Pattern Recognition, Vol. 24, No. 5, pp. 421–431, 1991
Printed in Great Britain

0031-3203/91 \$3.00 + .00
Pergamon Press plc

RECOGNITION OF HANDWRITTEN DIGITS USING TEMPLATE AND MODEL MATCHING*

PAUL GADER, BRIAN FORESTER, MARGARET GANZBERGER, ANDREW GILLIES, BRIAN MITCHELL,† MICHAEL WHALEN and TODD YOCUM
Environmental Research Institute of Michigan, P.O. Box 8618, Ann Arbor, MI 48107-8618, U.S.A.

(*Received* 23 *March* 1990; *in revised form* 28 *September* 1990; *received for publication* 19 *October* 1990)

Abstract—A pipeline strategy for handwritten numeral recognition that combines a two-stage template-based technique and a model-based technique is described. The template matcher combines multiple information sources. The second stage of the template matcher was trained on rejects from the first stage. The template matcher classifies 70–80% of the digits with reliability rates over 99%. It also generates class membership hypotheses for the remaining digits which constrain the model-based system. Recognition rates of 94.03–96.39% and error rates of 0.54%–1.05% are obtained on test data consisting of over 13,000 well-segmented digits from ZIP codes in the USPS mail.

Handwritten numeral recognition   Template matching   Model-based classificaton
k-Nearest neighbor   Cascaded systems

1. INTRODUCTION

In this paper, an approach to solving the problem of computer recognition of handwritten numerals is described. The approach yields high recognition rates and low error rates. It is well known that computer recognition of handwritten numerals is a difficult problem.

We demonstrate that simple techniques can be very useful in handwritten numeral recognition when combined with more sophisticated approaches in a pipelined fashion. We also demonstrate that training one stage of a pipelined system on the rejects of the preceding stage can yield higher overall reliability at a lower computational cost. We further demonstrate that using the initial stages of a pipeline classifier as hypothesis generators for the latter stages can also result in higher overall reliability at a lower computational cost. In particular, the development of a two-stage template-matching numeral recognition technique and the integration of it with an existing model-based numeral recognition technique is described. A high-level diagram of the system is shown in Fig. 1.

The template matcher combines multiple information sources, including match strength and k-nearest neighbour measurements from two different metrics. Experiments with training the second stage of the template matcher on the rejects from the first stage are described. It is shown that, on test sets consisting of over 13,000 digits taken from real ZIP codes from the USPS mail, the template matcher together with a moment-based 1 recognizer classifies 70–80% of the digits with reliability rates over 99% and generates class membership hypotheses for the remaining digits. These hypotheses are used to constrain the search space of the model-based system, resulting in increased computational efficiency and high reliability.

Experiments with test data yielded recognition rates of 94.03–96.39% and error rates of 0.54–1.05%. The system was tested on all 6000 digits in the Suen[1] dataset. The recognition rate was 95.13% with 1.05% substitution, and therefore a reliability rate of 98.9% was obtained. When forced to make a decision on the Suen dataset, the system performance was 97.98% correct decisions and 2.02% incorrect decisions.

In the following sections, we discuss the system design and the results of experiments on test data taken from the U.S. mail. In Section 2 we provide an overview of the system and the motivation for developing it. In Section 3, we describe our methodology for digit normalization. In Section 4, we describe the experiments and methodology used in the development of the template matcher. In Section 5, we discuss the overall recognition strategy employed that culminated from the initial template matching experiments and the integration of the template matcher with the digit models. Finally, in Section 6, we present results on test data.

2. SYSTEM OVERVIEW

The initial approach to handwritten numeral recognition developed by our group was model-based.

* This work was supported by the Office of Advanced Technology, United States Postal Service, Contract No. 104230-86-H-0042.
† Current affiliation: Cybernet Systems, Ann Arbor, MI, U.S.A.

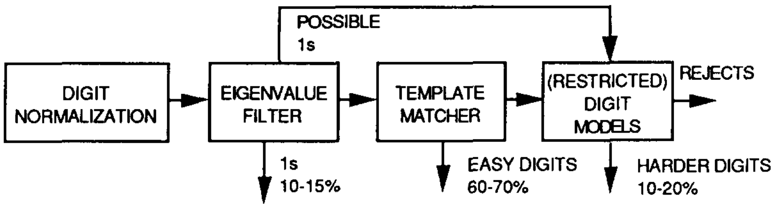

Fig. 1. Digit recognition system overview.

A model-matching language was developed that provided for matching lists of features against predefined syntactic descriptions of digit classes. Thirty-three different digit models were developed to take into account subclasses of digit classes, such as 2s with loops and 2s without loops. These models are described by Gillies and Mitchell.[2]

The models have the property that they *enable particularization*, that is, they enable algorithm developers to develop descriptions of very subtle differences in digits. This property resulted in a digit recognition system that was very reliable. When tested on over 10,000 digits, they averaged recognition rates of 89% with average error of 0.9%.[3] Thus, the reliability of the models, when applied to arbitrary digits, is about 99%. Reliability is measured by computing the number of correct classifications divided by the total number of classifications.

