MIT Sloan School of Management

Working Paper 4461-04
January 2004

HANDWRITTEN BANK CHECK RECOGNITION OF COURTESY AMOUNTS

Rafael Palacios, Amar Gupta, Patrick S.P. Wang

HANDWRITTEN BANK CHECK RECOGNITION OF COURTESY AMOUNTS

RAFAEL PALACIOS
Universidad Pontificia Comillas
Alberto Aguilera 23, E-28015 Madrid, Spain
Email: Rafael.Palacios@iit.upco.es

AMAR GUPTA
Massachusetts Institute of Technology
1 Broadway, Cambridge, MA, 02139 U.S.A
Email: agupta@mit.edu

PATRICK. S.P. WANG
Northeastern University
360 Huntington Av, Boston, MA, 02115 U.S.A
Email: pwang@ccs.neu.edu

In spite of rapid evolution of electronic techniques, a number of large–scale applications continue to rely on the use of paper as the dominant medium. This is especially true for processing of bank checks. This paper examines the issue of reading the numerical amount field. In the case of checks, the segmentation of unconstrained strings into individual digits is a challenging task because of connected and overlapping digits, broken digits, and digits that are physically connected to pieces of strokes from neighboring digits. The proposed architecture involves four stages: segmentation of the string into individual digits, normalization, recognition of each character using a neural network classifier, and syntactic verification. Overall, this paper highlights the importance of employing a hybrid architecture that incorporates multiple approaches to provide high recognition rates.

Keywords: Handwritten checks; Reading of unconstrained handwritten material; neural network based reading; automation of banking systems.

1. Introduction

According to the U.S. Federal Reserve Bank, checks account for 60% of the non-cash transactions, and nearly 50 billion checks worth $47.7 trillion were processed in 2001 in the United States alone.[1] Since most of the checks need to be partially processed by hand, there is significant interest in the banking industry for new approaches that can be used to read paper checks automatically.

Character recognition systems are of two types: On-line systems, where the sequence in which the characters are written is known; and Off-line systems, where only the final image is available. In most information technology applications, on-line processing requires a greater effort because of time constraints; however, in optical character recognition the most difficult area is off-line reading.[2, 3]

The account number and the bank code are printed on the checks in magnetic ink and are the only two fields that can be processed automatically with near-perfect accuracy by magnetic ink character recognition (MICR) systems. Since the character

set used for account numbers is a special type font (E-13B or CMC-7 OCR), these fields can be easily read using MICR machines or optical (OCR) systems. The other fields may be handwritten, typed, or printed; they contain the name of the recipient, the date, the amount to be paid (textual format), the courtesy amount (numerical format) and the signature of the person who wrote the check.

The official value of the check is the amount written in words; this field of the check is called "legal amount". The amount written in numbers is supposed to be for courtesy purposes only and is therefore called "courtesy amount". Nevertheless, most non-cash payment methods use only the amounts written in numbers. The information contained in a check is frequently handwritten, specially considering that most of the checks that were written by computer systems have been gradually replaced by newer methods of electronic payment. Handwritten text and numbers are difficult to read by automatic systems (and sometimes even for humans); so check processing normally involves manual reading of the checks and keying in their respective values into the computer. Accordingly, the field of automatic check processing has witnessed sustained interest for a long time. This has led to complete systems with reading accuracy in the range 20–60% and reading error in the range 1–3% beginning to be installed in recent years.[4] The performance in handwriting recognition is greatly improved by constraining the writing, which addresses the problem of segmentation and makes the people write more carefully. Nevertheless, banks are not willing to change the format of the checks to impose writing constraints such as guidelines or boxes to specify the location where each particular piece of information should be recorded. Instead they are interested in reducing the workload of the employees manually reading the paper check. Since employees also make mistakes reading or typing the amount of the checks, a single manual read rarely drives the whole process. A system that is able to read checks automatically would be very helpful, especially if it is fast and accurate. Even if misclassification occurs, the mistake could potentially be detected during the recognition process; however it is more desirable that the system rejects a check in case of doubt (as described in section 3) so that it can be directed to manual processing from the beginning.

The system described in this paper performs all the tasks necessary to translate the image of the check into a format that can be readily processed by the central computer of the bank. It highlights the need to use a multi-pronged strategy to enhance accuracy rates. The system locates and reads the courtesy amount,[5] which is the main field that banks use to process the check. Other researchers have previously described or implemented systems to read courtesy amount, legal amount and date fields on checks.[6, 7, 8, 3, 9, 10, 11, 12] This illustrates the broad, almost universal, interest in the area of automatic reading of bank checks.

From a historical perspective, document understanding systems (DUS) were designed to read forms that included identification codes and position marks on them. Frequently, the writer was required to use black pen only; this greatly facilitates the recognition process. However, this is not the case with checks issued in the US where individuals are permitted to use pens with inks of any color. Also, many forms are preprinted with individual boxes for each character; this restriction helps to drastically reduce the incidence of connections among adjacent characters. If such boxes could be preprinted on checks, a number of preprocessing tasks (binarization, noise reduction and line removal) can be performed more easily. The same is true for the

segmentation. However, the banks have been reluctant to adopt such restrictive measures on paper checks. Because of the absence of restrictions on the color of the ink and formatting limitations, U.S. checks present the full challenge of attempting to "read" totally unconstrained handwritten material.

