An Integrated Architecture for Recognition of Totally Unconstrained Handwritten Numerals

Amar Gupta
M.V. Nagendraprasad
A. Liu
Patrick Wang
S. Ayyadurai

WP #3765 January 1993
PROFIT #93-04

Productivity From Information Technology
"PROFIT" Research Initiative
Sloan School of Management
Massachusetts Institute of Technology
Cambridge, MA 02139 USA
(617)253-8584
Fax: (617)258-7579

 The research described herein has been supported (in whole or in part) by the Productivity From Information Technology (PROFIT) Research Initiative at MIT.

EXECUTIVE OVERVIEW

Financial enterprises rely heavily on paper-based documents to conduct various operations; this is true both for external operations involving customers and other financial institutions, as well as internal operations involving various departments.

Researchers at MIT have looked at the possibility of taking information directly from paper documents, especially handwritten documents, to computer-accessible media. Automated reading involves several steps as follows:

(i) Scanning of document;
(ii) Location of area to be "read";
(iii) Decomposing the selected area into separate characters;
(iv) Adjusting size and slant of each character;
(v) Recognizing each character; and
(vi) Testing whether input has been correctly read.

Based on several years of sustained research, the researchers have attained very high "reading" speed and accuracy, even in situations where the quality of the input material is poor. Patent rights for some of the new techniques have been applied for. Sponsor companies are eligible to test the new techniques in their respective environments at no charge.

The work performed so far is described in a number of published paper and working papers. The list of working papers is as follows:

IFSRC # 107-89	Optical Image Scanners and Character Recognition Devices: A Survey and New Taxonomy	Amar Gupta Sanjay Hazarika Maher Kallel Pankaj Srivastava
IFSRC # 123-90R	An Improved Structural Technique for Automated Recognition of Handprinted Symbols Revised October 1990	Patrick S. P. Wang Amar Gupta
IFSRC # 124-90	Integration of Traditional Imaging, Expert Systems, and Neural Network Techniques for Enhanced Recognition of Handwritten Information	Evelyn Roman Amar Gupta John Riordan
IFSRC # 151-91	Handwritten Numeral Recognition Using Dynamic Programming Neural Networks on an Off-Line Basis	Ronjon Nag Alexis Lui Amar Gupta
IFSRC # 162-91R PROFIT 93-03	Algorithms for Thinning and Rethickening Binary Digital Patterns	M. Nagendraprasad Patrick S. Wang Amar Gupta
IFSRC # 173-91	A New Algorithm for Slant Correction of	Vanessa C. Feliberti

	Handwritten Characters	Amar Gupta
IFSRC # 214-92	An Algorithm for Segmenting Handwritten Numeral Strings	Peter L. Sparks M. V. Nagendraprasad Amar Gupta
IFSRC # 215-92	A New Algorithm for Correcting Slant in Handwritten Numerals	M. V. Nagendraprasad Amar Gupta Vanessa Feliberti
IFSRC # 218-92	A System for Automatic Recognition of Totally Unconstrained Handwritten Numerals	M. V. Nagendraprasad
IFSRC # 219-92	A Collection of Papers on Handwritten Numeral Recognition	Amar Gupta
IFSRC # 261-93	An Adaptive Modular Neural Network with Application to Unconstrained Character Recognition	Lik Mui Arun Agarwal P. S. P. Wang
IFSRC # 287-94 PROFIT 93-04	An Integrated Architecture for Recognition of Totally Unconstrained Handwritten Numerals	Amar Gupta M. V. Nagendraprasad A. Liu Amar Gupta S. Ayyadurai
IFSRC # 288-94 PROFIT 93-09	Detection of Courtesy Amount Block on Bank Checks	Arun Agarwal Len M. Granowetter Amar Gupta P. S. P. Wang
IFSRC # 289-94 PROFIT 94-14	A Knowledge Based Segmentation Algorithm For Enhanced Recognition of Handwritten Courtesy Amounts	Karim Hussein Amar Gupta Arun Agarwal Patrick Shen-Pei Wang

The research has been funded by a number of organizations, via the International Financial Services Research Center (IFSRC) and the Productivity from Information Technology (PROFIT) Initiative. Individuals in such sponsor companies should contact their designated contact person at MIT to receive copies of the papers, and the software developed at MIT.

The Principal Investigator for the imaging area is Dr. Amar Gupta, Co-Director, "PROFIT" Initiative, MIT Sloan School of Management, Room E53-311, 77 Massachusetts Avenue, Cambridge, MA 02139, USA; Telephone: (617)253-8906; Fax: (617)258-7579; e-mail: agupta@mit.edu. Your comments and suggestions are encouraged.

AN INTEGRATED ARCHITECTURE FOR RECOGNITION OF TOTALLY UNCONSTRAINED HANDWRITTEN NUMERALS

AMAR GUPTA, M. V. NAGENDRAPRASAD*, A. LIU,
P. S. P. WANG† and S. AYYADURAI
Massachusetts Institute of Technology, Cambridge, MA 02142, USA

A multi-staged system for off-line handwritten numeral recognition is presented here. After scanning, the digitized binary bitmap image of the source document is passed through a preprocessing stage which performs segmentation, thinning and rethickening, normalization, and slant correction. The recognizer is a three-layered neural net trained with back-propagation algorithm. While a few systems that use three-layered nets for recognition have been presented in the literature, the contribution of our system is based on two aspects: elaborate preprocessing based on structural pattern recognition methods combined with a neural net based recognizer; and integration of neural net based and structural pattern recognition methods to produce high accuracies.

Keywords: Numeral recognition, integrated architecture, neural networks, skeletonization (thinning).