Several problems arose in the development of the models. As more particularizations were included in the models to account for the wide variety in handwritten digits, the amount of processing required to classify easy digits became large. Furthermore, many of the easy digits were being rejected because of the conservative nature of the models and because of the need to distinguish every digit class from every other class. It became difficult to exceed 90% recognition while maintaining less than 1% error. The need to distinguish every digit class from every other class also caused the models to become so complex that it became difficult to understand the interactions between the clauses in the models.

These problems led to the notion of developing a pipelined approach to the digit recognition problem. A computationally simple technique that could be used in front of the digit models was developed. In designing the front end we had two primary goals: (1) to classify easy digits in a highly reliable and computationally simple fashion and (2) to generate hypotheses concerning the digit class membership of digits that could not be reliably classified into a single class.

We chose to experiment with binary template matching on size- and moment-normalized digits. Binary template matching is certainly computationally simple. Our experiments have demonstrated that it can also be used to satisfy our design criteria with respect to classification and hypothesis generation.

Our implementation makes use of two stages of template matching. In this two-stage system, the rejects from the first stage are used to train the second stage of the system. We present experimental results that demonstrate that this approach yields similar recognition rates with lower error rates and with lower computational cost. If the two-stage template matcher cannot make a decision on a given digit, then it generates hypotheses concerning the digit class membership and passes the digit to the digit models. The system can be run in either restricted or non-restricted mode. In restricted mode, only those models indicated by the hypotheses are used in performing the recognition whereas in unrestricted mode, all models are used regardless of the hypotheses. We present experimental results that demonstrate that both modes yield similar recognition rates and that significantly lower error rates are achieved with the system in restricted mode. In other words, use of the hypotheses generated by the template matcher results in higher reliability without a corresponding reduction in recognition rate.

Most of the data used in this project were collected from the United States mail under the auspices of the USPS by the State University of New York (SUNY) at Buffalo. Handwritten mail pieces were scanned using an eight-bit gray-level scanner at 300 dpi and the address blocks manually extracted. The ZIP codes on these address blocks were truthed by hand. The images were preprocessed and binarized as described in Gillies *et al.*[2,3] Digits were then automatically segmented and truthed from the truthed address blocks. A total of over 28,000 handwritten digits were collected in this fashion.

Some of the digits (6000) were also obtained from Professor C. Y. Suen at Concordia University, Montreal, Quebec, Canada, and are those digits that Lam and Suen[1] suggested be used by other researchers in the area. The resolution of these digits is unknown but they are also from the U.S. mail.

3. DIGIT NORMALIZATION

We chose to normalize our digits to a standard size to reduce variation. The height to width ratios of 17,000 digits were computed. Based on these computations, a size of 22×16 was chosen. We also chose to perform normalization that would remove some of the orientation and skewing variation. This alleviates some of the difficulties associated with slanted mail pieces, although the normalization would be sensitive to a large amount of skew. After some initial investigation with different techniques, we decided to use a moment normalization technique based on the one reported in reference (4). This technique scales the digit uniformly in the vertical dimension and scales the digit non-uniformly in the horizontal direction. It uses a system of equations obtained by defining constraints on the second moments of the normalized digit.

In our system, the moment and size normalization steps are combined. The moment normalization, as reported in reference (4), is a linear transformation that depends upon one parameter h. We introduce a second parameter k. By varying these parameters, the size of the output digit can be varied. These parameters are used in the equations defining the constraints on the new digit. Specifically, we require that the second moment of the normalized digit in the x direction is k, that the second moment of the normalized digit in the y direction is h, and that the second cross-moment of the normalized digit is 0. The parameters h and k are used to define a linear transformation of the plane onto itself that transforms the input digit into a normalized digit satisfying the above constraints.

Since a closed form solution was not found for computing both h and k as a function of the desired normalized digit size (22×16), an iterative procedure is used. An alternative approach is to fix the parameters h and k, apply the moment normalization transformation, and then scale the result to the appropriate size. This alternative could result in digits that violate the moment constraints and was therefore not used. The equations we used are similar to those presented by Casey[4] and are therefore not presented. Examples of digits normalized by this technique are shown in Fig. 2.

The normalization significantly distorts the 1s. We investigated the suggested method of measuring the minimum eigenvalue of the moment matrix to identify 1s. We found that the minimum moment eigenvalue, the ratio of the second moment in the x direction to the second moment in the y direction (mxx/myy), and the presence of holes, could be used to define a simple 1-filter that correctly classified about 99–100% of the 1s and incorrectly classified less than 0.5% of the non-1s. These measurements used are readily available from the computations used to perform the moment normalization.

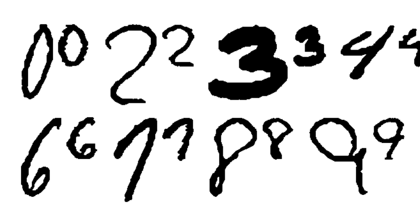

Fig. 2. Some digits before and after size and moment normalization.