In this paper, we describe our approach to "reading" checks using a hybrid architecture that incorporates multiple traditional and emerging approaches. After one has detected the courtesy amount field within the image of the check and read that field in binary mode, the new image is dissected into individual characters. This task, described in section 2, is the most challenging part of the process and actually involves the application of multiple splitting algorithms inside a feedback loop that is driven by the results obtained by the recognition module. The recognition module (Section 3) uses a combination of neural networks to classify digits with very high levels of confidence. The process must be computationally efficient since it is applied several times per character in case of connected or overlapped digits. The final post-processing module is described in Section 4; it verifies the syntax of the amount to minimize the instances involving incorrect readings.

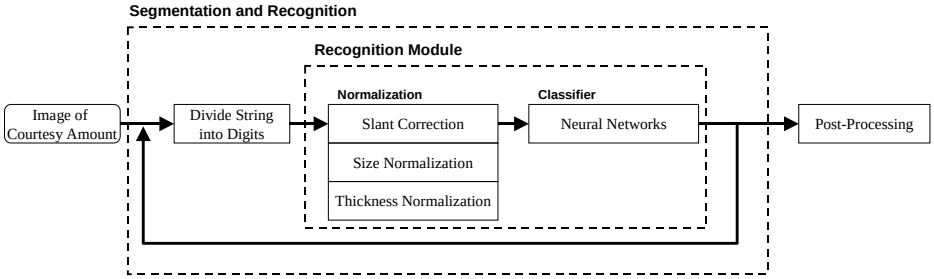

Figure 1: Key steps in reading the courtesy amount

2. Segmentation of Courtesy Amount

Several methods and strategies for segmentation are discussed in.[13] Since the courtesy amount field may be written in different formats, sizes, and styles, (as shown in Figure 2), the segmentation process is complicated. It involves the separation of touching characters, and the merging of character fragments with other pieces. Figure 2a shows numeral '2' connected to a numeral '9', and a numeral '0' connected to a delimiter line. Figure 2b shows numeral '5' made up of two independent strokes that are not touching each other. After the analysis of 1500 real check from Brazilian banks, it was found that 20% of them contained connected digits.

Instead of trying to segment the word into isolated letters, some researchers have attempted to recognize complete words. The latter approach can only be applied in situations that involve a predetermined set of valid strings of characters. More often, the images are divided into digits using structure-based techniques and then recognized as individual characters; this approach is called segment-then-recognize. A different approach is the segment-by-recognition that works by applying the recognition algorithm within a sliding window that is moved along the text line. If the window covers portions of two consecutive characters, then the recognition module gives an

error. However, when the sliding window takes the correct position and the size to hold a complete character, then recognition occurs. The problem with handwritten text is that character isolation using rectangular windows is very unlikely, because characters are usually slated and connected to each other.

The software used by postal services to recognize zip codes exploits the fact that a zip code consists of exactly five digits; or ten characters (including the dash) in the case of extended US zip codes.[14, 15] Since dollar amounts are strings of variable length, the algorithms used to recognize zip codes cannot be applied directly to bank checks. Nevertheless, there are some restrictions on the number of characters between commas and periods. The latter type of domain knowledge has been utilized to reduce the number of read errors within the Post-Processing module (discussed later in this paper) and not as part of the segmentation strategy, which was developed to read any unconstrained number.

With respect to U.S. checks, a finite state automaton was proposed in [16] to segment and analyze the variety of styles found in the decimal part of these checks. In other countries the decimal part of the amount is rarely found in a style other than scientific format. A major characteristic of check amounts is the extensive use of delimiters as suffixes or prefixes of the amounts (some examples are shown in Figure 2). In countries where the currency symbol is written after the number, delimiters are used in front of the number. Further, in countries where there is no decimal part in monetary values, delimiters are common at the end of the number.

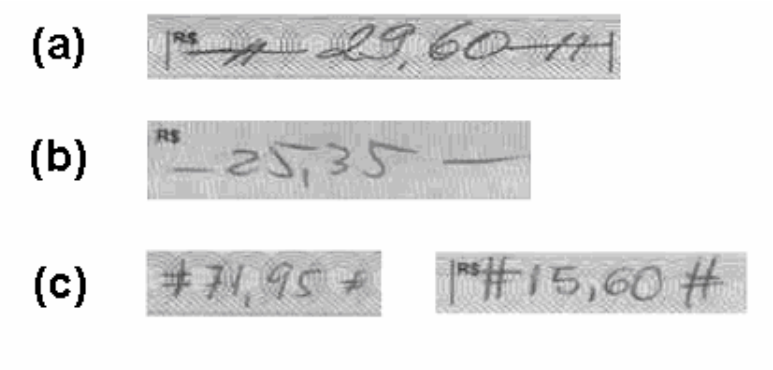

Figure 2: Examples of delimiters in Brazilian checks

2.1. *Feedback Strategy for Segmentation*

The algorithm implemented by our team commences the segmentation process by extracting the connected components from the image of the amount. These connected components may comprise digits, fragments of digits, groups of digits, delimiters, punctuation, or noise. Very small segments are classified as noise and very wide segments are classified as delimiter lines. All other segments are sent sequentially to the recognition module for normalization and classification by the neural network. If the recognition module rejects a segment, then incorrect segmentation is assumed and the alternative splitting algorithms or the fragment merging algorithms are applied. This approach is similar to the strategy proposed by Congedo and Dimauro that also

alternates between segmentation and recognition;[14, 17] however their systems lack the capability to correct fragmented digits.

Fragments of digits can be detected at early stages of segmentation by performing overlap analysis of the minimum bounding boxes of the connected components. In the current implementation, merging of fragments is completed before entering the segmentation loop, even at the risk of merging two independent digits. The segmentation loop was developed mainly to separate connected digits; in the unlikely event of merging two independent digits subsequent separation is not a problem. Nevertheless, in the feedback-based strategy, it is possible to perform both fragment merging and multiple-digit separation concurrently based on the results obtained from the recognition module; this is because segments are processed sequentially.

