and Development Sets	Evaluation Set	79.9%
Human Authentication	Development Set	81.7%
Human Authentication	Evaluation Set	84.1%

Table 5.1: Character classification performance based on the evaluation data.

ing as well. This gives a glimpse of how additional training data may affect system performance. Only the champion representation was applied in these studies.

The results for character classification are summarized in Table 5.1. The somewhat higher accuracy achieved by training on perturbed data is not shown. In comparing classification results across test sets, evaluation data gave the better results. A similar difference was observed in the authentication studies. Interestingly, additional training data lowered system accuracy somewhat. These two facts suggest that some part of the development set is particularly difficult to classify, perhaps because in some respect it is unlike the remainder of the data. This explanation is reinforced by the poor classification accuracy for some subjects' data, as reported on page 84.

Similar experiments were conducted for character recognition. Tests were run without and with a character bigram grammar. Neither case included explicit regions for pen skips and ligatures. The results from these studies are shown in Table 5.2. The pattern of improved accuracy on the evaluation set is repeated here. Unlike for classification, additional training data from the development set had a positive impact on recognition performance.

It is impossible to make a direct comparison between these performance figures and system performance described in the literature. Certainly, others have claimed higher accuracies. However, the relative difficulty of the task must be considered in making a qualitative evaluation. Given that my work was based on a character set with highly confusable symbols, that handwriting was collected in a relatively uncon-

Without Bigram Grammar				
		Performance		
Training Data	Testing Data	Correct	Insertions	Deletions
Training Set	Development Set	67.4%	8.3%	2.7%
Training Set	Evaluation Set	71.7%	6.8%	3.0%
Training and Development Sets	Evaluation Set	74.4%	4.7%	2.9%

With Bigram Grammar				
		Performance		
Training Data	Testing Data	Correct	Insertions	Deletions
Training Set	Development Set	76.4%	7.6%	2.1%
Training Set	Evaluation Set	79.3%	5.6%	2.1%
Training and Development Sets	Evaluation Set	82.7%	3.2%	2.4%

Table 5.2: Character recognition performance based on the evaluation data.

strained manner from a large number of subjects, and that testing was performed on an independent data set without the benefit of high-level constraints, I feel that the performance I have reported is at least comparable to earlier results.

5.2 Future Directions

There are many possible ways this work could be continued. In fact, it is my hope that the corpus I collected will be made available to other researchers. Parts of this document serve to eliminate replication of the basic studies which should be common to all recognition experiments performed with this handwriting data.

5.2.1 Optimization

Performance could drive many extensions to my studies. For example, the concurrent optimization of all system aspects is likely to result in the highest recognition accuracy. Due to time limitations, the systems I developed are far from optimum. If nothing more, the parameters controlling classification and search could be better tuned to the chosen representation.

More powerful modeling techniques should also result in higher accuracy. Better language models, perhaps driven by a particular application, would place tighter constraints on the search space. At the character level, context dependent models would account for more of the data's variability. Various subject-dependent constraints could be modeled explicitly, perhaps to the extent of creating adaptive systems. Additional contextual information could be incorporated to capture relative character size and placement.

Performance gains can also be achieved by relaxing some of the assumptions I have made in system development. This requires scrutinizing each experiment and testing the obvious. For example, describing the best representation found as "the Cartesian coordinates of 10 equally spaced points" should raise many questions I have not addressed. I have shown that equally spaced sampling was somewhat better than equally timed samples, but perhaps some other spacing criteria would yield better performance. One can imagine sampling 100 equally spaced points from each character. Through an optimization procedure, a subset of these points could be selected to achieve the highest classification performance. The result could be a small number of unequally spaced samples.

Similarly, such an optimization might discard some of the position components from some samples. The completeness of taking two coordinates from all points may be appealing, but higher accuracy might result from extracting only x coordinates from some points and only y coordinates from others. This optimization can also be applied after principal component analysis. A search for the best components is required rather than a simple truncation. Earlier components capture the greater variances in the data, but this is not equivalent to providing the best dimensions for discrimination. More formally, some type of discriminant analysis [15] may prove beneficial.