1. INTRODUCTION

With the proliferation of computers and their integration into activities of daily existence, one formidable problem is to build computers that can recognize handwritten material without human intervention. During the last twenty years, a number of schemes for handwriting recognition have been proposed.[1–9] The sustained effort is justified not only by the possibility of superior man–machine interaction but also by the potential for automating many mundane tasks like postal ZIP Code reading, check verification, and preservation of the contents of old and valuable handwritten texts and notebooks.

Most of the work in this field, until recently, has focused on syntactic and structural pattern recognition techniques. However, the discovery of training methods for multi-layer neural networks has motivated new approaches for handwriting recognition systems.[4,10,11] An excellent list of relevant papers can be found in Ref. 10. Even before the "back-propagation age" there have been attempts at neuronal approaches. Fukushima's neogonitron,[12] for example, is a multi-layered system whose architecture was based on the cat's visual system. Neocognitron was reasonably successful but it employed slow and cumbersome training methods.

*Present address: Dept. of Computer Science, University of Massachusetts at Amherst, Amherst, MA 01003, USA.
†Permanent address: College of Computer Science, Northeastern University, Boston, MA 02115, USA.

International Journal of Pattern Recognition and Artificial Intelligence Vol. 7 No. 4 (1993) 757–773

2. THE PROBLEM

The handwriting recognition problem is essentially a classification problem. The biggest challenge lies in dealing with the infinite variety in handwriting styles. Just as no two persons have the same fingerprints, no two persons have the same handwriting. Even for a particular person, the handwriting varies depending on his or her mental disposition.

The field of handwriting recognition can be subdivided into two subfields: alphabet recognition and numeral recognition. Among these two, the numeral recognition problem appears to be the easier one due to the fewer number of classes involved. However, numeral recognition has issues peculiar to itself. For example, while most of the connections in an alphabetic string are intentional, in the sense that the writer has a particular style of connecting adjacent letters, all the connections are unintentional in the numeral scenario. Since the connections between adjacent numerals are accidental in nature, these connections are random, and require a greater effort to separate as compared to separation of alphabetic text.

Many recognition schemes have been presented that assume alphabets and digits are pre-segmented. Handwritten input usually occurs as connected or partially connected strings of characters, and the first stage in the recognition procedure usually involves "chopping" the strings into locally separate entities. This process is known as segmentation. Recognition is then performed using these entities. Since segmentation is a hard problem, the assumption of pre-segmented inputs skirts one of the major problems in automated recognition of handwritten material.

There are, basically, two forms of handwriting recognition models: on-line and off-line. On-line methods require that the user write the script on a special transducer tablet. This tablet is connected to a computer and the handwritten information is extracted "on-line", as the individual writes successive letters. Since the information is extracted on-line, explicit "stroke sequence" is available as each letter is written. The availability of the stroke sequence greatly simplifies the overall problem.[1,3,9,13,14]

The off-line situation is more complicated. Here the input is presented after it has been written on a medium, typically paper. The input is scanned using a digital scanner and the processing is done using this scanned material. Note that no explicit stroke sequence is available in this scenario.[2,4,11,12,15,16]

We investigated various approaches to automated recognition of handwritten numerals in the specific context of recognition of courtesy amounts on bank checks. This involves recognition of unconstrained handwritten numerals in an off-line manner with very high accuracy. The architecture of our prototype system and the salient features of various modules are discussed in the following sections of this paper.

3. ARCHITECTURE OF THE RECOGNITION SYSTEM

In view of the specific objective of recognition of the courtesy dollar amounts on bank checks, the system design was influenced by the following considerations:

- The system must be able to deal with the diverse handwriting styles of various customers who write checks. The infinite variety of styles, which vary across regions and even within a region, must be taken into consideration.
- The time required for the system to perform recognition must be very small. A typical human operator in the existing banking system takes anywhere between five to ten seconds to process a check, and the automated system must offer equivalent or better speeds. Thus each module of the system has to be optimized to achieve a high recognition speed.

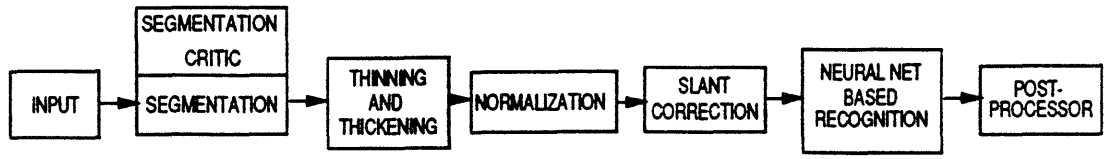


Fig. 1. Architecture for handwritten numeral recognition system.

The components of the system, shown in Fig. 1, are described in detail in the following sections.

3.1. Input and Preliminary Preprocessing

The input to the system is obtained by scanning the numerals of the check which is placed at a specified location on the scanner. This raster image of the numerals and its immediate neighborhood are then converted into a binary bitmap of two values – "1" for a pixel that is "on" and "0" for a pixel that is "off". Most of the checks contain a box or a line where the amount is supposed to be written in numerical format. As such, the input bitmap is passed through a box-removal program which eliminates the box surrounding the numerals. Finally, any noisy areas in the bitmap are eliminated.

3.2. Segmentation

Most of the algorithms discussed in the literature for segmenting unconstrained numeral strings are intended for ZIP Code string segmentation.[2,5,17,18] In such a case, there is an *a priori* knowledge of the number of digits in the string. However, in banking applications, this vital piece of information is missing. As such, we developed an algorithm for segmenting unconstrained numeral strings with an unknown number of digits in them. This algorithm[19] was specifically designed to function independently of the size of the numerals. Unlike ZIP Code recognition and other applications in which all the numerals in the string are of almost the same size, the dollar component and the cents component in our target application usually very significantly from each other in size.