Any segment that needs to be recognized is pre-classified as a digit, a set of multiple connected digits (denoted multiples), a punctuation mark or a delimiter. This pre-classification is achieved based on size, aspect ratio, relative position, and number of strokes. Only digits and multiples are handled at this stage; digits are sent directly to the recognition module, while multiples are divided into pairs of segments before being passed on to the recognition module.

When a digit is rejected at the recognition module, one of the following situations applies:

- The segment is comprised of two digits in such a way that it was pre-classified as single digit. This is the most likely case and includes examples of double zeros and of numeral '1' being connected to another digit. Additional segmentation in this case will separate the segment into two digits, solving the problem.
- The segment is comprised of a single numeral that is not recognized by the neural network. This case has a small incidence (see accuracy results of the neural networks in the following section) and leads to the rejection of the check.
- The segment is comprised of a symbol or letter, not a numeral. For example, symbol '#' is used in some Brazilian checks. In this case, the check is also rejected.

Since the scenario of multiple segments is the most likely event associated with digit rejection, additional segmentation algorithms are applied. If the segment is a single character, then additional segmentation produces pairs of fragments that are also rejected. Hence, the result will be check rejection anyway, and no mistake has occurred because of excessive fragmentation.

In order to avoid infinite loops, additional segmentation effort is applied only once for segments pre-classified as digits. But if a big segment is pre-classified as a multiple connected set, the splitting algorithms are applied and resulting segments are pre-classified again in a recursive process; until the resulting elements are small enough to be considered digits. Since several splitting algorithms can be applied, many combinations are analyzed for each multiple segment. Moreover, if a digit is identified but with inadequate level of confidence is not attained, the digit is also rejected (to avoid reading mistakes). As a consequence, additional segmentation tasks are initiated often, and the number of calls to the recognition module increases significantly. The proper selection of splitting algorithms can increase the performance of the system by

eliminating unnecessary calls to the recognition module, as explained further in the next subsection.

2.2. *Splitting Algorithms*

Several contour splitting algorithms were considered for the purpose of dividing multiple segments into pairs of digits.[13] The research team found drop fall algorithms accurate if several implementations are used concurrently.

Drop fall algorithms simulate the path produced by a drop of acid falling from above the character and sliding downwards along the contour. When the drop gets stuck, it "melts" the character's line and then continues to fall.

The dividing path found by a Drop Fall method depends on three aspects: a starting point, movement rules, and orientation. The starting point should be located near the middle line between the two digits, so it can find its way between both characters. If the starting point is selected far from the middle line, then the drop will fall around the exterior contour. Starting from above is different than starting from bellow (with negative gravity) since joints will produce the opposite movement, cut in one direction and fall around in the other. The current implementation employs Hybrid Drop Fall (HDF) and Extended Drop Fall algorithm (EDF); these are very similar and differ only in the rules that define the movement while cutting the segment (EDF tends to make longer cuts than HDF and is better at dividing a cluster of double zeros). Hybrid Drop Fall (HDF) and Extended Drop Fall (EDF) methods are described in greater detail in [18]. An example of the dividing paths that can be obtained is shown in Figure 3.

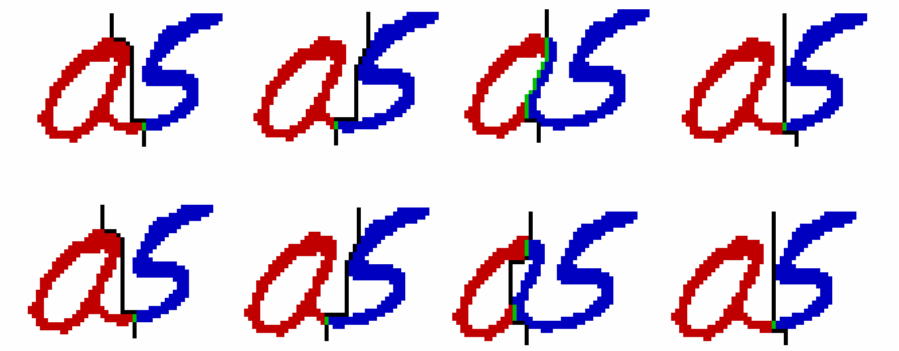

Figure 3: Different Results from Fall Algorithm based on Starting Point and Direction.

Splitting algorithms lead to a set of 8 possible paths to divide a connected component into individual digits Since no single algorithm is able to solve all possible situations, so all of these alternative paths must be considered. Determining which splitting algorithm is producing the correct separation is a problem itself, which can be solved using a neural network approach as proposed in [15]. Another approach is to rank the paths heuristically considering the number of cuts, the length of the cuts, the type of junction, etc. An alternative strategy is to undertake an exhaustive search and to select

the path on the basis of the confidence levels provided by the recognition module. This approach is less efficient but attains the best possible results. The current implementation of the system utilizes the exhaustive approach and additional research is been carried out to find a proper approach to eliminate similar paths and to select only the most promising paths, thereby improving the execution time. The ability of the neural network to automatically recognize multiple segments, delimiters or double zeros, instead of expecting a rejection in those cases, can improve the selection of the correct path and can also enhance the strategy in the feedback loop.[19, 20]

3. Recognition of Individual Characters

The recognition module incorporates neural networks techniques that require preprocessing of high-resolution images in order to obtain a fixed number of features. Typical inputs to the neural network are the values of each pixels in a low-resolution version of the image to be classified. The objective of preprocessing is to obtain small images of the digits that are as uniform as possible, on multiple dimensions. Instead of simply resizing to convert each character into a standard matrix, the slant is also corrected and the thickness of the strokes is also normalized. The normalization process is summarized in Figure 4. Slant correction is a very efficient procedure that is performed at the original resolution. Then a resizing operation is executed; it can shrink or enlarge the image depending on the writing size to produce a 51x36 matrix. At this standard size, the thickness of the pen strokes is normalized, and then a final resizing operation is performed to convert the character image into a 17x12 matrix (204 pixels).