Additional work could be task driven. Many applications of handwriting recognition will involve highly portable systems. These systems can be characterized as having limited memory and processing capability to conserve space, weight, and power. The demand for efficient algorithms is amplified by a desire for real time systems. None of my studies have addressed the trade-off between computational requirements and system accuracy.

Finally, additional training data can improve the performance of almost any classification or recognition system.

5.2.2 New Areas

The studies I have conducted could also be extended in ways that are more than simple refinements. For example, there are an unlimited number of representations one might consider. Alternate segmentation approaches, particularly ones which better handle connected characters, are another fertile area of study. I have already mentioned the possibility of searching a 2-dimensional segment graph. Another alternative is an exhaustive segmentation which encompasses all contiguous point sequences.

I have completely ignored commonly applied preprocessing techniques for handwriting. To establish robust techniques, features such as character baselines should be hand marked in the data. These would be used to develop and test normalization procedures. Traditional goals such as rotating baselines to the horizontal are secondary to providing some consistent end product. For example, a principal component analysis of data points might transform data consistently but distort writing from a visual standpoint. Ultimately, it is the effect of these manipulations on recognition performance that must be measured.

Additional data could supply handwriting needed to extend the domain of the recognition task. For example, I collected boxed character data but gave it only a cursory examination. The subjects supplying data for my studies were directed to print their responses. By changing or eliminating this instruction, the very same procedures I developed could be applied to collecting cursive or mixed writing styles. The character set could be extended as well. One possibility is to include more of the punctuation and symbols found on keyboards. Another is to include the diacritics rarely used in English texts but important for foreign proper nouns and some borrowed words. Handling word, line, and paragraph breaks will be important for many applications.

To successfully deploy handwriting recognition systems in real applications, studies of additional areas are needed. For example, thinking of the writing process as

inscribing characters sequentially simplifies automatic processing. However, a range of spontaneous production phenomena do not fit this model. Cross-outs, insertions, restarting, and overwriting are some of the ways errors can be corrected within the non-linear writing stream. Better systems will be able to interpret these indications rather than requiring the subject to use artificial editing gestures.

The view of handwriting recognition I have worked with is purely automatic and always returns a response. An alternative is to view the recognition process as a cooperative one between the writer and the recognizer. In this arrangement, the recognizer may call on the user to clarify ambiguous input. This also serves to provide feedback to the user on writing clearly. Identifying ambiguity requires the development of appropriate rejection criteria.

5.3 Parting Comments

I have proposed that recent advances in speech recognition system development can be transferred to on-line handwriting recognition. This requires more than just a sharing of technology: applying the correct methodology is vital. Systems *must* be developed using large amounts of training data to be reliable. Writing has to be collected under conditions matching actual usage to the degree possible. Care should be taken to avoid unduly influencing the subjects. Only the most aberrant material may be discarded. Test data needs to be disjoint from training data and cannot be used for system tuning. Consistency is demanded between experiments to attribute performance changes to individual factors. Incremental advances should be sought in suggesting new avenues of study. All sources of constraint should be exploited. Every aspect of a system requires justification, consideration, and optimization. The obvious should not escape this scrutiny and even the absurd occasionally should be evaluated. Finally, automatic methods can better cope with the volume of material to be processed than can researcher-intensive tallying and estimation.

In my work I have strived to maintain these ideals as best as resources would permit. While it is difficult, at present, to compare my results directly to existing systems, I feel I have shown that competitive performance is possible using relatively simple techniques. Furthermore, these results were achieved without fully tuning the

recognition system. I expect somewhat better performance could be brought forth with nothing more than better optimization of system control parameters. It is not difficult to concoct simple corrections to some known deficiencies of the best system described without resorting to the rule-based preprocessing techniques or representations so common in other handwriting recognition studies.

