Multiple Algorithms for Handwritten Character Recognition

Jonathan J. HULL, Alan COMMIKE and Tin-Kam HO

Department of Computer Science
State University of New York at Buffalo
226 Bell Hall
Buffalo, New York 14260
hull@cs.buffalo.edu

Abstract

The recognition of handwritten characters that were written without constraints is considered. The particular domain of interest is postal addresses. It has been seen that because of the wide variety of writing styles in this domain, a set of three algorithms applied in parallel has yielded high rates of digit recognition performance. A similar strategy is being employed for character recognition. Three independent algorithms that use different styles of features (holistic, contour, and structural) are being developed. By utilizing independent feature information, it is expected that high rates of success can be achieved. This paper discusses the current status of the development of this approach, work in progress, problems and future challenges.

1. Current Status

The specific problem that is addressed by this work is the recognition of handwritten words in postal addresses. An isolated character recognition method is needed for these words so that ZIP Codes can be assigned to addresses without them and ZIP Codes can be verified on other addresses [3]. The scope of this problem can be seen in Figure 1 where various examples of handprinted city and state names are shown.

1.1. Methodology

The design strategy we have employed is illustrated in Figure 2. Three independent character recognition algorithms are applied to each character image and their results are combined. The character recognition algorithms were chosen because they use different features that have yielded high performance and independent errors for handwritten digit recognition. In particular, we are using a template matching algorithm, a statistical classifier of structural features, and a syntactic classifier of contour features. A similar algorithmic structure has yielded better than 91 percent correct with less than a 1.5 percent error rate on digit recognition within unconstrained handwritten ZIP Codes [5].

1.2. Assumptions

Our methods assume that the input is one of A-Z or a-z. To simplify the recognition process, certain pairs of visually similar upper and lower case characters are combined into single classes. These are Cc, Kk, Oo, Ii, Pp, Ss, Uu, Vv, Ww, Xx, Yy, and Zz. Also, the U and V classes are combined for the same reason. Thus, overall 40 classes are recognized.

The database that is currently being used for these experiments consists of about 20,000 isolated handprinted characters. These were extracted from about 2000 handwritten postal addresses that were

Int. Workshop on Frontiers in Handwriting Recognition, Montreal, Canada, April 2-3, 1990.