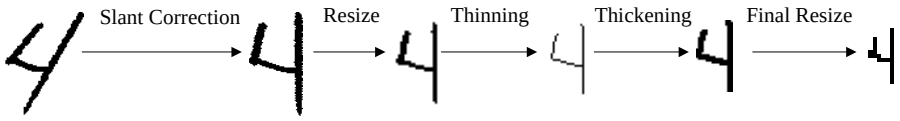

Figure 4: Scheme of normalization process

All these algorithms, although individually efficent from a computational viewpoint, have been optimized in performance and can be executed in a short time. The normalization of 60092 digits extracted from NIST special database of handwritten forms took 40919 seconds on a Sun Fire 880 machine;[21] this comes to less than 0.7s per digit (actually a range from 0.5s to 0.8s depending on the specific digit).

3.1. *Slant Correction*

This type of preprocessing is applied to obtain the same kind of image for the same set of numbers irrespective of the personal writing tilt. Without this correction, the accuracy of most classifiers is degraded significantly by the presence of slant, which is very common in handwritten text. The algorithm used by the research team is based on

the idea that if a numeral is rotated through a series of slanted positions, it usually attains its minimum width when it is least slanted.[19]

No real rotation is performed because the algorithm is slow and sometimes produces bad quality contours. Instead, all pixels in a row are moved to the left or the right, depending on the sign of the correction angle. This is an approximation to a rotation in which the lowest row is never moved, while displacement of upper rows is larger than in lower rows.

The algorithm to obtain the least slanted position finds the rotation angle by using a binary search strategy to minimize the width. Binary search is performed dividing the rotation angle by two in each iteration until a precision of 1 degree is reached.

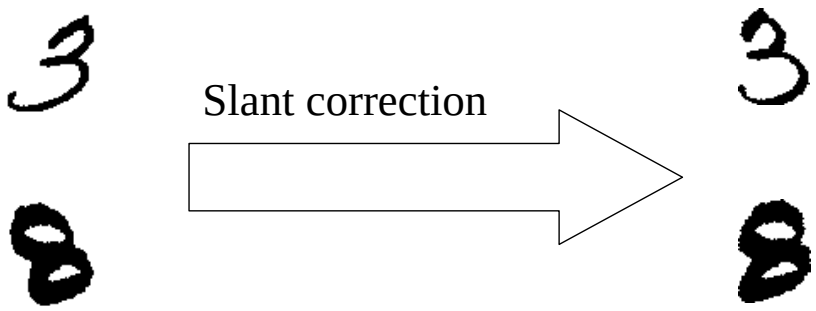

Figure 5: Slant correction applied to positive and negative tilted digits

3.2. *Size Normalization*

The resizing of characters is needed in order to make the recognition operations process independent of the writing size and the resolution of the scanner. Also, algorithms for other normalization operations perform faster on small images, and the topology of the neural networks is simplified, too. Some of our earlier approaches used a matrix size of 256 pixels (16x16).[19, 22] Later the final size was changed to 117 pixels (13x9), while preserving the original aspect ratio. The size currently being used is 204 pixels (17x12); this provides slightly better quality and also preserves the original aspect ratio. The accuracy has greatly improved with the adoption of this approach.

The resizing algorithm is applied twice during the normalization process. The first resizing operations is performed after the slant correction operation and before the thickness normalization operation; it results in a 51x36 matrix. This resolution is adequate to retain the quality of the character and is more efficient for the purpose of thickness normalization than the usual original size. The second resizing operation constitutes the final step in the normalization process and converts the image (already normalized in slant and thickness) to a smaller image with just 204 pixels that will be used as the input to the neural network recognizer. It is important to convert the original image to a standard size before thickness normalization; otherwise the final resize operation into a 17x12 matrix may result in missing strokes or strokes with excessive thickness.

3.3. *Thickness Normalization*

Performance tests conducted by members of the project team revealed the fact that the accuracy of the ultimate recognition process depended significantly on the level of uniformity in the thickness of characters used in the training process and the test process itself. Instead of attempting to attain uniformity of thickness in a single stage, we found that it was computationally more efficient to just "thin" them and "rethicken" them. We begin this process by transforming the raw black and white image into a line drawing of unit thickness by deleting redundant pixels without impacting the overall appearance of the character. This process of skeletonization was implemented very carefully in order to maintain preserve the basic structure and connectivity of the original pattern. Previous researchers have addressed this problem either using a sequential approach or a parallel approach.[23] In the former method, the value of a pixel at the *n* iteration depends on pixels from the *n* iteration, and usually also on pixels from earlier iterations. In the other method, each pixel is obtained as a function of values from the previous iteration only (so the computation can be performed in parallel for every pixel of the image). The algorithm used by us is a parallel method based on the algorithm described in,[24] and offers one of the best results in terms of performance and accuracy, as compared to nine other thinning algorithms.[25] Subsequent to the publication of that comparative evaluation, the algorithm was improved even further by eliminating a number of time consuming steps.[26] Further improvements on this algorithm have been proposed by Carrasco and Forcada in [27].

Definitions:

- The **neighbors of point *p*** are the 8 pixels that surround *p*, and they are numbered from 0 to 7 in the clockwise direction beginning by the point which is on the top of *p*.
- One pixel is considered to be a **contour point** if at least one of its neighbors is white.
- A **contour loop** is defined as a set of contour points, which are connected into a loop.