The minimum moment eigenvalue measures thickness in the direction of the minor axis. Some thin digits (e.g. 7s) that are not in digit class 1 can have small measurements. Therefore, we label some digits as being close to 1s. These digits are passed directly to the digit models with no template matching performed. This illustrates a feature of the pipeline approach: at each stage there are some digits that can be labeled as too confusing and passed directly to the latter stages of the pipeline for processing by the more sophisticated modules. The algorithm used to classify 1s and possible 1s is:

Algorithm—1-filter
Let mxx and myy be the second moments of the digit.
Let min-eig be the minimum eigenvalue of the moment matrix
Cond:

1. If the digit has holes classify as a non-1
2. If min-eig < thresh1 then
   if the digit has no holes classify as a 1
   otherwise classify as a non-1
3. If min-eig > thresh2 then classify as a non-1
4. If (mxx/myy) < thresh3 then
   if the digit has no holes classify as a 1
   otherwise classify as a non-1
5. If (mxx/myy) > thresh4 then classify as a non-1
6. If all of the above are false, classify as a possible-1.

The Cond statement in the pseudo-code above is the LISP cond, that is, it represents a command to execute the following statements in order until the antecedent of one is true. In that case, execute the consequence and exit from the Cond. The thresholds were chosen by computing the measurements to be thresholded on as set of 17,000 digits and choosing thresholds that would result in zero error.

4. TEMPLATE MATCHER DEVELOPMENT

The template matcher uses the binary digit image data for matching. After initial experimentation with several similarity measures, the Jaccard and Yule similarity measure were chosen for extensive investigation. These similarity measures were compared favorably with several other measures in reference (5). After extensive investigation, a pipelined recognition strategy that involves both these measures and a combination of match strength and k-nearest neighbor information was developed.

In this section, we describe the construction of the templates and the experiments performed to understand the capabilities of template matching for our problem. We then discuss the resulting recognition strategy. Given the context of the recognition strategy, we describe how the rejects of the first stage were used to train the second stage and how the hypothesis generation capabilities of the template matcher are currently used. Finally we discuss some of the experiments that were performed to determine the usefulness of the template matcher in generating hypotheses.

We define the similarity measures. Let T and D be binary arrays representing a template and a digit, respectively. Let n_{ij} = number of times $T(x, y) = i$ and $D(x, y) = j$ for $i, j = 0, 1$. The Jaccard and Yule similarities are defined by:

$$\text{Jaccard: } \frac{n_{11}}{(n_{11} + n_{10} + n_{01})}$$

$$\text{Yule: } \frac{(n_{11}n_{00} - n_{10}n_{01})}{(n_{11}n_{00} + n_{10}n_{01})}.$$

Template construction is performed in two stages: sequential clustering and extraction of a single template from each cluster. Each digit class, except the 1s, is treated separately. Templates are constructed using the Yule measure and the Jaccard measure separately. The sequential clustering is standard: a digit is compared to the comparison digit in each cluster, or bin, and a similarity measure computed. If the highest similarity measure is over a threshold, the digit is placed in that cluster. Otherwise, a new cluster is created and the digit becomes the comparison digit for that cluster. The pseudo-code is as follows:

```
        best_bin = null
highest_similarity = similarity_threshold
Loop for bin in bins
      for bin_number from 1 to length_bin
do
      comparison_digit = first digit in bin
                  sim = similarity (comparison_digit digit)
                If sim>highest_similarity
                  highest_similarity = sim
                      best_bin = bin_number
finally
      Put digit into best_bin
```

The threshold used in the clustering algorithm is referred to as the similarity threshold. By varying it, the number of templates generated can be controlled.

Templates are constructed from the clusters by constructing a new 22×16 binary array from each cluster that had the value 1 at the location (x, y) if and only if at least $P\%$ of the digits in the cluster had the value 1 at that location. Thus, the parameter P controls the thickness of the template strokes. This parameter is referred to as the percentage threshold. Typical values used for P were 45–55. A cutoff parameter was used to control the number of templates and at the same time to eliminate outliers. Thus, if a cutoff of 2 was set, then no cluster with 2 or less digits was used to construct templates.

Our initial experimentation was used to determine relationships between parameters for the clustering algorithm, numbers of templates, and recognition capability. All experiments were performed for both the Yule and the Jaccard similarity measures. Templates were constructed from a data set consisting of 1200 digits from each digit class except 1s. This data set was not used in further testing or training. The templates were used to perform matching on another data set of 500 digits from each class. The numbers of templates used were recorded as were numbers of times the correct digit had the highest similarity value, the second highest, and so forth. The results are summarized in Table 1.

Based on these results, we chose to investigate the Jaccard templates constructed using similarity threshold 0.5 and percentage threshold 0.45 (754 templates) and the Yule templates with similarity threshold 0.88 and percentage threshold 0.45 (854 templates). Later in our study we constructed new Yule templates based on digits rejected by the Jaccard and thereby reduced the number of Yule templates used. The results presented in Table 1 indicate that the template matcher should be a reliable hypothesis generator since the true class membership is included in the top three matches over 99% of the time.