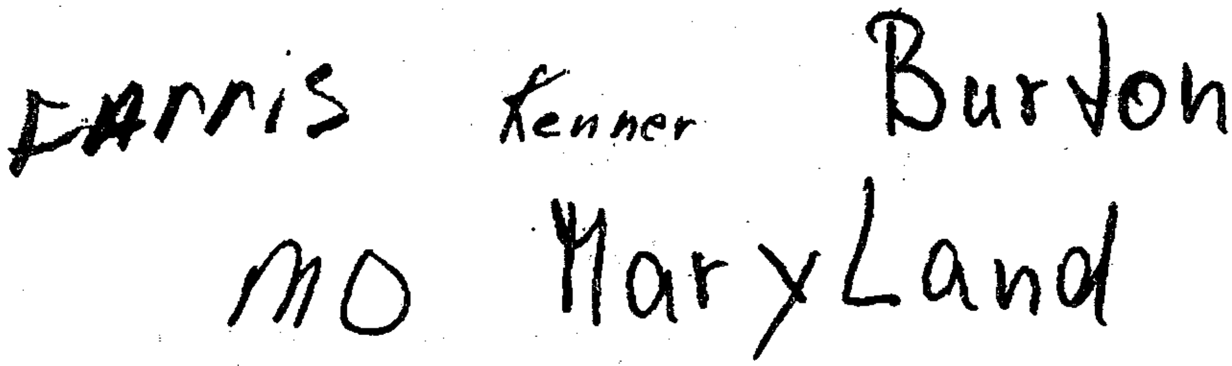

Figure 1. Examples of handprinted city and state names

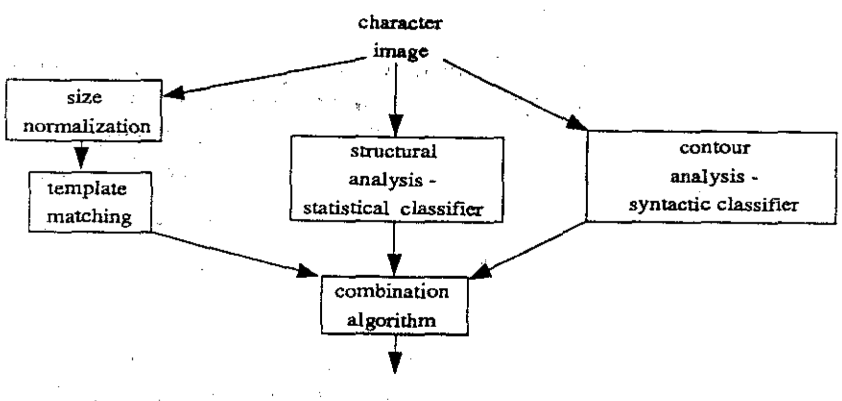

Figure 2. Algorithmic structure for character recognition.

digitized at 300 pixels per inch with eight bits of grayscale per pixel. Connected component analysis was used to locate blobs that were about the size of characters. The characters were manually identified and stored in the database.

1.3. Template Matching Algorithm

The template matching algorithm has been fully implemented and tested. An input character is size-normalized to a 16x16 grid and compared by a Hamming distance to a set of size-normalized prototypes. The N classes or the N prototypes that most closely match an input character are then determined. Up to 18,000 prototypes have been used at one time with this technique. Experiments have shown that performance steadily improves as more prototype data is added. This is because the large number of variations in handprinted text are better represented as some of the more obscure prototypes are added to the training data.

The performance of this algorithm has been determined with a training set of 18,000 characters and a test set of 2,000 characters. The results of this analysis are shown in Table 1. The percentage correct is shown for a given number of classes. The error rate is 100 minus this value. It is seen that the technique is 95.8 percent correct at guessing the input is among four of the 40 classes.

Number of Choices	1	2	3	4
Percent Correct	90.1	93.7	95.2	95.8

Table 1. Current Performance of Template Matching

1.4. Structural Analysis Algorithm

The structural analysis algorithm has been fully implemented and tested. It partitions a character with a 5x5 grid and determines the presence or absence of a horizontal or vertical stroke, a hole, a crosspoint, an endpoint, or a small or large concavity in each grid cell. These 175 features are supplemented with five more that describe global features of the character. This feature vector of 180 components is then input to a Bayesian classifier that determines the top N classes that most closely match the input character [4].

This approach has also been tested with the same 18,000 training and 2000 test data that were used for the template matching program. The results of this test are shown in Table 2. The first choice performance of this technique is the same as that of the template matcher. However, the four-choice performance of the structural method is better than that of the template matcher.

Number of Choices	1	2	3	4
Percent Correct	90.1	96.0	97.1	97.6

Table 2. Current Performance of Structural Method

1.5. Contour Analysis Algorithm

A contour analysis method has been fully developed for digit recognition and is being modified for character recognition. This method calculates the curvature at every point along the inner and outer contours of a binary image. Eight types of features are defined based on the amount of curvature present at any point. The features are similar to those described in [1]. Three features are used for concave curvature, and five for convex curvature. Each feature is also associated with its direction and location. The feature string extracted from an unknown character is matched against a rule base to achieve recognition.

Figure 3 shows examples of the eight features as well as how direction and location are defined. Direction is quantized to eight compass points and location is quantized to a 4x4 cartesian grid with the origin in the upper left.

The feature string that was extracted from the image of the three shown in Figure 4 is shown below:

B5x2y2 S4x1y1 B5x3y0 S4x0y0 A0x3y0 B0x2y1 A0x3y2 S3x0y3 B3x2y3

Each entry describes the presence of one feature: type, direction, and location from left to right, the first feature that was extracted was a Bay in the middle of the image facing SW. Then a Spur was detected towards the top facing W, and then another Bay, and so on.

The feature string is then compared against a rule base that contains over 130 rules. Classification is done by a decision tree. Each rule narrows down the possible choices for the next branch of the tree. If no rule matched in the current branch, the classifier backtracks to the previous branch and tries again for a match.