The algorithm is applied sequentially eliminating points of one contour loop at a time. In each iteration, not all contour points are deleted but only those that satisfy the following conditions:

For the first and odd iterations, the contour point *p* is eliminated if:

Condition 1) $B(p) > 1$, and $B(p) < 7$
Condition 2) $A(p)=1$, or $C(p)=1$
Condition 3) $E(p)=0$

For the second and even iterations, the contour point *p* is eliminated if:

Condition 1) $B(p) > 1$, and $B(p) < 7$
Condition 2) $A(p)=1$, or $D(p)=1$
Condition 3) $F(p)=0$, where:

$B(p)$ is the number of white neighbors of *p*,

$A(p)$ is the number of white-to-black transitions of the neighbors of *p* in the clockwise direction.

$C(p)$ is equal to one only if any of the following conditions is true:

$p(0)=p(1)=p(2)=p(5)=0$ and $p(4)=p(6)=1$
$p(2)=p(3)=p(4)=p(7)=0$ and $p(6)=p(0)=1$;

D(p) is equal to one only if any of the following conditions is true:

$p(1)=p(4)=p(5)=p(6)=0$ and $p(0)=p(2)=1$;
$p(0)=p(3)=p(6)=p(7)=0$ and $p(2)=p(4)=1$;

$E(p)=(p(2)+p(4))*p(0)*p(6)$;
$F(p)=(p(0)+p(6))*p(2)*p(4)$;

The best performance is obtained by rastering all black pixel in the image and checking if they belong to the contour, instead of walking along the contour. The number of operations performed is smaller and the algorithm is converted into a parallel method.

After obtaining the skeleton (one pixel thickness) of the digit, it is necessary to dilate the image uniformly to obtain a standard thickness. Therefore the final thickness of the digit is independent of the type of pen used, and at the same time the character appears clear of noise at the edges.

The re-thickening algorithm is a dilation operation using unitary matrix 3x3. The result of this approach exhibits a uniform thickness of about 3 pixels. One example of thickness normalization, applied to numeral '3' and numeral '8', is shown in Figure 6.

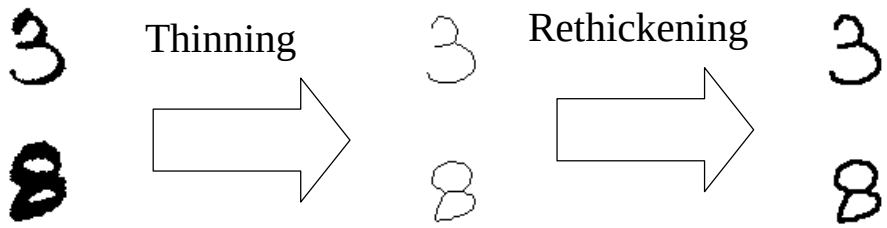

Figure 6: Thickness normalization of non-uniform numeral '3' and thick numeral '8'

3.4. *Neural Network Based Recognition*

While template matching, structural analysis and neural networks have been very popular classification techniques for character recognition, neural networks are now increasingly used in handwriting recognition applications.[6, 28] In our experience with a number of diverse applications, no single neural network model appears to be inherently better than others to a significant extent; instead, higher accuracy rates can be achieved by tailoring network models to the particular problem environment. A network trained to recognize just digits offers better accuracy for check amount recognition than a generalized neural network trained to recognize both letters and numbers.

The topology preferred by the authors is the multilayer perceptron (MLP) topology; this yielded very accurate results and the execution time was shorter as compared to other structures that were evaluated. Other researchers have successfully

employed radial basis function networks (RBFN) and time delay neural networks (TDNN) for character recognition in French checks.[11]

In our endeavor, the recognition module is implemented as an array of four neural networks working in parallel. The results produced by these networks are analyzed by an arbiter function that evaluates the results of all the networks and then produces an overall result. The recognition procedure is shown in Figure 7.

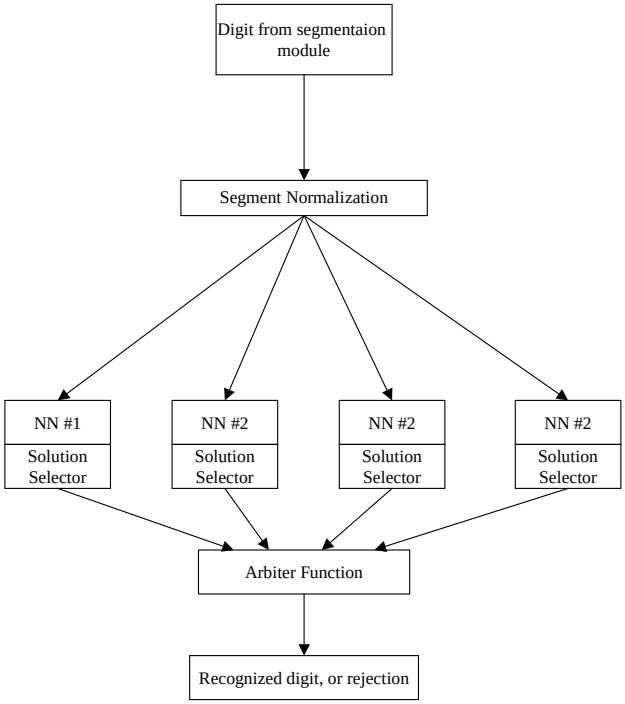

Figure 7: General scheme of Recognition Module

The fact that concurrent execution of different classifiers can increase the accuracy of the system is widely accepted.[2, 29, 30] The outputs of several classifiers can be used together to increase the recognition rate, to reduce the probability on misclassifications, and even to improve both parameters simultaneously. The actual behavior is mandated by the arbiter function, which can be tuned to optimize the relative importance to the different independent results in any way deemed appropriate by the domain experts.