The segmentation stage comprises a "blackboard-like" segmentation system.[a] The system consists of two major components – a segmentation critic which represents the control component[20] of the system and the set of segmentation modules which represents the knowledge sources.[20] of the system. The common memory, also called the blackboard,[20] is the place where the results of the segmentation process are posted. All the knowledge sources and the segmentation critic have access to this memory. The organization of our blackboard is relatively flat, unlike most systems in literature where a hierarchical organization is used. This is due to the nature of our knowledge sources, which we describe here.

Before discussing the three knowledge sources in our architecture, and the underlying algorithms, we present two terms[6] which will be useful for discussing the algorithms. The *top profile* of a bitmap is the contour which represents the collection of the pixels of the bitmap that can be seen by an observer viewing the bitmap from its top. The *bottom profile* is similarly defined except that the observer now views it from the bottom. Figure 2 shows an example of a connected pair of numerals and their profiles.

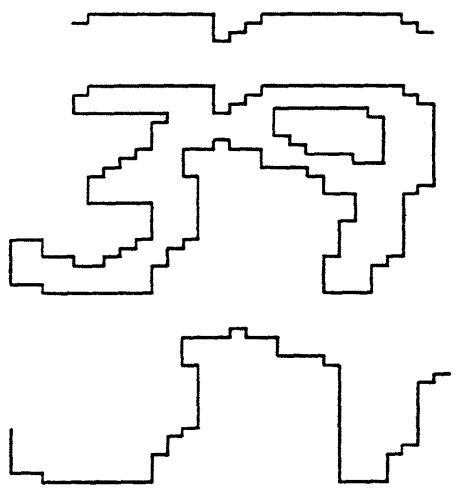


Fig. 2. Top and bottom profiles for a connected pair of numerals.

The three knowledge sources operate as described below:

(a) Connected Component Extraction: Here the array is scanned and the connected components in the bitmap are separated out. This is done by a recursive graph tracing algorithm which looks at each pixel and its immediate neighborhood; all pixels that are "1" in this neighborhood are assumed to belong to the same component as the present pixel. The algorithm is recursively called on each of the 1_pixels of the neighborhood. Each of the connected components extracted from the bitmap is placed in a common memory area.

[a]We call our system "blackboard-like" rather than "blackboard" because our control (represented by a segmentation critic) is not truly opportunistic in the conventional sense of the term.[20] Besides, the blackboard organization is relatively flat in our case.

(b) Upper-Lower Contour Decomposition: The second pass, derived from the algorithm in Ref. 6, utilizes the upper and lower profiles of the numeral to locate the best co-ordinates from which to perform the decomposition. The maximum point on the lower profile and the minimum point on the upper contour are identified as the best points for performing the split. The search for the maximum and the minimum begins at the center and extends to a fraction of the length of the component on either side of the center. A straight line is drawn between the two points and two vertical lines are drawn connecting the minimum point on the top profile and the maximum point on the bottom profile to the respective top/bottom edges of the component bitmap. Again the output is stored in the common memory area.

(c) Hit and Deflect Strategy (HDS): Based on the ideas presented in Ref. 17, the goal of the HDS is to steadily build a dividing line by moving through the space between the two characters, bouncing off the contour and deferring intersection with them as long as possible. When there is no alternative but to cut through the contour lines, the cut is performed.

Each of the knowledge sources places its results in a common memory area for the next source to be triggered, based on these results. The segmentation critic decides the next knowledge source to be triggered. It makes its decisions based mostly on the gross structural features of the input bitmaps. If the aspect ratio of a bitmap exceeds a certain threshold, it recycles the input data for further segmentation. Heuristics based on relative weights and heights of the segmented pieces are employed to check for the validity of a cut. Heuristics concerning other properties such as the nature of the cuts are also utilized.

The system described above is flexible in the sense that it is easy to incorporate new algorithms which could be triggered strictly on an as-needed basis. While each of the knowledge sources is adapted from existing algorithms, we believe that a flexible architecture like ours is needed to integrate a variety of algorithms to maximally exploit the advantages specific to each of them. Besides, this flexibility also helps in tackling uncertainties inherent in the task on our hands. Unlike the algorithms related to postal ZIP Code segmentation, our algorithm does not assume any *a priori* knowledge of the number of digits in the string. The segmentation critic contains heuristics to "guess" when it needs to stop segmenting the string further. These heuristics are based on statistical studies conducted on the National Institute of Standards and Technology (NIST) Handprinted Numerals Database. Efforts are continuing to make the segmentation critic more knowledge intensive.

Figure 3 shows some of the strings successfully segmented by the system.

3.3. Thinning and Rethickening

The recognition stage is neural network based. A three-layered neural network was trained over a large set of training patterns to cover a broad spectrum of variations in the numeral patterns. However, if the neural net is required to deal with fewer variations in the numeral patterns, both training and recognition processes will be

Fig. 3. Examples of successful segmentations.

more efficient. Details of the advantages of this stage and the slant correction stage are discussed later in this paper.

The system must cater to inputs written with diverse kinds of pens, including felt pens, ball pens, and micro-tipped pens. Each type of pen creates characters of a different thickness. In order to reduce these variations, the thickness of the input pattern is first reduced and then systematically increased to a uniform thickness. Before we describe our skeletonization algorithm,[21] we define several terms.

The eight neighbors of a pixel $p : [i, j]$ are $p[k]$, $k = 0, \ldots, 7$, $p[8] = p[0]$, as shown in Fig. 4. A contour point of a bitmap is a 1_pixel that has at least one neighbor that is white.