The final rule that matched the above feature string is:

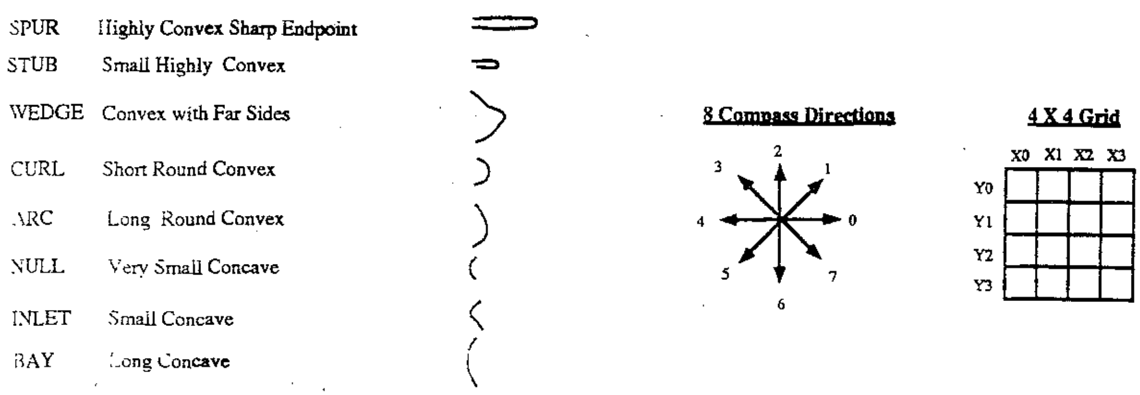

Figure 3. Features and their possible directions and locations

Figure 4. The image of a '3'.

Class: 3
Rule: Plain Three

clause 1:IBN 017 x1x2x3y1y2 Y
clause 2:IBN 067 x1x2x3y1y2y3 N
clause 3:BI 23456 x0x1x2x3y1y2y3 Y
clause 4:BI 23456 x0x1x2x3y0y1y2 Y
clause 5:SWCA 23456 x0x1x2y0y1y2 Y

This rule describes many images that fall into the class of threes. Once the first clause matches a part of the feature string, all other clauses must follow in order. Clause 2 denotes negative information, it can not match any part of the feature string between clause 1's invocation and clause 3's invocation.

The first clause of a rule tries to define a unique feature of it's class. In this case the first clause is described as:

feature type - Inlet, Bay, or Null
direction - facing towards East (directions 0 1 7)
location - on the right side, in the middle (x-coordinate x1 x2 x3,
y-coordinate y1 y2)
positive information - Yes, these features must be present

The first clause matched feature B0x2y1. Clause 2 is negative so it should not match any features until clause 3 is matched. Clause 3 matches B3x2y3, and the rest of the clauses are also matched. The classifier skips information in the feature strings that is not needed. This enables general rules to be written for varying handwriting styles in the same class.

This method has been developed and tested on handwritten digits. The classifier was trained on 2456 unconstrained handwritten digits and tested on 540 test digits. A correct rate of 85 percent has been achieved with a 9 percent substitution rate. The performance of this method is being improved and it will shortly be evaluated for use in the multi-classifier method for character recognition.

2. Work in Progress

In addition to continued development of the individual methods for character recognition, several other research projects are being pursued. Of particular interest is a technique for automatic rule generation for a syntactic classifier and a method for combining the results of several individual classifiers.

2.1. Automatic Rule Generation for a Syntactic Classifier

The design of a rule base for a syntactic classifier can take many man-months to complete. To design a new set of rules every time a new class is added to the system would not be feasible because of the interaction among different rules. We are developing a technique for the automatic generation of a rule base to avoid these problems.

The algorithm uses rule relaxation with a split and merge technique on two sets of data. The data are called the training and testing sets. A starting rule base is created from all the feature strings in the training set. This gives the training set initially a 100% correct rate. These rules are then relaxed and generalized to give acceptable performance on the testing set. In practice, the starting rule base generally achieves a 3% - 6% correct rate on the testing data.

A first rough clustering is done only using the feature type information. A string edit distance is calculated from each rule to every other rule, creating a complete graph. All rules within distance N of each other are grouped into their own cluster, where N varies from 0 - 6. These clusters are called major clusters.