The structure of our preferred neural network is a multi-layer perceptron (MLP) with 204 inputs (1 or 0 values corresponding to 204 pixels of the 17x12 normalized matrix), 50 hidden neurons, and 10 outputs (corresponding to the certainty that the image is digit '0', '1',. '9' respectively).

The ten outputs of the neural network are analyzed by a function called SolutionSelector that returns the most likely result and the corresponding confidence level, or "not_recongnized", which is the predefined threshold for the value in case of doubt. When the confidence level is not attained, the confidence level is low or when contradiction between two possible digits is detected, the output of SolutionSelector will be "not_recognized"; this implies that the neural network is not able to read the digit with the desired level of confidence. If any character in the amount of a check is not recognized, the check is rejected. Rejection means that the system is not able to read the handwriting with enough level of confidence; as such, the check must be read by a person. Since all checks are currently read by bank employees, rejection only sends the check back to the normal procedure in which the check is read by a person. Incorrect recognition can cause more damage; this will happen in only a small percentage of cases with our approach. We prefer the option of having very few incorrect readings, especially if the mistake cannot be detected on subsequent tests. As explained below, a concerted effort has been made to minimize the number of incorrect readings.

3.5. *Evaluation and Testing*

To evaluate the performance of the recognition system, 50000 digits from the NIST Database 19 (Handprinted Forms and Characters Database) were used.[21] Training was performed using 25000 images in the training set (2500 samples corresponding to each digit), and 10000 images in the validation set. The neural networks were trained by adjusting internal weights (parameters) after evaluating the error between the outputs of the network and the expected unitary vectors. The error of the training set always decreases during this process. Further, an independent set of images (validation set), which were not used by the optimization algorithm, was also evaluated in order to avoid over-training of the neural network. The training process was terminated when the error on the validation set began to increase.

Four neural networks of the same structure were trained using the same algorithm and the sample data samples. Since the initial weights of the network were initialized randomly, the resulting networks were slightly different from each other. This is a valuable characteristic that can be exploited in cooperative evaluation to combine the results of the networks into a better classifier, as highlighted in the following sub-section.

In addition to these MLPs, two other topologies were used: Elman backpropagation (ELM), which is a variant of the MLP; and generalized regression neural network (GRNN), which is a kind of radial basis network.[31] Table 1 shows the results of testing the neural networks with 15000 digits from the NIST database. In general, all the networks performed well, though the GRNN produced the least number of reflections but at the cost of a higher level of incorrect recognitions. However, one must consider that the NIST database is not perfect and contains instances of incorrectly segmented digits;[20] therefore, 100% accuracy is not feasible. (The overall results for the training set are: 96.5% correctly recognized, 0.5% incorrectly recognized).

Table 1: Performance of different neural networks developed

Neural Netwok	Number of Cases			Percentage		
	Correct	Rejection	Wrong	Correct	Rejection	Wrong
MLP1	13815	868	317	92.1%	5.8%	2.1%
MLP2	13789	882	329	91.9%	5.9%	2.2%
MLP3	13752	901	347	91.7%	6.0%	2.3%
MLP4	13739	936	325	91.6%	6.2%	2.2%
ELM	13811	886	303	92.1%	5.9%	2.0%
GRNN	13725	557	718	91.5%	3.7%	4.8%

The biggest difference between GRNN and the other topologies is that the training process is faster in the former case: only about 5 seconds for GRNN instead of 10 to 15 minutes for MLPs or ELM. On the other hand, the evaluation process is slower: 1.2s per sample, instead of 0.1 ms for MLP; these times were measured using Matlab neural network toolbox in a Sun Fire 880 workstation environments. For check recognition applications, it is more desirable to have fast evaluation than fast training; this is especially true in the case of our feedback-based architecture because the latter makes a lot of neural network calls.

3.6. *Combining classifiers*

In previous work, it was shown that by combining MLPs, one could improve the accuracy of check recognition.[19, 22] In this paper, we show the level of improvement attained by combining multiple MLPs and also by combining one MLP with GRNN and ELM.

As shown in Figure 7, the Arbiter is the module responsible for combining the results of several networks, and it may adopt different approaches. Since our different classifiers offer very similar accuracy levels, it is not necessary to establish different weights to each individual classifier. The strategy implemented for the arbiter function is a simple voting approach in which a definite level of agreement among the results of the independent classifiers is required. Three strategies have been used:

- Arbiter 2/3: The agreement among any 2 networks out of 3 is required. Otherwise the digit is rejected. As can be deduced from this rule, if the digit is rejected by more than one individual network, it is also rejected by the Arbiter.
- Arbiter 3/3: Total agreement of the 3 networks used is required
- Arbiter 3/4: In a set of four networks, the agreement of at least 3 of them is required.

The results of applying various Arbiter approaches to different sets of neural networks are shown on Table 2. By combining different network topologies, it is possible to improve the probability of correct recognition and simultaneously decrease the probability for wrong reading, using the Arbiter 2/3 approach. Further, in a total

agreement scenario (Arbiter 3/3), the incidence of wrong reading is greatly diminished but the recognition rate also decreases.