Further experimentation was performed to understand the relationships between recognition rates and error rates for each of these template sets. The experimentation included applying the following rules with various threshold values:

(1) declaring a match if top similarity value obtained from either the Jaccard or the Yule measures was over a threshold;

(2) declaring a match if the digit class for which the top similarity value occurred is the same for both the Jaccard and the Yule measures, and if the Jaccard and Yule values are over thresholds; and

(3) declaring a match if at least k of the ten closest templates to a given digit are all from the same class (k-nearest neighbor).

Table 1. Results of initial template matching experiments. The percentages indicate the percentage of the digits for which the true class membership of the digit was included in the top n matches for $n = 1, 2, 3, 4$

Templates	Number	Top 1	Top 2	Top 3	Top 4
Jaccard	394	93.76	98.49	99.29	99.71
Jaccard	467	94.44	98.31	99.31	99.64
Jaccard	549	94.49	98.38	99.29	99.51
Jaccard	643	94.78	98.33	99.16	99.56
Jaccard	754	95.69	98.60	99.40	99.69
Jaccard	868	95.62	98.51	99.38	99.67
Yule	552	91.98	98.04	99.11	99.53
Yule	687	91.84	97.98	99.22	99.71
Yule	854	93.07	98.22	99.20	99.67
Yule	1018	93.67	98.40	99.38	99.64

The phrase "nearest neighbor" is used somewhat loosely here since we are using similarity measures and not distance measures.

The experiments yielded sets of thresholds that are digit class dependent and that are used in the ultimate digit recognition strategy employed. The thresholds were chosen by computing the values on a set of 4500 digits and setting the thresholds so that the error in each digit class was no more than 0.4%. These experiments also led to the following conclusions:

(1) the Jaccard measure was more reliable as a single source of information;

(2) higher recognition rates could be attained with equivalent error rates if the k-nearest neighbor approach was used than if the top similarity value alone was used; and

(3) higher recognition rates could be attained with equivalent error rates if the Jaccard and Yule measures were used together.

Based on these considerations, we formulated decision strategies that utilized the 10 "nearest" templates to a given digit in both the Yule and Jaccard sense, regardless of the digit class membership of the templates which we shall refer to as the (Yule or Jaccard) top ten list. The decision strategies are described in the next section.

5. OVERALL RECOGNITION SYSTEM

5.1. *Pipeline Recognition Strategy*

The conclusions we listed led us to develop the recognition strategy depicted in Fig. 3. The decision rules indicated are the following:

Decision 1

If the Jaccard top match is over a (digit class dependent) threshold and some percentage (also dependent on the digit class) of the top ten are from the same class as the top match, then classify as that class.

If some number, h, of the Jaccard top ten are from the same digit class and the Jaccard top match is not over a threshold, then classify based on the top ten.

If the Jaccard top match is over its threshold and there are not h of the Jaccard top ten from any one digit class, then classify based on the top match.

Decision 2

If both the Yule top match and Jaccard top match are from the same digit class and are strong enough, classify as that digit class.

If k of the Yule top ten and j of the Jaccard top ten are from the same digit class and the top Yule similarity value is from the same class and high enough, then classify as that class.

If $m > k$ of the Yule top ten and $n > j$ of the Jaccard top ten are from the same digit class and the top Yule similarity value is not high enough, classify according to the top ten class.

Decision 3

If the Jaccard top match is over a threshold and some number p of the Jaccard top ten are from a different digit class, bypass the rest of the template matching process and run the digit models.

The thresholds and percentages of top ten used in the decision making process are digit class dependent. The thresholds and percentages of top ten used when information is available from the Jaccard and Yule matches are generally lower than those used when a decision is made solely on the basis of Yule or Jaccard evidence.

It may be expected that one stage of a classification

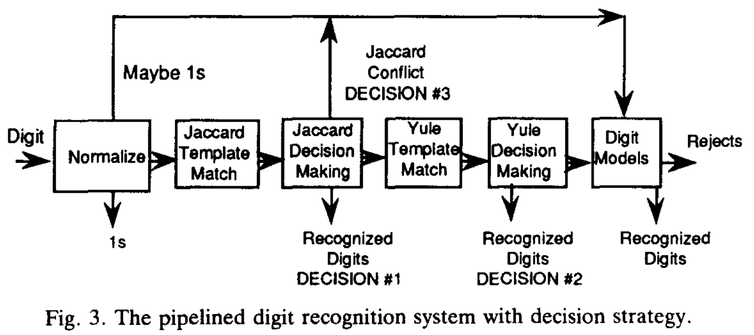

Fig. 3. The pipelined digit recognition system with decision strategy.

pipeline should perform better if it is trained on the rejects from the stage preceding it. We tested this idea in the context of our system. The Jaccard template matching was applied to the data set that the templates were constructed from and those digits that were rejected were collected. The rejects were used as input to the clustering procedure used to construct templates. The threshold values used were the same as initially chosen for the Yules: 0.88 and 0.45. The number of Yule templates originally constructed with these thresholds and cutoff value of 2 was 657, whereas the number of Yule templates constructed from the rejects with these thresholds and cutoff value of 2 was 400. Thus, significantly fewer templates were formed.