A rule from each major cluster is chosen to represent that cluster. An attempt is made to merge every other rule into the major cluster. Once a merge is done, the classifier is run on the testing and training data. If performance improves, the merge is conditionally accepted. Before a merge is accepted, the minor clusters are checked to see if a merge with any of them will return better performance. If the performance decreases, a new minor cluster is created for that rule.

As an example of a merge of two rules, consider the previous feature string for the digit three.

1) B5x2y2 S4x1y1 B5x3y0 S4x0y0 A0x3y0 B0x2y1 A0x3y2 S3x0y3 B3x2y3

Another feature string for a three might look like:

2) B1x0y3 B5x3y2 S4x1y1 B4x2y0 S5x0y0 A1x2y0 I0x2y1 A7x3y3 S0x1y2

If the second string is circularly shifted, the similarity of the strings is apparent:

1) B5x2y2 S4x1y1 B5x3y0 S4x0y0 A0x3y0 B0x2y1 A0x3y2 S3x0y3 B3x2y3
2) B5x3y2 S4x1y1 B4x2y0 S5x0y0 A1x2y0 I0x2y1 A7x3y3 S0x1y2 B1x0y3

These two strings can be merged to become:

3) B5x2x3y2 S4x1y1 B45x2x3y0 S45x0y0 A01x2x3y0 BI0x2y1 A07x3y2y3 S03x0x1y2y3 B13x0x2y3

This splitting and merging process is continued until an acceptable level of performance is achieved. The control of the splitting and merging determines the performance that will be achieved. The splitting and merging portion and the other parts of the algorithm are currently under development and have shown promise in several sample domains. We plan to apply it to the generation of rules for handprinted character recognition.

2.2. Classifier Combination

A technique has been developed to combine the results of several classifiers that are applied to word images and rank a dictionary that is assumed to contain the word [2]. The application of this methodology to combining the results of a number of character recognition algorithms is being explored. In this algorithm, each classifier is run on every image and the classifier ranks the classes from most to least likely. The combination algorithm uses this information to arrive at a consensus decision.

The situation that can occur is illustrated in Table 3. Ten classifiers have been applied to 10 images. Overall, there are 1091 classes possible. The entries in the table show the ranking of the correct classes by the classifier in the column. It is seen that in some cases a classifier can perform very poorly and in other cases it can perform quite well. For example, classifier 5 ranks image number 2 as its top choice. However, classifier 5 can only rank image number 5 as choice number 761. This same image is ranked in the top choice by classifier number 7. Thus, the correct choice is available, the problem is to determine it from the data.

The combination technique uses a set of training data to determine the number of decisions that should be used from each classifier. Each time an image is recognized, these decisions are added to a consensus set. The consensus set itself is then ranked and this becomes the output of the method. This technique has been very successful in recognizing highly degraded word images. It is expected that similar success will be obtained for isolated character recognition.

3. Problems Encountered and Challenges Ahead

The strategy of using multiple classifiers has been very useful for difficult recognition problems. Some significant work remains in the development of the individual character classifiers and methods of combining their results.

However, beyond the level of the individual algorithms, significant work remains in expanding our methods to the recognition of any city or state name. Figure 5 shows some typical city and state name

	classifier number									
image number	1	2	3	4	5	6	7	8	9	10
0	102	1	55	1	597	34	393	1	303	19
1	25	4	478	193	498	213	707	45	956	996
2	29	342	2	86	1	941	36	798	448	260
3	2	5	208	76	5	4	16	9	565	183
4	221	90	84	351	259	1038	838	725	819	617
5	570	130	513	33	761	274	1	351	345	105
6	77	40	91	569	11	505	250	271	366	155
7	219	8	356	218	725	72	8	18	139	19
8	41	11	1	2	19	4	16	9	59	44
9	70	82	79	30	338	3	36	26	38	100

Table 3. Ranking of ten word images by ten classifiers.

images. Both cursive and printed text can be present in the same word or the word can be completely cursive. This implies that a more encompassing technique must be developed to recognize such words.