Let $A(p)$ be the number of white-to-dark transitions during a clockwise traversal in the neighborhood of pixel p and let $B(p)$ be the number of neighbors of p which are not equal to zero.

$$C(p) = \begin{cases} 1 & \text{if } p[k-1] + p[k] + p[k+1] + p[k+4] = 0 \\ & \text{and } p[k+3] = 1 \text{ and } p[k+5] = 1 \\ & \qquad k = 1 \text{ or } k = 3 \\ 0 & \text{otherwise} \end{cases}$$

The function *contour(p)* tests whether a point is lying on a contour or not; it looks for a white space in the neighborhood of the point.

p[7]: [i-1, j-1]	p[0]: [i-1, j]	p[1]: [i-1, j+1]
p[6]: [i, j-1]	p: [i, j]	p[2]: [i, j+1]
p[5]: [i+1, j-1]	p[4]: [i+1, j]	p[3]: [i+1, j+1]

Fig. 4. A pixel and its neighbors.

The function *deletion(Q, p)* deletes a pixel p in the array Q.

Given the above functions and definitions, the algorithm for skeletonization is as follows:

```
Algorithm Skeletonization:
(Input is assumed to be in array Q1)
g = 1;
repeat
  h = 0;
  Q = Q1;
  if g == 1 then g = 0; else g = 1;
  for each array element p of Q
    if contour(p) then
      case g of
        0:  if 1 < B(p) < 7 and (A(p) = 1 or C(p) = 1)
                            and (p[2] + p[4] * p[0] * p[6] = 0)
            then deletion(Q1, p); h = 1;
        1:  if 1 < B(p) < 7 and (A(p) = 1 or C(p) = 1)
                            and (p[0] + p[6] * p[2] * p[4] = 0)
            then deletion(Q1, p); h = 1;
      endcase;
    endfor;
until h = 0;
```

The above algorithm reduces the numeral pattern to a thickness of one pixel. This "skeletal" bitmap of the numeral is now rethickened to a uniform thickness. As such, numerals of varying thickness are rethickened to a uniform thickness. Even within the bitmap array of a numeral, the thickness of the numeral may be different in different parts. Thinning and rethickening can also smoothen these variations to a large extent. The algorithm for rethickening looks at a pre-specified neighborhood of each 1_pixel and fills this neighborhood with 1's. Figure 5 shows examples of digits

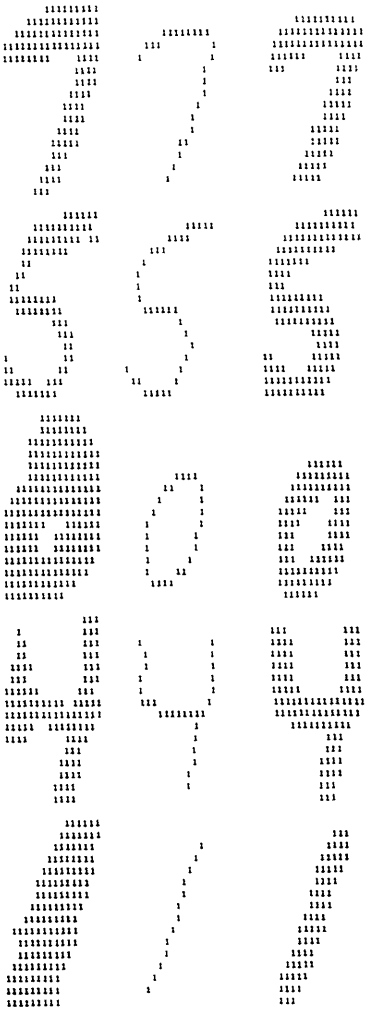


Fig. 5. Examples of numerals, their skeletonized and rethickened versions.

which have been skeletonized and then rethickened. Note that the digits in the first column, which area the original digits, are of varying thickness. Certain numerals like "5" are extremely narrow whereas others like "1" are of greater thickness. Even within the bitmap of numerals like "7", there are variations in thickness. The third column represents the rethickened digits which are distinguished by the uniformity in their thickness not only within a numerical bitmap but also within a set of numeral bitmaps.

While previous systems have used skeletonization for recognition purposes, there has been no previous attempt to use skeletonization followed by rethickening to enhance the stroke uniformity of the digits in order to improve the quality of the input to the neural net based recognizers.

3.4. Normalization

Normalization is a form of "size-reduction" or a demagnification process by which each of the numerals of varying sizes is cast into a form suitable for the recognizer. Up to this stage, the size of the digit varies, but is typically around 30 pixels ×

50 pixels in our case. Since the recognizer in our system is a backpropagation neural net, the numerals must be cast into a standard format. Each of the numeral bitmaps is reduced to an array of size 16 pixels × 16 pixels. The normalization algorithm passes a window over the input bitmap and based on the 1-pixels inside the window, the pixel corresponding to the present position of the window is either made a 1-pixel or a 0-pixel. This transformation preserves the structural features of the digit while reducing its size.

3.5. Slant Correction

Slant correction is another stage which, like the thinning–rethickening stage, is designed to minimize structural variations in the input prior to the recognition stage. The recognizer in our case is tolerant to small amounts of slant and rotation but larger variations reduce overall accuracy. One way to overcome this problem is to train the network with numerals of all possible slants and rotations; this is not a practical solution when one considers the enormous variation that characterizes handwritten characters.