As mentioned in Section 2, 33 digit models were constructed for the model-based technique currently at the back end of the pipeline. The system can be run in either of two modes: the unrestricted models mode and the restricted models mode. In the unrestricted mode, if a digit is rejected by both layers of the template matching, then all of the digit models are applied to the digit. In the restricted mode, a list is constructed consisting of all of the digit classes that appear on the Jaccard and Yule top ten lists, that is, all digit classes that are found to be one of the 10 nearest neighbors to the digit in either the Jaccard or the Yule sense. Only the digit models that appear on this list are applied to the digits. The results we have indicate that error rates were cut by a very significant margin, as much as 46%, using the restricted mode.

6. TEST RESULTS

The system was tested on three data sets: the bd data, the bs data and the Suen data. The bd data and bs data were taken from gray-level address block images collected by SUNY and processed as described previously. The bd data set consists of 9397 digits, 4500 of which were used to determine thresholds, the bs data 2219 digits, and the Suen data 6000 digits. Thus, the bs data and the Suen data are pure test data and over 50% of the bd data are pure test data. We point out again that the templates were constructed from entirely different digits.

The system was not optimized for speed. Therefore, the template matching is currently somewhat slow, operating at an average of about 4 per digit on the symbolics. The Jaccard similiarity measure has been implemented at 45 ms on a transputer system, however. Thus, the matcher has real time potential.

The results of running the system in unrestricted mode and a comparison of the results of using the original Yule templates with those of using the reject Yule templates are shown in Table 2. The number of decisions made by each stage of the pipeline is indicated as are the reliability rates at each stage. Reliability is defined to be the number of correct decisions divided by the total number of decisions. Table 3 gives the same information except that the results indicated were obtained by running the system in restricted mode. Confusion matrices for tests run with the Yule templates constructed from Jaccard rejects and with the system in restricted mode are given in the Appendix.

Table 2. Results with and without using Yule templates trained on Jaccard rejects in unrestricted mode

	bd Non-rejects	bd Rejects	bs Non-rejects	bs Rejects
Number of digits	9397	9397	2219	2219
Jaccard templates	754	754	754	754
Yule templates	657	400	657	400
1-Filter decisions	1402	1402	287	287
Reliability	99.6%	99.6%	100%	100%
Jaccard decisions	4431	4431	1101	1101
Reliability	99.71%	99.71%	99.46%	99.46%
Yule decisions	1475	1179	385	307
Reliability	98.31%	99.07	98.7%	99.67%
All template decisions	5906	5600	1486	1408
Reliability	99.4%	99.8%	99.3%	99.5%
Model decisions	1603	1850	359	417
Reliability	96.8%	96.9%	98.3%	98.8%
Recognition rate	93.81%	93.38%	95.31%	94.64%
Reliability	98.9%	99.0%	99.2%	99.4%
Error rate	1.02%	0.93%	0.77%	0.54%

Table 3. Results with and without using Yule templates trained on Jaccard rejects in restricted mode. The Suen data was only run with the reject trained templates and with the system in restricted mode

	bd Non-rejects	bd Rejects	bs Non-rejects	bs Rejects	Suen Rejects
Number of digits	9397	9397	2219	2219	6000
Jaccard templates	754	754	754	754	754
Yule templates	657	400	657	400	400
1-Filter decisions Reliability	1407 99.29%	1409 99.15%	287 100%	287 100%	605 98.51%
Jaccard decisions Reliability	4376 99.73%	4359 99.68%	1049 99.62%	1049 99.62%	2961 99.63%
Yule decisions Reliability	1394 98.78%	1159 99.05%	408 98.53%	320 99.38%	687 98.71%
All template decisions Reliability	5770 99.5%	5518 99.55%	1437 99.3%	1369 99.6%	3648 99.5%
Model decisions Reliability	1737 99.3%	1960 99.3%	374 98.9%	445 99%	1464 97.7%
Recognition rate Reliability Error rate	94.31% 99.4% 0.55%	94.03% 99.4% 0.54%	94.82% 99.3% 0.63%	94.28% 99.6% 0.41%	95.13% 98.9% 1.05%

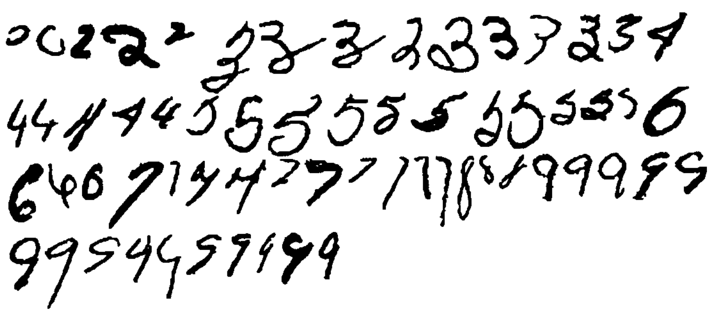

Fig. 4. All of the digits misclassified with the system in forced recognition mode and applied to 6000 test digits.