While my studies were all geared towards recognizing handwriting, by no means do I view it as the perfect means of interacting with a computer. No single method will be best for all users performing all chores. Handwriting will provide an additional option. Even then, recognition is not necessary for all tasks. For example, recorded handwriting is sufficient for note taking. In this context, accurate recognition may be less important than algorithms to compress handwriting for storage and to match handwriting for searching.

Humans are extremely proficient at a variety of pattern recognition problems. Recognizing speech and handwriting are so natural to us they seem easy at first inspection. It is hard to say which of these tasks is more difficult, and it is not even clear that a meaningful comparison can be made between them. However, both fields offer rich areas of study that are likely to offer new research challenges for many years. The uninformed are likely to dismiss instructing computers to perform speech and handwriting recognition as simple. However, as one examines data the subtleties of the problem are revealed. Rather than contempt, for these very difficult problems familiarity breeds respect.

Bibliography

[1] A. Apte, V. Vo, and T. D. Kimura, "Recognizing multistroke geometric shapes: An experimental evaluation," in *6th Annual Symposium on User Interface Software and Technology*, (Atlanta, Georgia), pp. 121–128, ACM, ACM Press, Nov. 1993.

[2] D. H. Ballard and C. M. Brown, *Computer Vision*, pp. 235–236. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1982.

[3] D. H. Ballard and C. M. Brown, *Computer Vision*, pp. 123–131. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1982.

[4] E. J. Bellegarda, J. R. Bellegarda, D. Nahamoo, and K. S. Nathan, "A fast statistical mixture algorithm for on-line handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, pp. 1227–1233, Dec. 1994.

[5] D. Beymer, A. Shashua, and T. Poggio, "Example based image analysis and synthesis," A.I. Memo 1431, MIT Artificial Intelligence Laboratory, Cambridge, Massachusetts, Nov. 1993.

[6] M.-Y. Chen, A. Kundu, and J. Zhou, "Off-line handwritten word recognition using a hidden Markov model type stochastic network," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, pp. 481–496, May 1994.

[7] K. Church, W. Gale, P. Hanks, and D. Hindle, "Parsing, word associations and typical predicate-argument relations," in *International Workshop on Parsing Technologies*, (Pittsburgh, Pennsylvania), pp. 389–398, Carnegie Mellon University, 1989.

[8] E. Cohen, J. J. Hull, and S. N. Srihari, "Understanding handwritten text in a structured environment: Determining zip codes from addresses," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, pp. 221–264, June 1991.

[9] S. Di Zenzo, M. Del Buono, M. Meucci, and A. Spirito, "Optical recognition of hand-printed characters of any size, position, and orientation," *IBM Journal of Research and Development*, vol. 36, pp. 487–501, May 1992.

[10] Y. A. Dimitriadis, J. L. Coronado, and J. L. C. Vidal, "An adaptive resonance theory architecture for the automatic recognition of on-line handwritten symbols of a mathematical editor," in *Artificial Neural Networks: International Workshop IWANN '91*, (Grenada, Spain), pp. 216–226, Springer Verlag, Sept. 1991.

[11] H. Drucker, R. Schapire, and P. Simard, "Improving performance in neural networks using a boosting algorithm," in *Advances in Neural Information Processing Systems* (S. J. Hanson, J. D. Cowan, and C. L. Giles, eds.), vol. 5, (Denver Colorado), pp. 42–49, Morgan Kaufman, 1993.

[12] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, pp. 11–15, Jan. 1972.

[13] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, pp. 103–105. New York: John Wiley and Sons, Inc., 1973.

[14] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, pp. 230–237. New York: John Wiley and Sons, Inc., 1973.

[15] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, pp. 114–121. New York: John Wiley and Sons, Inc., 1973.

[16] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics: Principles and Practice*, pp. 72–81. Reading, Massachusetts: Addison-Wesley Publishing Company, Inc., second ed., 1990.

[17] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics: Principles and Practice*, pp. 132–142. Reading, Massachusetts: Addison-Wesley Publishing Company, Inc., second ed., 1990.

[18] G. D. Forney, "The Viterbi algorithm," *Proceedings of the IEEE*, vol. 61, pp. 268–278, Mar. 1973.