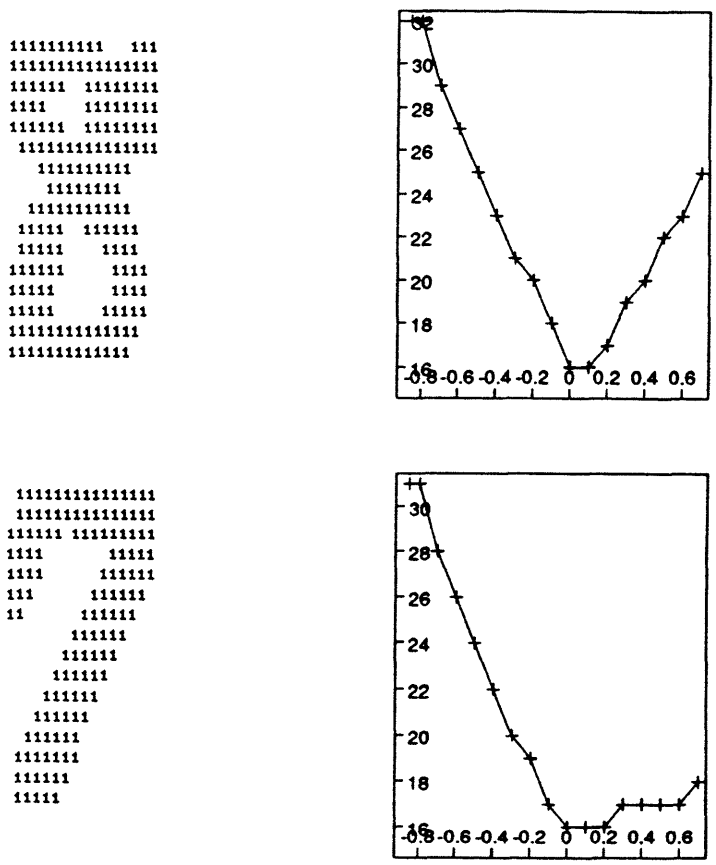


Fig. 6. Numerals and their width profiles. x-axis: angle of slant (radians); y-axis: width of the numeral.

In view of the above, a new strategy has been developed.[22] Before we present the algorithm, we discuss the basis for it. Our analysis revealed that numerals are rarely slanted beyond 45 degrees. While pathological cases involving bigger angles can be found, this hypothesis appears to be statistically valid, based on our detailed study of characters in the NIST Handprinted Numerals Database.

Further, a numeral possesses minimum width when it has least slant. Extensive studies conducted by members of our research team support this hypothesis. Again, there are exceptional cases where this assumption is not true, but these are rare. Width profiles of characters from the NIST Handprinted Numerals Database were constructed showing the width of the numeral versus its angle of slant for various degrees of slant (see Fig. 6). Our algorithm searches for a minimum on the width profile of a numeral using a form of constrained search on the angle of slant. The justification for this approach is described in Ref. 22.

Our algorithm uses the procedure presented in Ref. 15 to perform transformation of the given bitmap into another one with a given degree of slant. The transformation formulae are as follows:

Function Transform_bitmap_through_angle(slant_angleγ)
$\forall$ pixels (x, y) such that (x, y) is a 1_pixel
$x' = x - y = times \tan(\gamma)$;
$y' = y$

```
Algorithm Blob
{
  gamma = 45
  while gamma > .5{
    w1 = WidthOfArray(input_bitmap);
    new_bitmap = TransformArrayByAngle(gamma);
    (Slants the input_bitmap by angle gamma)
    w2 = WidthOfArray(new_bitmap);
    new_bitmap = TransformArrayByAngle(-gamma);
    (Slant in the negative gamma direction)
    w3 = WidthOfArray(new_bitmap);
    min_width = min (w1, w2, w3);
    if(min_width == w2)
      input_bitmap = TransformArrayByAngle(gamma);
    elseif (min_width == w3)
      input_bitmap = TransformArrayByAngle(-gamma);
    (Otherwise leave the input as it is)
      gamma = gamma/2;
  }
  Printout(input);
}
```

A sample of results of slant correction obtained with the above algorithm is shown in Fig. 7.

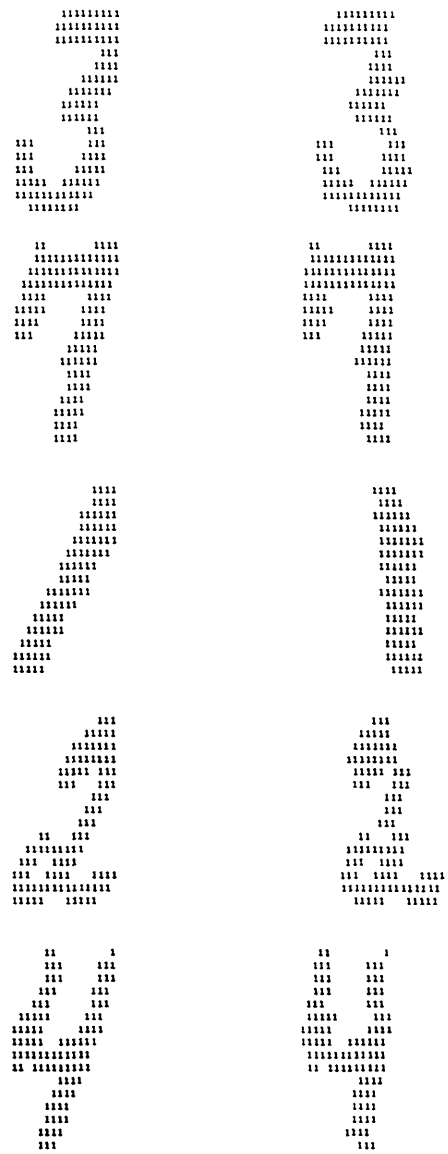


Fig. 7. Examples of some numerals and their slant corrected versions.