Table 4 shows the results of forcing recognition on the data sets using the system in restricted mode. Recognition is forced in the following way: if a digit passes through the entire pipeline without being recognized, then it is assigned to the digit class associated with the highest similarity value obtained from the Jaccard matcher. Confusion matrices for forced recognition are also given in the Appendix.

Table 4. Results of forcing the system to make a recognition decision

	Recognition	error
bd	97.95%	2.05%
bs	98.69%	1.31%
Suen	97.98%	2.02%

The digits that were misclassified by the system using forced recognition are shown in Fig. 4.

The classifications returned by the system are shown in Table 5. The numbers in the table are in one-to-one correspondence with the digits in Fig. 4 and indicate the erroneous system response to those digits.

7. CONCLUSION

We have described a pipelined system for handwritten digit recognition that achieves high recognition rates and low error rates on test sets consisting of over 13,000 digits and from different scanners. We have discussed experiments that demonstrate that training a stage of the system on the rejects from the preceding stage can yield higher reliability without a corresponding loss in recognition

Table 5. The system responses to the digits in Fig. 4. The numbers in the table are in one-to-one correspondence with the digits in Fig. 4 and indicate the erroneous system response to those digits

2 2 7 8 8 8 8 8 2 2 8 7 2 5 9
9 5 8 9 9 0 6 3 0 8 6 2 0 2 2 3 0
0 4 0 9 1 4 4 2 9 1 1 1 1 1 1 1 1 7 7 8 8 5
8 7 5 4 5 5 7 1 8 4

performance. We have also demonstrated that using the early stages of the system as hypothesis generators for the later stages can also yield higher reliability without a corresponding loss in recognition performance.

SUMMARY

A pipeline strategy for handwritten numeral recognition is described. In particular, the development of a two-stage template-matching numeral recognition technique and the integration of it with an existing model-based numeral recognition technique is described. The template matcher combines multiple information sources, including match strength and k-nearest neighbor measurements from two different metrics. Experiments with training the second stage of the template matcher on the rejects from the first stage are described. It is shown that, on test sets consisting of over 13,000 well-segmented digits taken from real ZIP codes from the USPS mail, the template matcher together with a moment-based 1 recognizer, classifies 70–80% of the digits with reliability rates over 99% and generates class membership hypotheses for the remaining digits. These hypotheses are used to constrain the search space of the model-based system resulting in increased computational efficiency and high reliability. Experiments with test data yielded recognition rates of 94.03–96.39% and error rates of 0.54–1.05%. In particular, the system was tested on all 6000 digits in the Suen[1] dataset. The recognition rate was 95.13% with 1.05% substitution, and therefore a reliability rate of 98.9% was obtained. When forced to make a decision on the Suen dataset, the system performance was 97.98% correct decisions and 2.02% incorrect decisions.

Acknowledgements—This work was supported by the Office of Advanced Technology, United States Postal Service, Contract No. 104230-86-H-0042. We would like to thank Dr Gary Herring and Mr Carl O'Conner of the Office of Advanced Technology, and Dr John Tan of Arthur D. Little, Inc. for their support.

REFERENCES

1. L. Lam and C. Y. Suen, Structural classification and relaxation matching of totally unconstrained handwritten zip-code numbers, *Pattern Recognition* **21**, 19–31 (1988).
2. A. Gillies and B. Mitchell, A model-based approach to handwritten digit recognition, *Mach. Vision Applications* **2**, 231–243 (1989).
3. B. Mitchell, A. Gillies, M. Whalen and P. Gader, A model-based computer vision system for the recognition of handwritten address ZIP codes, *Final Technical Report Draft, USPS Office of Advanced Technology* (1989).
4. R. G. Casey, Moment normalization of handprinted characters, *IBM J. Res. Dev.*, p. 548 (1970).
5. J. D. Tubbs, A note on binary template matching, *Pattern Recognition* **22**, 359–365 (1989).

About the Author—PAUL GADER received his Ph.D. in Applied Mathematics from the University of Florida in August 1986. Since then, Dr Gader has held positions as Senior Research Scientist at Honeywell Systems and Research Center and as Assistant Professor of Mathematics at the University of Wisconsin – Oshkosh. He is currently a research engineer at the Environmental Research Institute of Michigan. He has performed research in applied mathematics, image algebra, obstacle detection, scene analysis, automatic target recognition, and, most recently, recognition of handwritten ZIP codes. His research interests are in the areas of computer vision, pattern recognition and applied mathematics.

About the Author—BRIAN D. FORESTER obtained a B.S. degree in Computer Science from Eastern Michigan University in 1989. He is a research engineer at ERIM. His research interests include artificial intelligence, image processing and character recognition.