[19] K. S. Fu, *Syntactic Pattern Recognition and Applications*. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1982.

[20] T. Fujisaki, T. T. Chefalas, J. Kim, C. C. Tappert, and C. G. Wolf, "On-line run-on character recognizer: Design and performance," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, pp. 123–137, June 1991.

[21] T. Fujisaki, C. C. Tappert, M. Ukelson, and C. G. Wolf, "Online recognition of unconstrained handprinting: A stroke based system and its evaluation," in *From Pixels to Features III: Frontiers in Handwriting Recognition* (S. Impedovo and J. C. Simon, eds.), (Bonas, France), pp. 297–312, International Association for Pattern Recognition, North-Holland, 1992.

[22] R. G. Gallager, *Information Theory and Reliable Communication*, pp. 16–27. New York: John Wiley and Sons, Inc., 1968.

[23] D. K. Gifford, D. A. Segal, and R. Cote, "Clipping service user's manual (version 1.3)," Technical Report MIT/LCS/TR-398, Massachusetts Institute of Technology Laboratory for Computer Science, Cambridge, Massachusetts, July 1987.

[24] L. Gillick and S. J. Cox, "Some statistical issues in the comparison of speech recognition algorithms," in *International Conference on Acoustics, Speech, and Signal Processing*, vol. S1, (Glasgow, Scotland), pp. 532–535, IEEE Signal Processing Society, May 1989.

[25] J. D. Gould and L. Alfaro, "Revising documents with text editors, handwriting-recognition systems, and speech-recognition systems," *Human Factors*, vol. 26, no. 4, pp. 391–406, 1984.

[26] V. K. Govindan and A. P. Shivaprasad, "Character recognition – a review," *Pattern Recognition*, vol. 23, no. 7, pp. 671–683, 1990.

[27] W. Guerfali and R. Plamondon, "Normalizing and restoring on-line handwriting," *Pattern Recognition*, vol. 26, no. 3, pp. 419–431, 1993.

[28] A. Gupta, M. V. Nagendraprasad, A. Liu, P. S. P. Wang, and S. Ayyadurai, "An integrated architecture for recognition of totally unconstrained handwritten numerals," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 7, pp. 757–773, Aug. 1993.

[29] I. Guyon, "Applications of neural networks to character recognition," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, pp. 353–382, June 1991.

[30] L. D. Harmon, "Automatic recognition of print and script," *Proceedings of the IEEE*, vol. 60, pp. 1165–1176, Oct. 1972.

[31] J. J. Hull and R. K. Fenrich, "Large database organization for document images," in *Fundamentals in Handwriting Recognition*, vol. 124 of *NATO ASI Series F: Computer and Systems Sciences*, pp. 397–414, Berlin: Springer Verlag, 1993.

[32] S. Impedovo, L. Ottaviano, and S. Occhinegro, "Optical character recognition – a survey," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, pp. 1–24, June 1991.

[33] F. Jelinek, "Self-organized language modeling for speech recognition," in *Readings in Speech Recognition* (A. Waibel and K.-F. Lee, eds.), section 8.1, pp. 450–455, San Mateo, California: Morgan Kaufman Publishers, Inc., 1990.

[34] F. Jelinek, "Self-organized language modeling for speech recognition," in *Readings in Speech Recognition* (A. Waibel and K.-F. Lee, eds.), section 8.1, pp. 472–477, San Mateo, California: Morgan Kaufman Publishers, Inc., 1990.

[35] R. A. Johnson and D. W. Wichern, *Applied Multivariate Statistical Analysis*, ch. 8. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., third ed., 1992.

[36] A. Kay and A. Goldberg, "Personal dynamic media," *IEEE Computer*, vol. 10, pp. 31–41, Mar. 1977.

[37] D. D. Kerrick and A. C. Bovik, "Microprocessor-based recognition of handprinted characters from a tablet input," *Pattern Recognition*, vol. 21, no. 5, pp. 525–537, 1988.