Table 2: Results of the classifier combining the information of several neural networks

	Arbiter	correct	rejection	wrong
3 or 4 MLPs	Arbiter 2/3	92.7%	5.6%	1.7%
	Arbiter 3/4	91.4%	7.3%	1.2%
	Arbiter 3/3	87.3%	11.8%	0.8%
MLP+GRNN+ELM	Arbiter 2/3	93.1%	5.7%	1.3%
	Arbiter 3/3	84.0%	15.6%	0.4%

4. Post-processing Module

A post-processing module was designed for evaluating the result from the Arbiter. This module is based on a set of syntactic rules that are used to check if the amount read is meaningful as a monetary amount; this can help to reduce the impact of incorrect readings even further.

Contextual knowledge can often be exploited to enhance the recognition accuracy in most OCR applications. Applications relating to the postal service are usually constrained to a fixed number of state/province values and to a fixed number of digits in postal codes (ZIP codes). Similarly, the courtesy amount could be verified with the legal amount that is written in textual format. Some systems that read monetary amounts exploit contextual information via syntax verification. This involves the notion of pattern parser that determines the validity of a string in terms of monetary amounts.[32] The A2iA Interchange system includes the verifier module within the recognition module to read French checks.[10] In another French system,[9] the syntactic analysis takes place after the recognition system. In the system being described in this paper, the syntactic verifier is applied after recognition and is intended to minimize the number of wrong readings and to show the values in a standard format. The post-processing module is based on a set of rules that incorporate the meanings of the decimal separator and the grouping symbol. There is no distinction between a period and a comma; both are treated as separators, and the analysis is based on the number of digits in each group. This strategy was adopted because the period sign and the comma sign are difficult to distinguish in handwritten form, and because they have opposite connotations in different countries. As an example, the norms for the new Euro currency allow for the use of the decimal symbol and the digit grouping symbols according to the national rules and practices of each country.

The rules for accepting writing formats can be expressed using basic regular expressions that describe, in a very general way, the valid sequences of digits and commas within the amount string:

Set 1: If the decimal part exists, which is the most common case, it must contain 2 digits. If special punctuation is used to make groups in the integer part of the amount; then these groups must be comprised of exactly three digits. The regular expressions for amounts with decimal part are:

```
^[0-9]*,[0-9][0-9]$
^[0-9],[0-9][0-9][0-9],[0-9][0-9]$
^[0-9][0-9],[0-9][0-9][0-9],[0-9][0-9]$
^[0-9][0-9][0-9],[0-9][0-9][0-9],[0-9][0-9]$
```

Set 2: Without the decimal part, the separator is optional. So the regular expressions for amounts without decimal part are:

```
^[0-9]*[,]*$
^[0-9],[0-9][0-9][0-9][,]*$
^[0-9][0-9],[0-9][0-9][0-9][,]*$
^[0-9][0-9][0-9],[0-9][0-9][0-9][,]*$
```

The application of these rules can be depicted as a Deterministic Finite Automata DFA (see Figure 8). It reads the string from right to left and the states change depending on the kind of symbol found in the string: D for digit, and P for punctuation. The system takes states q_i until it reaches a final state (denoted by double circles) or it reaches an error state. Additional description of this DFA diagram and all possible cases are included in [18] and [22].

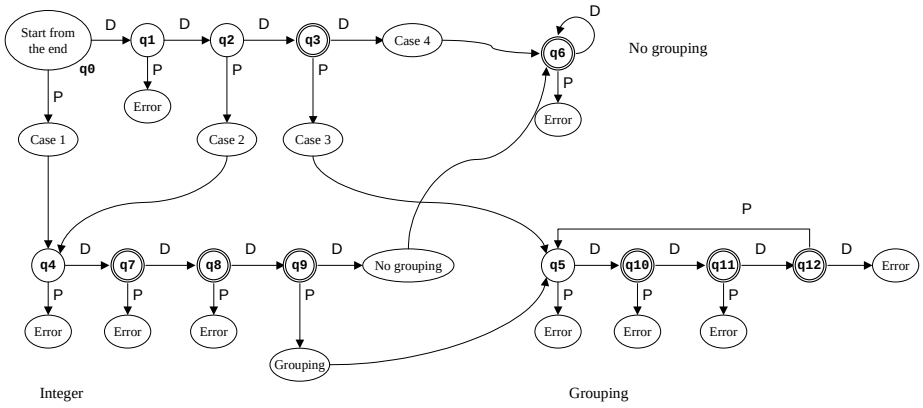

Figure 8: Deterministic Finite Automata to check validity of amounts

5. Conclusion

This paper described a set of algorithms and techniques that can be applied to read checks from many countries, as well as paper documents from other operational environments. By using a multi-staged hybrid architecture, one is able to obtain better recognition rates as compared to those attainable with a single approach only.

The strategy of using multiple techniques in parallel is exemplified by the segmentation strategy in which the courtesy amount field is dissected into individual digits using several splitting algorithms. The segmentation module receives feedback from the recognition module and selects a different splitting algorithm if any digit is not properly recognized. Several splitting algorithms based on the drop-fall

methodology have been presented in this paper. None of the algorithms provides the best separation of connected or overlapped digits in every possible situation, but the system tries the most likely one first and if it fails, then it uses the others in a predefined manner.

In the section on recognition of individual characters, the need for normalization of digit size, thickness of strokes, and slant was highlighted. The neural network architecture employs a set of four neural networks of different types that are run in parallel to minimize the likelihood of erroneous readings. This architecture offers superior accuracy and performance, as compared to structures comprised of single nets only. The importance of an independent classifier that is able to detect incorrectly segmented digits, as well as to reject unknown characters (such as delimiters), has also been emphasized. The final step in the recognition of digits is syntactic post–processing, which helps validate the text string obtained by checking if the result is in conformity with the rules of valid monetary amounts.