About the Author—MARGARET J. GANZBERGER is completing a Bachelor of Science degree in Computer Science and Mathematics at the University of Michigan – Dearborn. She has been a research assistant at the Document Reading Systems Center at ERIM since January 1989. Her research interests include artificial intelligence, image processing and character recognition.

About the Author—ANDREW GILLIES recieved the B.S.(Hon.) degree, Electrical Engineering, in 1975 from Carnegie-Mellon University, Pittsburg, and the Ph.D. degree, Computer and Communication Sciences, in 1985 from the University of Michigan, Ann Arbor. He has worked since 1975 at ERIM on projects including synthetic aperture radar image formation, automatic target detection and classification, genetic algorithms, mathematical morphology, parallel computer algorithms, handwritten character recognition, and most recently, contextual analysis of address block images. His current research interests include image analysis, artificial intelligence, pattern recognition and machine learning.

About the Author—BRIAN T. MITCHELL obtained his B.S. degree in Mathematics (1973), M.S. degree in Statistics (1976), M.S. degree in Computer Science (1977) and Ph.D. degree in Computer Science (1983) from Western Michigan University. Over the past decade he has held a wide variety of computing positions in both academia and industry. He has served as Assistant Director of the Academic Computer Center and Assistant Professor of Computer Science at WMU, Chief of the Artificial Intelligence Group at PAR Technology, Principal Scientist of Artificial Intelligence at General Dynamics, and Director of the Document Reading Systems Center at ERIM. Dr Mitchell is currently the Vice-President and Director of Technology Development at Cybernet Systems Corp. in Ann Arbor, MI. His current research interests include: image understanding, image processing, artificial intelligence, reasoning under uncertainty, machine learning, planning and robotic applications.

About the Author—MICHAEL WHALEN obtained his B.S. degree with distinction in Electrical Engineering in 1986 from Wayne State University. He received his Masters degree from the University of Michigan in 1990, majoring in Signal Processing. Mr Whalen is currently a Research Engineer at the Environmental Research Institute of Michigan in Ann Arbor, MI. His current research interests lie in the areas of signal/image processing, robotics, machine learning, artificial intelligence and parallel processing.

About the Author—TODD YOCUM is completing a Bachelor of Science degree in Computer Science, Mathematics, and Electrical Engineering at Michigan State University. He has been a research assistant at the Document Reading Systems Center at ERIM since January 1990. His research interests include artificial intelligence, image processing, character recognition, secure computing and parallel processing.

Table S1. Overall Report: System Performance on Suen data

Input class	Digits number	Recognition output 0	1	2	3	4	5	6	7	8	9	Error	Recog. rate
0	600	591	0	1	0	0	0	1	0	0	0	0.003	0.985
1	600	0	598	0	0	0	0	0	0	0	0	0.000	0.997
2	600	0	0	583	0	0	0	0	0	3	0	0.005	0.972
3	600	0	0	3	574	0	1	0	1	4	0	0.015	0.957
4	600	0	0	0	0	572	1	0	0	1	4	0.010	0.953
5	600	3	0	3	2	0	522	2	0	1	0	0.018	0.870
6	600	3	0	0	0	1	0	582	0	0	0	0.007	0.970
7	600	0	5	1	0	2	0	0	573	0	2	0.017	0.955
8	600	0	3	0	0	0	0	0	0	572	0	0.005	0.953
9	600	0	1	0	0	2	4	0	4	5	541	0.025	0.902
Total Number of digits:												6000	
Number of digits erroneously recognized:												63	1.05%
Number of digits rejected:												229	3.82%
Number of digits correctly recognized:												5708	95.13%
Reliability													98.91%

Table F1. Overall Report: Forced Recognition on Suen data

Input class	Digits number	Recognition output 0	1	2	3	4	5	6	7	8	9	Error	Recog. rate
0	600	595	0	3	0	0	0	2	0	0	0	0.008	0.992
1	600	0	598	0	0	1	0	0	0	1	0	0·003	0.997
2	600	0	0	594	0	0	0	1	1	4	0	0.010	0.990
3	600	0	0	5	582	1	3	0	2	7	0	0.030	0.970
4	600	0	0	2	0	585	1	0	1	3	8	0.025	0.975
5	600	3	0	3	8	1	578	0	0	3	0	0.037	0.963
6	600	3	0	0	0	1	0	596	0	0	0	0.007	0.993
7	600	2	5	1	0	2	0	0	584	0	6	0.027	0.973
9	600	0	3	1	5	0	2	0	0	589	0	0.018	0.982
9	600	1	1	0	1	3	4	0	7	5	578	0.037	0.963
Total number of digits												6000	
Number of digits erroneously recognized												121	2.02%
Number of digits rejected												0	0.00%
Number of digits correctly recognized												5879	97.98%
Reliability													97.98%