[38] P. Kierkegaard, "A method for detection of circular arcs based on the Hough transform," *Machine Vision and Applications*, vol. 5, pp. 249–263, 1992.

[39] T. Kohonen, *Self-Organization and Associative Memory*, ch. 5, pp. 125–161. Berlin: Springer-Verlag, 1984.

[40] J. Krevisky and J. L. Linfield, *The Bad Speller's Dictionary*. New York: Random House, 1967.

[41] T. T. Kuklinski, "Components of handprint style variability," in *7th International Conference on Pattern Recognition*, vol. 2, (Montreal, Canada), pp. 924–926, International Association for Pattern Recognition, IEEE Computer Society Press, July 1984.

[42] J. M. Kurtzberg and C. C. Tappert, "Segmentation procedure for handwritten symbols and words," *IBM Technical Disclosure Bulletin*, vol. 25, pp. 3848–3852, Dec. 1982.

[43] L. F. Lamel, R. H. Kassel, and S. Seneff, "Speech database development: Design and analysis of the acoustic-phonetic corpus," in *Proceedings, Speech Recognition Workshop* (L. S. Baumann, ed.), (Palo Alto, California), pp. 100–104, Defense Advanced Research Projects Agency, Science Applications International Corporation, Feb. 1986.

[44] V. F. Leavers, *Shape Detection in Computer Vision Using the Hough Transform*. Berlin: Springer Verlag, 1992.

[45] K.-F. Lee, *Automatic Speech Recognition: The Development of the SPHINX System*, section 6.3, pp. 100–102. Boston: Kluwer Academic Publishers, 1989.

[46] C. G. Leedham and A. C. Downton, "Automatic recognition and transcription of Pitman's handwritten shorthand: An approach to shortforms," *Pattern Recognition*, vol. 20, no. 3, pp. 341–348, 1987.

[47] H. C. Leung, *The Use of Artificial Neural Networks for Phonetic Recognition.* Ph.D. thesis, Massachusetts Institute of Technology, May 1989. pp. 54–56.

[48] J. Makhoul, T. Starner, R. Schwartz, and G. Chou, "On-line cursive handwriting recognition using hidden Markov models and statistical grammars," in *Proceedings of the Human Language Technology Workshop*, (Plainsboro, New Jersey), pp. 432–435, Advanced Research Projects Agency, Mar. 1994.

[49] S. Manke and U. Bodenhausen, "A connectionist recognizer for on-line cursive handwriting recognition," in *International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, (Adelaide, South Australia), pp. 633–636, IEEE Signal Processing Society, Apr. 1994.

[50] G. J. McLachlan and K. E. Basford, *Mixture Models: Inference and Applications to Clustering*, ch. 2. New York: Marcel Dekker, Inc., 1988.

[51] M. L. Meeks and T. T. Kuklinski, "Measurement of dynamic digitizer performance," in *Computer Processing of Handwriting* (R. Plamondon and C. G. Leedham, eds.), pp. 89–110, Singapore: World Scientific, 1990.

[52] Merriam-Webster Inc., *Merriam-Webster Pocket Dictionary.* Merriam-Webster Inc., 1964. Computer readable form.

[53] S. Mori, C. Y. Suen, and K. Yamamoto, "Historical review of OCR research and development," *Proceedings of the IEEE*, vol. 80, pp. 1029–1058, July 1992.

[54] C. Nadal, R. Legault, and C. Y. Suen, "Complementary algorithms for the recognition of totally unconstrained handwritten numerals," in *10th International Conference on Pattern Recognition*, vol. 1, (Atlantic City, New Jersey), pp. 443–449, International Association for Pattern Recognition, IEEE Computer Society Press, June 1990.

[55] R. Nag, K. H. Wong, and F. Fallside, "Script recognition using hidden Markov models," in *International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, (Tokyo Japan), pp. 39.7.1–39.7.4, IEEE Signal Processing Society, Apr. 1986.