3.6. Neural Net Based Recognition

The recognizer consists of a three-layered neural net. The input layer consists of 16×16 linear units, while the output and the hidden layers are composed of logistic units. The hidden layer has 20 units while the output layer has 10 units. Each of the units in the output layer represents a digit (see Fig. 8). The activation of the units vary from 0 to 1.

We trained the above net with the backpropagation algorithm discussed in Ref. 23. If the network is "over-trained", then its generalization ability becomes

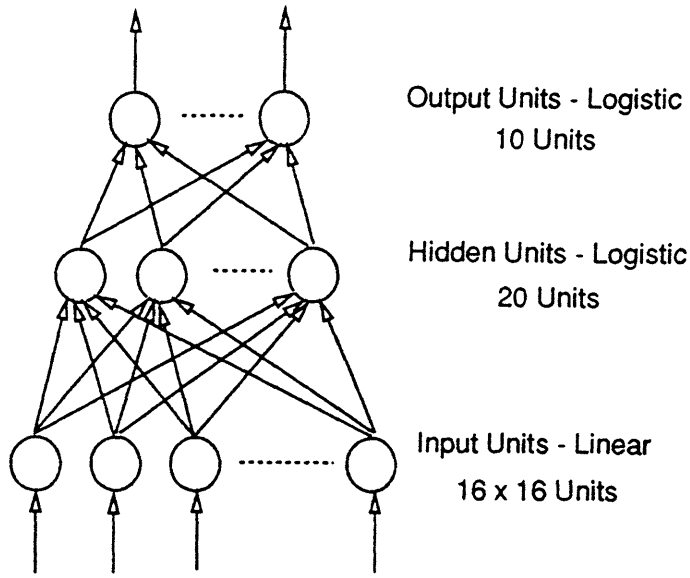


Fig. 8. Three-layered neural net.

poor whereas if it is "under-trained", its recognition accuracy falls. So the network was required to be trained to an "optimal accuracy range".

The significance of the stages involving thinning and rethickening, and slant correction deserves mention here. The backpropagation architecture is not very tolerant to slant and rotation. Though one could have trained the net with all possible slants of all the numerals, this is an impractical strategy as discussed earlier. Usually, the higher the number of different patterns the neural net must encode, the fuzzier are the boundaries in the pattern space over which the net is performing discrimination.

3.7. Post-Processing

The post-processor, which follows the recognition stage, attempts to overcome the shortcomings of the latter. The good generalization abilities which provide neural networks an edge over other types of systems can also become their shortcoming in certain situations due to "over-generalization". The post-processor is a knowledge based pattern discrimination system that utilizes structural methods. It looks at the numeral bitmap and the prediction about the bitmap by the neural net and produces a confidence value. The system uses gross structural features such as strokes and contours to generate its confidence values. In our present system, the prediction from the neural network is used by the post-processor to "focus its attention" on specific features to look for, in the skeletonized bitmap of the numeral. For example, the neural network has a tendency to confuse a "3" with an "8". Thus it may have high values for the activations at the nodes representing both "3" and "8" in the output layer. The post-processor, on detecting high activation values at these

nodes, triggers a module which looks for loops in the lower half of the skeletonized pattern. There are other such modules based on structural features like slant of a stroke, loop in a pattern, center of gravity of the pattern, and so on. The need for a post processor arises only when there is no "clear winner" in the neural net based prediction. We define a node to be a clear winner of its activation is the highest and also if the difference between its activation and the second highest activation is above a threshold.

4. IMPLEMENTATION AND RESULTS

The system described above has been implemented in C language on an IBM PC 486. The neural net has 256 linear input units, 35 logistic hidden units and 10 logistic output units. It has been trained with data extracted from the NIST database, hosted on a CD-ROM containing one million digits of unsegmented data. From this database, a subset of 8000 character bitmaps was generated as the training set for the neural net. An additional 3000 numerals were generated as the test set.

The segmentation stage provided an overall accuracy of about 96.8%. Table 1 shows the tabulated results of the segmentation algorithm.

Table 1. Effectiveness of segmentation algorithm.

Total number of strings segmented	529
Total number of characters evaluated	1964
Correct segmentation of entire strings (overall string accuracy)	96.8%
Correct segmentation of overlapping and non-touching characters	100%
Correct splitting of strings containing connected characters	75.4%

The neural net was trained until the mean square error converged to a prespecified low value. The net was then tested both on the training and the test set of patterns. We introduced a rejection threshold of 0.7 (on a scale of 0 to 1.0) which represents a threshold activation for the neural net. If the value of the highest activation of the output units of the recognizer was below this value, the numeral was rejected. During this stage of testing, the post processor was not integrated with the system. The results shown in Table 2, based on tests with 400 strings *directly scanned off a collection of bank checks.* The strings contained a total of about 1900 digits, that is, an average of about five digits per string. The system recognized 89.6% of these characters with a rejection rate of 4.08%.

The above numbers should be compared with the previous studies and results, obtained by other researchers, keeping the complexity of the different target applications in view. As compared to the task of postal ZIP Code recognition, our application has more inherent uncertainties in the form of unknown number of digits in the string. This leads to lower accuracies in segmentation and consequently in

the overall system. In the case of pre-segmented digits, systems with accuracies of up to 99.0% have been achieved.[4] However, the availability of pre-segmented digits is inconsistent with our application. For other related work, see Ref. 24.

Table 2. Accuracies on training and test data for the recognizer.