Table S2. Overall Report: System Performance on bd data

Input class	Digits number	0	1	2	3	4	5	6	7	8	9	Error	Recog. rate
		Recognition output											
0	1428	1378	6	0	0	0	0	2	0	1	0	0.006	0.965
1	1417	0	1402	0	0	0	0	0	3	0	0	0.002	0.989
2	1011	2	0	955	0	0	0	0	1	1	0	0.004	0.945
3	825	0	0	3	764	0	1	0	0	1	0	0.006	0.926
4	857	0	0	0	0	784	0	0	2	1	1	0.005	0.915
5	751	2	0	0	1	0	662	1	0	1	0	0.007	0.881
6	820	0	1	0	0	0	0	786	0	0	0	0.001	0.959
7	801	0	4	1	0	0	0	0	756	0	1	0.007	0.944
8	728	3	1	0	1	0	0	2	0	667	0	0.010	0.916
9	759	1	1	0	1	1	0	0	3	0	682	0.009	0.899

Total number of digits	9397	
Number of digits erroneously recognized	51	0.54%
Number of digits rejected	510	5.43%
Number of digits correctly recognized	8836	94.03%
Reliability		99.43%

Table F2. Overall Report: Forced Recognition on bd data

Input class	Digits number	0	1	2	3	4	5	6	7	8	9	Error	Recog. rate
		Recognition output											
0	1428	1403	6	1	3	1	5	6	0	2	1	0.018	0.982
1	1417	2	1402	3	0	3	1	0	3	2	1	0.011	0.989
2	1011	5	0	995	1	0	1	2	3	2	2	0.016	0.984
3	825	1	0	5	804	0	5	0	2	3	5	0.025	0.975
4	857	1	0	3	1	834	0	6	3	3	6	0.027	0.973
5	751	3	0	1	3	0	729	7	2	6	0	0.029	0.971
6	820	2	1	2	0	0	0	813	0	2	0	0.009	0.991
7	801	0	4	2	2	6	0	0	778	1	8	0.029	0.971
8	728	4	1	1	3	0	5	4	0	708	2	0.027	0.973
9	759	1	1	0	3	8	1	0	6	1	738	0.028	0.972

Total number of digits	9397	
Number of digits erroneously recognized	193	2.05%
Number of digits rejected	0	0.00%
Number of digits correctly recognized	9204	97.95%
Reliability		97.95%

Table S3. Overall Report: System Performance on bs data

Input class	Digits number	0	1	2	3	4	5	6	7	8	9	Error	Recog. rate
		Recognition output											
0	354	344	0	0	0	0	0	0	0	1	0	0.003	0.972
1	290	0	288	0	0	0	0	0	0	0	0	0.000	0.993
2	224	1	0	215	0	0	0	0	0	1	0	0.009	0.960
3	210	0	0	0	199	0	2	0	0	2	0	0.019	0.948
4	184	0	0	0	0	176	0	0	1	0	0	0.005	0.957
5	117	0	0	0	0	0	111	0	0	0	0	0.000	0.949
6	246	1	0	0	0	0	0	235	0	1	0	0.008	0.955
7	222	0	0	0	0	1	0	0	212	0	0	0.005	0.955
8	192	0	0	0	0	1	0	0	0	185	0	0.005	0.964
9	180	0	0	0	0	0	0	0	0	0	174	0.000	0.967

Total number of digits	2219	
Number of digits erroneously recognized	12	0.54%
Number of digits rejected	68	3.06%
Number of digits correctly recognized	2139	96.39%
Reliability		99.44%

APPENDIX: CONFUSION MATRICES

In this Appendix, full confusion matrices associated with applying our system to the Suen, bd, and bs data sets are given. There are two sets of confusion matrices: the first set of confusion matrices resulted from applying our system in restricted mode and with the Yule templates constructed from the Jaccard rejects. This set is labeled as System Performance tables. The second set of confusion matrices resulted from forcing a recognition decision on those digits that are rejected by our system. The recognition is forced by classifying a rejected digit in the class associated with the highest Jaccard similarity value obtained for that digit. This set is labeled as Forced Recognition tables.

Table F3. Overall Report: Forced Recognition on bs data

Input class	Digits number	Recognition output 0	1	2	3	4	5	6	7	8	9	Error	Recog. rate
0	354	352	0	0	0	0	0	1	0	1	0	0.006	0.994
1	290	0	288	1	0	0	0	0	0	1	0	0.007	0.993
2	224	1	0	220	0	0	0	0	0	3	0	0.018	0.982
3	210	1	0	0	203	0	3	0	0	3	0	0.033	0.967
4	184	0	0	0	0	183	0	0	1	0	0	0.005	0.995
5	117	0	0	0	0	0	116	0	0	1	0	0.009	0.991
6	246	2	0	0	0	1	0	242	0	1	0	0.016	0.984
7	222	0	0	1	0	1	0	0	218	0	2	0.018	0.982
8	192	0	0	0	2	1	1	0	0	188	0	0.021	0.979
9	180	0	0	0	0	0	0	0	0	0	180	0.000	1.000

Total number of digits	2219	
Number of digits erroneously recognized	29	1.31%
Number of digits rejected	0	0.00%
Number of digits correctly recognized	2190	98.69%
Reliability		98.69%