[56] K. S. Nathan, J. R. Bellegarda, D. Nahamoo, and E. J. Bellegarda, "On-line handwriting recognition using continuous parameter hidden Markov models," in *International Conference on Acoustics, Speech, and Signal Processing*, vol. 5, (Minneapolis, Minnesota), pp. 121–124, IEEE Signal Processing Society, Apr. 1993.

[57] National Institute of Standards and Technology, "Alignment and scoring software." CD-ROM, NIST Speech Disc 17-4.2, March 1995.

[58] F. Nouboud and R. Plamondon, "On-line recognition of handprinted characters: Survey and beta tests," *Pattern Recognition*, vol. 23, no. 9, pp. 1031–1044, 1990.

[59] F. Nouboud and R. Plamondon, "A structural approach to on-line character recognition: System design and applications," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, pp. 311–335, June 1991.

[60] T. Pavlidis, "Recognition of printed text under realistic conditions," *Pattern Recognition Letters*, vol. 14, pp. 317–326, Apr. 1993.

[61] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," in *Readings in Speech Recognition* (A. Waibel and K.-F. Lee, eds.), section 6.1, pp. 267–285, San Mateo, California: Morgan Kaufman Publishers, Inc., 1990.

[62] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition.* Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1993.

[63] S. Rhee. Personal communication, 1992. Microsoft Corporation.

[64] D. Sankoff and J. B. Kruskal, eds., *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, ch. 1. Reading, Massachusetts: Addison-Wesley Publishing Company, Inc., 1983.

[65] M. Schenkel, I. Guyon, and D. Henderson, "On-line cursive script recognition using time delay neural networks and hidden Markov models," in *International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, (Adelaide, South Australia), pp. 637–640, IEEE Signal Processing Society, Apr. 1994.

[66] M. Schenkel, H. Weissman, I. Guyon, C. Nohl, and D. Henderson, "Recognition-based segmentation of on-line hand-printed words," in *Advances in Neural Information Processing Systems* (S. J. Hanson, J. D. Cowan, and C. L. Giles, eds.), vol. 5, (Denver Colorado), pp. 723–730, Morgan Kaufman, 1993.

[67] L. Schomaker, "Using stroke- or character-based self-organizing maps in the recognition of on-line, connected cursive script," *Pattern Recognition*, vol. 26, no. 3, pp. 443–450, 1993.

[68] J. W. Schoonard, J. D. Gould, M. Bieber, and A. Fusca, "A behavioral study of a computer hand print recognition system," Research Report RC 12494, IBM T.J Watson Research Center, Yorktown Heights, New York, Feb. 1987.

[69] E. Sicard, "An efficient method for the recognition of printed music," in *11th International Conference on Pattern Recognition*, vol. 3, (The Hague, Netherlands), pp. 573–576, International Association for Pattern Recognition, IEEE Computer Society Press, Aug. 1992.

[70] S. J. Smith, M. O. Bourgoin, K. Sims, and H. L. Voorhees, "Handwritten character classification using nearest neighbor in large databases," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, pp. 915–919, Sept. 1994.

[71] S. N. Srihari, "High-performance reading machines," *Proceedings of the IEEE*, vol. 80, pp. 1120–1132, July 1992.

[72] S. N. Srihari, "Recognition of handwritten and machine-printed text for postal address interpretation," *Pattern Recognition Letters*, vol. 14, pp. 291–302, Apr. 1993.

[73] T. Starner, J. Makhoul, R. Schwartz, and G. Chou, "On-line cursive handwriting recognition using speech recognition methods," in *International Conference on Acoustics, Speech, and Signal Processing*, vol. 5, (Adelaide, South Australia), pp. 125–128, IEEE Signal Processing Society, Apr. 1994.

[74] C. Y. Suen, "Handwriting generation, perception, and recognition," *Acta Psychologica*, vol. 54, pp. 295–312, 1983.

[75] C. Y. Suen, R. Legault, C. Nadal, M. Cheriet, and L. Lam, "Building a new generation of handwriting recognition systems," *Pattern Recognition Letters*, vol. 14, pp. 303–315, Apr. 1993.