Number of hidden nodes	Training data	Rejection	Testing data	Rejection
35	95.05%	4.1 %	96.37%	4.08%
40	97.96%	6.95%	96.5 %	7.2 %

We then integrated the post-processor with the system and ran a test of a smaller size, i.e. with about 100 checks. The system displayed an accuracy of about 94.2%, recognizing 377 characters out of 400 digits on the checks. The string accuracies exceeded 80%. Some digitized check samples used in the experiment are shown in the Appendix.

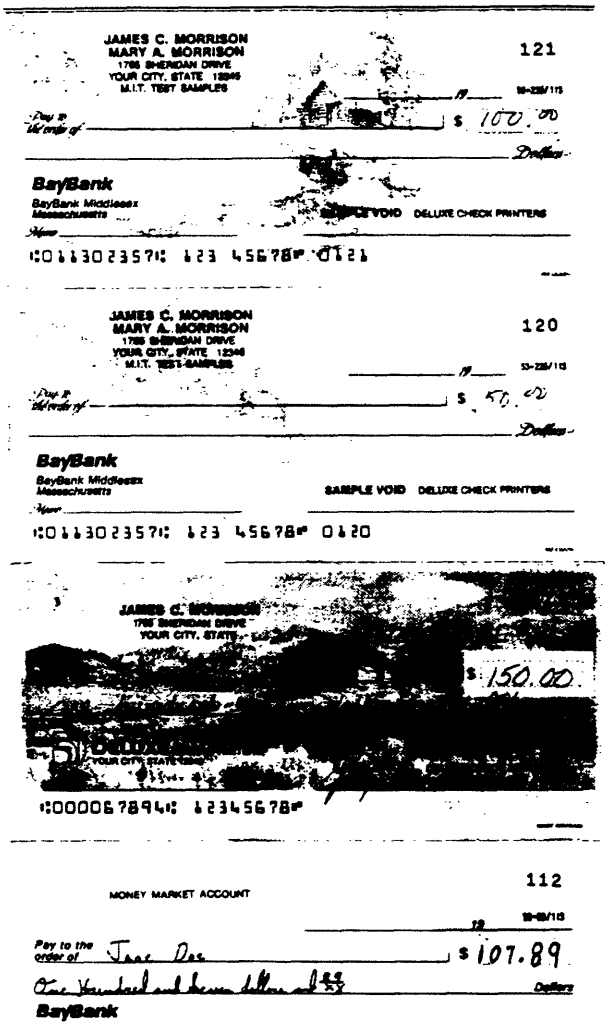
5. CONCLUSION

In this paper, a complete system for off-line handwritten numeral recognition has been presented. The recognizer is a three-layered neural net trained with backpropagation algorithm. The uniqueness of our system lies in the elaborate preprocessing based on structural pattern recognition methods combined with a neural net based recognizer. The preprocessing includes a blackboard-like segmentation system that utilizes multiple segmentation techniques to cater to a wide set of input styles, and offers high speeds when dealing with relatively simple inputs. In addition to segmentation, pre-processing also involves optimized thinning and thickening algorithms, normalization, and slant correction. At the recognition stage, we use the newer neural network based techniques, supplemented by the refinement of traditional structural techniques in the form of a post-processor module. We believe that much of the controversy regarding connectionist systems vs. symbolic systems is irrelevant in that the techniques can be exploited in a supplementary rather than in a mutually exclusive manner. Though our initial tests after the integration of the post-processor may not have been carried out on a statistically significant sample, it is our belief that the results indicate the benefits of building systems which exploit both connectionist and symbolic (structural techniques here) architectures to build hybrid systems.

ACKNOWLEDGEMENTS

We thank fellow researchers of the Automated Character Recognition Group at the Sloan School of Management, MIT, for their valuable inputs.

APPENDIX. SOME DIGITIZED CHECK IMAGES USED IN THE EXPERIMENT



REFERENCES

1. T. Fujisaki, T. E. Chefala, J. Kim, C. C. Tappert and C. G. Wolf, "Online run-on character recognizer: Design and performance", *Int. J. Pattern Recogn. Artif. Intell.* **5**, 1&2 (1991) 123–137.
2. 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", in *Proc. United States Postal Service Advanced Technology Conf.*, 1988.
3. K. Ikeda, T. Yamamura, Y. Mitamura, S. Fujiwara, Y. Tominaga and T. Kiyono, "On-line recognition of handwritten characters utilizing positional and stroke vector sequences", *Pattern Recogn.* **13**, 3 (1981) 191–206.
4. Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel, "Backpropagation applied to handwritten Zipcode Recognition", *Neural Comput.* **1**, 4 (1990) 191–208.
5. B. T. Mitchell and A. M. Gillies, " Advanced research in recognizing handwritten ZIP Codes", in *Proc. United States Postal Service Advanced Technology Conf.*, 1988.