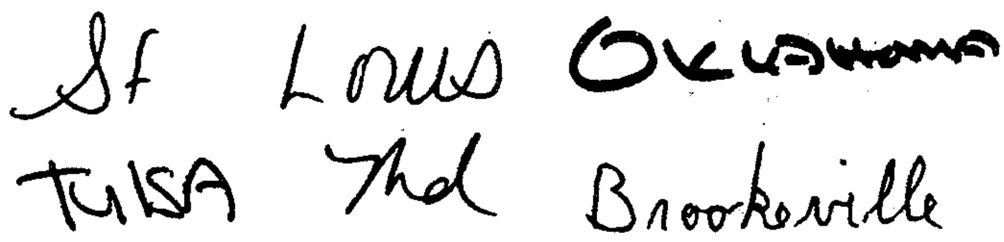

Figure 5. Other styles of city and state names

A project on cursive script word recognition is being pursued to supplement the work on isolated character recognition reported here. The preliminary design of this technique includes the generation of pre-segmentation points followed by the recognition of features or characters along with the confidence in those recognition results. Characters that were reliably recognized serve as anchor points for an island-driving strategy that is coupled to a dictionary of allowable words. This is an attractive strategy in a postal domain since the words that can occur in any particular position are constrained by the context of the address and by other words and numbers that might have been recognized.

Acknowledgments

Dar-Shyang Lee, and John Favata contributed to this paper.

References

1. D. D'Amato, L. Pintsov, H. Koay, D. Stone, J. Tan, K. Tuttle and D. Buck, "High speed pattern recognition system for alphanumeric handprinted characters," *Proceedings of the IEEE Computer Society Conference on Pattern Recognition and Image Processing*, Las Vegas, Nevada, June 14-17, 1982, 165-170.

2. T. Ho, J. J. Hull and S. N. Srihari, "Combination of Structural Classifiers," *submitted to SSPR-90*, New Jersey, June 1990.

3. J. J. Hull, S. N. Srihari, E. Cohen, C. L. Kuan, P. Cullen and P. Palumbo, "A blackboard-based approach to handwritten ZIP Code recognition," *International Conference on Pattern Recognition*, Rome, Italy, November, 1988, 111-113.

4. D. Lee, S. W. Lam and S. N. Srihari, "A structural approach to recognize hand-printed and degraded machine-printed characters," *submitted to the Symposium on Syntactic and Structural Pattern Recognition*, Murray Hill, New Jersey, 1990.

5. S. N. Srihari, E. Cohen, J. J. Hull and L. Kuan, "A system to locate and recognize ZIP Codes in handwritten addresses," *International Journal of Research and Engineering, Postal Applications Inaugural Issue* (1989), 37-56.

Position Statements

Jonathan J. Hull

1. **99.9 percent correct is unnecessary,** (for some problems)
or, A considered judgment should be applied to performance statistics.

At a recent symposium on handwritten ZIP Code recognition, one of the participants requested that a special database of "good" handwritten digits be provided. The purpose of this database was to allow him to report correct rates of greater than 99 percent. He contended that upon publication, his results would receive less respect than they deserved unless they at least matched the results reported by other researchers. Since the digits that occur in a random sample of ZIP Codes were too difficult to reach this goal, one method to achieve matching performance was to adjust the database.

While it is certainly true that removing difficult cases from test data will improve performance, it should not be necessary to report 99 percent correct for others to consider a research project to be successful. The point is that performance statistics cannot be interpreted merely on their face value. That is, an algorithm that is stated to have a correct rate of 99 percent is not necessarily better than one with a correct rate of 95 percent. Only if the problems and the data are *exactly* the same are such comparisons possible.

Very minor differences in even the same data set can have important implications on performance. An example of this occurred at the aforementioned symposium. Two research groups had been provided with the same set of 500 handwritten ZIP Codes and told to apply their segmentation and digit recognition algorithms and to report their digit recognition performance. Each group was to normalize the error rate to 1.5 percent. Research group A achieved 89 percent correct and research group B achieved 86 percent correct. Group A was rightfully proud of their work and was considered at that time to be the "winner". However, group A had manually cropped each ZIP Code from the addresses and group B had used a standard set of coordinates provided to each research group to perform the cropping. The effect was that a small number of the ZIP Codes used by group B contained artifacts that had been removed by group A. These artifacts confused group B's segmentation algorithm and had a negative effect on their recognition performance. When group B removed these artifacts and re-ran their algorithms, their normalized performance improved to 90 percent correct. Thus, group B should have been the winner.