[76] C. Y. Suen, M. Berthod, and S. Mori, "Automatic recognition of handprinted characters – the state of the art," *Proceedings of the IEEE*, vol. 68, pp. 469–487, Apr. 1980.

[77] C. Y. Suen, C. Nadal, R. Legault, T. A. Mai, and L. Lam, "Computer recognition of unconstrained handwritten numerals," *Proceedings of the IEEE*, vol. 80, pp. 1162–1180, July 1992.

[78] C. C. Tappert, "Speed, accuracy, and felxibility trade-offs in on-line character recognition," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, pp. 79–95, June 1991.

[79] C. C. Tappert, C. Y. Suen, and T. Wakahara, "On-line handwriting recognition – a survey," Research Report RC 14045, IBM T.J Watson Research Center, Yorktown Heights, New York, Dec. 1987.

[80] C. C. Tappert, C. Y. Suen, and T. Wakahara, "On-line handwriting recognition – a survey," in *9th International Conference on Pattern Recognition*, vol. 2, (Rome, Italy), pp. 1123–1132, International Association for Pattern Recognition, IEEE Computer Society Press, Nov. 1988.

[81] C. C. Tappert, A. S. Fox, J. Kim, S. E. Levy, and L. L. Zimmerman, "Handwriting recognition on transparent tablet over flat display," in *International Symposium. Digest of Technical Papers*, (San Diego, California), pp. 308–312, SID, Palisades Inst. Res. Services, May 1986.

[82] C. C. Tappert, C. Y. Suen, and T. Wakahara, "The state of the art in on-line handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 787–808, Aug. 1990.

[83] C. W. Therrien, *Decision Estimation and Classification: An Introduction to Pattern Recognition and Related Topics*, p. 218. New York: John Wiley and Sons, Inc., 1989.

[84] T. Wakahara, H. Murase, and K. Odaka, "On-line handwriting recognition," *Proceedings of the IEEE*, vol. 80, pp. 1181–1194, July 1992.

[85] P. S.-P. Wang and A. Gupta, "An improved structural approach for automated recognition of handprinted characters," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, pp. 97–121, June 1991.

[86] G. Ward, *MobyWords*. Illumind Unabridged, 571 Belden St., Ste. A, Monterey, CA 93940, macintosh version 1.3 ed., Feb. 1991.

[87] J. R. Ward and T. Kuklinski, "A model for variability effects in handprinting with implications for the design of handwriting character recognition systems," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 18, no. 3, pp. 438–451, 1988.

[88] J. R. Ward and B. Blesser, "Interactive recognition of handprinted characters for computer input," *IEEE Computer Graphics and Applications*, vol. 5, pp. 24–37, Sept. 1985.

[89] J. R. Ward and M. J. Phillips, "Digitizer technology: Performance characteristics and the effects on the user interface," *IEEE Computer Graphics and Applications*, vol. 7, pp. 31–44, Apr. 1987.

[90] A. M. Wing, "Variability in handwritten characters," *Visible Language*, vol. 13, no. 3, pp. 283–298, 1979.

[91] C. G. Wolf, "Understanding handwriting recognition from the user's perspective," in *Proceedings of the Human Factors Society 34th Annual Meeting*, vol. 1, (Orlando, Florida), pp. 249–253, The Human Factors Society, The Human Factors Society, Oct. 1990.

[92] V. Zue, J. Glass, D. Goodine, H. Leung, M. Phillips, J. Polifroni, and S. Seneff, "The VOYAGER speech understanding system: Preliminary development and evaluation," in *International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, (Albuquerque, New Mexico), pp. 73–76, IEEE Signal Processing Society, Apr. 1990.

[93] V. Zue, J. Glass, D. Goodine, M. Phillips, and S. Seneff, "The SUMMIT speech recognition system: Phonological modelling and lexical access," in *International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, (Albuquerque, New Mexico), pp. 49–52, IEEE Signal Processing Society, Apr. 1990.