6. M. Shridhar and A. Badreldin, "Recognition of isolated and simply connected handwritten numerals", *Pattern Recogn.* **19**, 1 (1986) 1–12.
7. P. S. P. Wang, M. V. Nagendraprasad and A. Gupta, "A hybrid approach to handwritten numeral recognition", in *Proc. Second Int. Workshop on Frontiers in Handwriting Recognition*, Paris, France, Sept. 1991, pp. 145–154.
8. P. S. P. Wang, Ed., *Character and Handwriting Recognition: Expanding Frontiers*, World Scientific, 1991.
9. K. Yoshida and H. Sakoe, "On-line handwritten character recognition system for a personal computer system", *IEEE Trans. Consumer Electronics* **28**, 3 (1982) 202–209.
10. I. Guyon, "Applications of neural networks to character recognition", *Int. J. Pattern Recogn. Artif. Intell.* **5**, 1 & 2 (1991) 353–382.
11. R. Nag, A. Lui and A. Gupta, "Handwritten numeral recognition using dynamic programming neural networks on an off-line basis", Discussion Paper #151-91, International Financial Services Center, Sloan School of Management, MIT, 1991.
12. K. Fukushima et al., "Neocognitron: A neural network model for a mechanism of visual pattern recognition", *IEEE Trans. Syst. Man and Cybern.* **13**, 5 (1983) 826–834.
13. K. Odaka, H. Arakawa and I. Masuda, "On-line recognition of handwritten characters by approximating each stroke with several points", *IEEE Trans. Syst. Man Cybern.* **12** (1982) 898–903.
14. C. C. Tappert, C. Y. Suen and T. Wakahara, "On-line handwriting recognition — A survey", in *Proc. 9th Int. Conf. on Pattern Recognition*, Rome, 1988.
15. R. Bozinovic and S. Srihari, "Off-line cursive script word recognition", *IEEE Trans. Pattern Anal. Mach. Intell.* **2** (1989) 68–83.
16. J. C. Simon and O. Baret, "Regularities and singularities in line pictures", *Int. J. Pattern Recogn. Artif. Intell.* **5**, 1 & 2 (1991) 57–77.
17. R. Fenrich and S. Krishnamoorthy, "Segmenting diverse quality handwritten digit strings in near real-time", in *Proc. United States Postal Service Advanced Technology Conference*, 1990, pp. 523–537.
18. A. Pervez and C. Y. Suen, "Segmentation of unconstrained handwritten numeric postal Zip Codes", in *Proc. 6th Int. Conf. on Pattern Recognition*, 1982, pp. 545–547.
19. P. L. Sparks, M. V. Nagendraprasad and A. Gupta, "An algorithm for segmenting handwritten numerals", presented at the Second International Conference on Automation, Robotics and Computer Vision, Singapore, Sept. 1992.
20. L. D. Erman, F. Hayes-Roth, V. R. Lesser and D. R. Reddy, "The HEARSAY-II speech understanding system: Integrating knowledge to resolve uncertainty", *ACM Comput. Surv.* **12** (1980) 213–253.
21. M. V. Nagendraprasad, P. S. P. Wang and A. Gupta, "An improved algorithm for thinning binary digital patterns", in *Proc. 11th Int. Conf. on Pattern Recognition*, The Netherlands, Sept. 1992, pp. 386–389.
22. M. V. Nagendraprasad, A. Gupta and V. Feliberti, "A new algorithm for correcting slant in handwritten numerals", Discussion Paper #215-92, International Financial Services Research Center, Sloan School of Management, MIT, 1992.
23. J. McClelland and D. Rumelhart, *Explorations in Parallel Distributed Processing*, MIT Press, 1988.
24. D. G. Elliman and I. T. Lancaster, "A review of segmentation and contextual analysis techniques for text recognition", *Pattern Recogn.* **23** (1990) 337–346.

Received 19 January 1992; first revision 30 July 1992; second revision 5 January 1993.

Amar Gupta is Senior Research Scientist at the Sloan School of Management, Massachusetts Institute of Technology and Co-Director of Productivity from Information Technology (PROFIT) initiative. He received his Ph.D. in computer science from the Indian Institute of Technology, New Delhi, a master's degree in management from MIT and a B.Tech degree in electrical engineering from the Indian Institute of Technology, Kanpur.

Dr. Gupta is a member of the Administrative Committee of the IEEE Industrial Electronics Society. He serves as a consultant to a number of corporations and government agencies on various aspects of computer technology.

Dr. Gupta has been involved in research and management activities since 1974. Since joining MIT in 1979, he has been active in the areas of multiprocessor architectures, distributed homogeneous and heterogeneous databases, expert systems, and information technology. He has written more than 70 technical articles and papers and produced seven books in these areas.

M. V. Nagendraprasad received his B.Tech in electrical engineering from the Indian Institute of Technology, Madras, and his M.S. degrees from IIT Madras and the Massachusetts Institute of Technology. Presently, he is a doctoral candidate at the University of Massachusetts, Amherst. His research focuses on artificial intelligence with an emphasis on distributed AI systems and machine learning. His interests also include software engineering and statistics.

Patrick S. P. Wang has been tenured full professor of computer science at Northeastern University since 1983, research consultant at MIT Sloan School since 1989, and adjunct faculty member of computer science at Harvard University Extension School since 1985. He received his Ph.D. in computer science from Oregon State University, his M.S. in I.C.S. from Georgia Institute of Technology, his M.S.E.E. from National Taiwan University and his B.S.E.E. from National Chiao Tung University. He was on the faculties of the University of Oregon and Boston University, and senior researcher at Southern Bell, GTE Labs, and Wang Labs prior to his present position.

In addition to his research experience at MIT AI Lab, Prof. Wang has been a visiting professor and has been invited to give lectures, do research and present papers in more than a dozen countries in Europe and Asia and many universities and industries in the U.S.A. and Canada. He has published over 70 technical papers and 7 books in Pattern Recognition, AI, and Imaging Technologies and has three OCR patents by US and Europe Patent Bureaus. As IEEE senior member he has organized numerous international conferences and workshops and served as reviewer for many journals and NSF grant proposals. Prof. Wang is currently a founding Editor-in-Charge of the *International Journal of Pattern Recognition and Artificial Intelligence*, an Editor-in-Chief of the series on *Machine Perception* and *Artificial Intelligence* by World Scientific Publishing Co., and elected chair of IAPR-SSPR (International Association of Pattern Recognition). In addition to his technical interests, he has also written several articles on the operas of Verdi, Puccini, and Wagner, and the symphonies of Mozart, Beethoven, and Tchaikovsky.