This is just one example of why it is essential that algorithms be evaluated on their merits, the scope of the problem being solved, and the data that was used to derive their performance figures. Often statistics can only be used as a rough guide to the merits of an algorithm. A considered, expert opinion is needed.

2. **Detailed performance comparison can only be done by a third party**
It is often desired to find the best algorithm for a task such as handwritten character recognition. Since "best" is ultimately defined by correct and error rates, one could conduct such a survey by compiling the statistics reported in papers. However, as pointed out above, such a technique is subject to error since even minor differences in experimental design can cause significant differences in performance.

The point of this discussion is that such performance comparison can be difficult and can involve many parameters. At a minimum, the training and testing data should be the same. However, many other factors may have to be considered. Because of such complications, testing should be done by a third party.

Therefore, a quantitative performance comparison is best done by a third party who is given the code for the subject algorithms along with instructions about how to run them. Sufficient opportunity should also be provided for the developers of the algorithms to make minor adjustments for the type of test data that will be used.

The third party should then be able to train and test the algorithms on data that has not been used before by the algorithm developers. A statistical performance comparison could then be obtained. This might be followed up by comments from the algorithm developers and a refinement of the test based on these comments.

An example of such a refinement would be if a certain statistical algorithm required in excess of 20,000 training samples to stabilize its coefficients and some number much less than this had been used in the initial test. This algorithm could be retrained on the larger number of samples and the test could be conducted again. However, this must be done with the knowledge of whether the additional training would be advantageous to the other algorithm. Appropriate action should be considered if this is the case. Perhaps a better approach would be to construct a performance matrix where the values along the axes are the amount of training data for each algorithm and the entries are the performance of both methods. This would allow the tradeoffs of the two approaches to be determined.

After a third party evaluation is completed it might emerge that there is no clear winner because each algorithm has its advantages and disadvantages. For example, one algorithm may recognize broken characters better than another. Such observations can be very important.

3 Multiple algorithms are often preferred

Since different recognition algorithms often have different levels of performance on different types of data, it is desired to achieve the advantages of multiple techniques and to minimize their weaknesses. This could be done by combining several methods in a single algorithm. However, this can be laborious if the methods are not sufficiently similar and the advantage of separate, independent development is lost.

An alternative strategy is to combine results after recognition has been performed. This allows the developers of the component algorithms to work independently on their solutions and maximize performance. If the techniques are sufficiently uncorrelated, the combined results are guaranteed to be better than the individual ones. This is particularly true in domains where the input is nearly unconstrained. The usefulness of multiple, independent algorithms has been observed for problems as diverse as skew correction, character recognition, and holistic word recognition.

The extent of the need for multiple methods is determined by the domain. The recognition of a single, clearly digitized machine printed font can probably be done with high accuracy by one technique. However, as the problem diverges to become more unconstrained, more than one method is desirable.

Biography

Jonathan J. Hull

Department of Computer Science
State University of New York at Buffalo
226 Bell Hall
Buffalo, New York 14260

716-636-3195 (voice)
716-636-3464 (fax)

hull@cs.buffalo.edu (email)

Jonathan J. Hull is a Research Assistant Professor in the Department of Computer Science at the State University of New York at Buffalo. He received a B.A. in Computer Science and Statistics in 1980, an M.S. in Computer Science in 1983, and a Ph.D. in Computer Science in 1988, all from the State University of New York at Buffalo. The topic of his Ph.D. dissertation was "A Computational Theory of Visual Word Recognition." This proposed a word recognition algorithm that was based on how humans read text.

Dr. Hull is an Associate Editor of the Pattern Recognition Journal and a member of the Technical Committee on Applications to Text Processing of the International Association for Pattern Recognition. He has been involved in research on problems of text recognition since 1980 and has focused on problems specific to postal address recognition since 1984. He presently has principal responsibility for research contracts with the United States Postal Service on "Handwritten ZIP Code Recognition," "Contextual Analysis of Address Structures," and "Real-time Handwritten ZIP Code Recognition." These projects all seek to provide future postal Optical Character Readers with advanced capabilities.

Databases

Jonathan J. Hull

The data used in my research are mainly derived from images of postal address blocks that were digitized either with a flatbed scanner at 300 pixels per inch (ppi) and eight bits of grayscale per pixel or with a working postal Optical Character Reader (OCR) at 212 ppi and one bit per pixel. All the images were taken from live mail as it was being processed at a postal facility. Thus, the addresses were prepared without knowledge that they would be incorporated in the databases.

The 300 ppi addresses were scanned at the Buffalo Post Office on two separate eight week occasions in 1986 and 1988. Approximately 5000 complete handwritten addresses and 5000 additional handwritten ZIP Codes were digitized. The samples were balanced by the first digit of the ZIP Code to provide a representation for cities and states across the country. A subset of 1000 of the ZIP Codes were included because they possessed "difficult" characteristics such as touching digits, confusing backgrounds, and so on. Of particular interest are the approximately 30,000 individual handprinted letters that have been segmented from the address blocks.

The 212 ppi addresses were either digitized during 1988 on postal OCRs either in Dallas, Texas, or Washington, D.C. The Dallas address sample consists of approximately 10,000 machine printed addresses that were drawn from different mailstreams such as collection mail, managed mail, and so on. The scanning of these addresses was done over several weeks. The images in this sample represent incoming and outgoing addresses at Dallas. The text in these addresses has been manually entered into separate files to provide for large-scale testing of address recognition algorithms.

The Washington, D.C. sample consists of approximately 40,000 handwritten addresses drawn from the same mailstreams as above. Similarly, these represent the population of incoming and outgoing addresses at Washington. The Washington sar ple has been processed to provide data about the location and truth value of the ZIP Codes. This allows for large-scale testing of ZIP Code recognition techniques.

Theses Supervised and Recent Publications on Text Recognition

Jonathan J. Hull

Theses co-supervised

Allen Commike, *Automatic Learning of Rules for a Structural Classifier*, M.S., 1990.

Tin-Kam Ho, *Multiple-classifier word recognition*, Ph.D., 1991.

Publications

1. J. J. Hull, "Inter-word constraints in visual word recognition," *Proceedings of the Conference of the Canadian Society for Computational Studies of Intelligence*, Montreal, Canada, May 21-23, 1986, 134-138.
2. J. J. Hull, "The use of global context in text recognition," *8th International Conference on Pattern Recognition*, Paris, France, October 28-31, 1986.
3. J. J. Hull and S. N. Srihari, "A computational approach to visual word recognition: hypothesis generation and testing," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Miami Beach, Florida, June 22-26, 1986, 156-161.
4. J. J. Hull, "Hypothesis generation in a computational model for visual word recognition," *IEEE Expert 1*, 3 (Fall, 1986), 63-70.
5. J. J. Hull, "Character recognition: the reading of text by computer," in *The Encyclopedia of Artificial Intelligence*, S. C. Shapiro (editor), John Wiley and Sons, 1987, 82-88.
6. J. J. Hull, "A computational theory and algorithm for fluent reading," *Proceedings of the Third IEEE Conference on Artificial Intelligence Applications*, Kissimmee, Florida, February 23-27, 1987, 176-181.
7. J. J. Hull and S. N. Srihari, "Knowledge utilization in handwritten ZIP code recognition," *Tenth International Joint Conference on Artificial Intelligence*, Milan, Italy, August 23-28, 1987, 848-850.
8. J. J. Hull, "Hypothesis testing in a computational theory of visual word recognition," *Sixth National Conference on Artificial Intelligence* , Seattle, Washington, July 13-17, 1987, 718-722.
9. S. N. Srihari, C. Wang, P. W. Palumbo and J. J. Hull, "Recognizing address blocks on mail pieces: Specialized tools and problem solving architecture," *AI Magazine*, Winter, 1987, 25-40.
10. J. J. Hull, S. N. Srihari, E. Cohen, C. L. Kuan, P. Cullen and P. Palumbo, "A blackboard-based approach to handwritten ZIP Code recognition," *International Conference on Pattern Recognition*, Rome, Italy, November, 1988.
11. J. J. Hull, "Feature selection and language syntax in text recognition," in *From Pixels to Features*, J. C. Simon (editor), North Holland, 1989, 249-260.
12. S. N. Srihari, E. Cohen, J. J. Hull and L. Kuan, "A system to locate and recognize ZIP Codes in handwritten addresses," *International Journal of Research and Engineering, Postal Applications Inaugural Issue* (1989), 37-56.