By evaluating the efficacy of our hybrid architecture with both U.S. and Brazilian checks, we continue to work towards the delineation of hybrid techniques that can be used to handle checks and other paper documents from many countries.

Acknowledgments

We are indebted to representatives of a number of banks located in North America, South America, Europe and Asia for assisting with research related to their specific domains. Parts of the research described in this paper are covered by a patent.[33] We thank the co-inventors and others who have been involved in this project. The reviewers' comments and suggestions that help make better presentation of the paper are also greatly appreciated.

References

1. United States Federal Reserve Board, "Fed Announces Results of Study of the Payments System. First Authoritative Study in 20 Years". Press Release, November 14th, 2001.
2. N. Arica and F.T. Yarman-Vural, "An overview of Character Recognition Focused on Off-Line Handwritting", IEEE Transactions on Systems, Man, and Cybernetics – PartC: Applications and Reviews, 31(2):216-233, 2001.
3. K. Han and I. Sethi, "An off-Line cursive handwritten word recognition system and its application to legal amount interpretation", Automatic Bankcheck Processing, Wold Scientific Press, 1997, pp. 295-308.
4. S. Kelland and S. Wesolkowski, "A comparison of Research and Production Architectures for Check Reading Systems", Fifth International Conference on Document Analysis and Recognition, pp 99-102, Bangalore, India.. 1999.
5. A. Agarwal, K. Hussein, A. Gupta, and P.S.P. Wang. "Detection of Courtesy Amount Block on Bank Checks" Journal of Electronic Imaging, Vol. 5(2), April 1996, pp. 214-224.
6. H.Bunke and P.S.P.Wang. Handbook of character recognition and document image analysis. Ed. Wolrd Scientific. ISBN 981-02-2270-X, 1997.
7. R. Fan, L. Lam, and C.Y. Suen, "Processing of date information on cheques", Progress in Handwriting Recognition, Editors A.C. Downton and C. Impedovo, World Scientific, 1997, pp.473-479.

8. N.Gorski, V.Anisimov, E.Augustin, O.Baret, D.Price, and J.C.Simon, "A2iA Check Reader: A Family of Bank Recognition Systems", Fifth International Conference on Document Analysis and Recognition, pp 523-526, Bangalore, India. (1999)
9. L. Heutte, P. Barbosa-Pereira, O. Bougeois, J.V. Moreau, B. Plessis, P. Courtellemont, and Y. Lecourtier, "Multi-Bank Check Recognition System: Consideration on the Numeral Amount Recognition Module", International Journal of Pattern Recognition and Artificial Intelligence, Vol 11(4), pp595-618, 1997.
10. S. Knerr, V. Anisimov, O. Beret, N. Gorski, D. Price and J.C. Simon, "The A2iA Intercheque System: Courtesy Amount and Legal Amount Recognition for French Checks", International Journal of Pattern Recognition and Artificial Intelligence, Vol 11(4), pp505-547, 1997.
11. E. Lethelier, M. Leroux, M. Gilloux, "An Automatic Reading System for Handwritten Numeral Amounts on French checks." 3rd International Conference on Document Analysis and Recognition, vol 1:92-97, 1995.
12. C.Y. Suen, Q. Xu, and L.Lam, "Automatic Recognition of Handwritten data on cheques – Fact or Fiction?", Pattern Recognition Letters. 20:1287-1295, 1999
13. R.G. Casey and E. Lecolinet, "A Survey of Methods and Strategies in Character Segmentation", in IEEE Transactions on Pattern Analysis and Machine Intelligence, 18(7):690-706, 1996.
14. G. Congedo, G. Dimauro, S. Impedovo, G. Pirlo "Segmentation of Numeric Strings." Proceedings of the Third International Conference on Document Analysis and Recognition, Vol. II 1038-1041. 1995.
15. F. Kimura and M. Shridhar, "Segmentation-Recognition Algorithm for Zip Code Field Recognition", Machine Vision and Applications 5 (3) 199-210. 1992.
16. K. Hussein, A. Agarwal, A. Gupta, and P.S.P. Wang. "A Knowledge-Based Segmentation Algorithm for Enhanced Recognition of Handwritten Courtesy Amounts", Pattern Recognition, Vol 32(2), pp. 305-316, Feb 1999.
17. G. Dimauro, S. Impedovo, G.Pirlo, and A.Salzo. "Automatic Bankcheck Processing: A New Engineered System", International Journal of Pattern Recognition and Artificial Intelligence, Vol 11(4), pp467-503, 1997.
18. R. Palacios, and A. Gupta, "A System for Processing Handwritten Bank Checks Automatically", submitted to Image and Vision Computing, 2002.
19. R. Palacios, A. Gupta, and P.S.P. Wang, "Feedback-Based Architecture for Reading Courtesy Amounts on Checks", Journal of Electronic Imaging, Vol 12, Issue 1, January 2003, pp 194-202.
20. R. Palacios, and A. Gupta, "Training Neural Networks for Reading Handwritten Amounts on Checks", submitted to the IEEE International Workshop on Neural Networks for Signal Processing, 2003.
21. M.D. Garris, J.L. Blue, G.T. Candela, P.J. Grother, S.A. Janet, C.L. Wilson. "NIST Form-Based Handprint Recognition System (Release 2.0)", US Dept. of Commerce, Technology Administration. National Institute of Standards and Technology. NISTIR 5959. 1997.
22. R. Palacios, A. Sinha, A. Gupta, "Automatic Processing of Brazilian Bank Checks", submitted to Machine Vision and Applications, 2002.
23. L. Lam, S.W. Lee, and C.Y. Suen, "Thinning Methodologies – A Comprehensive Survey", in IEEE Transactions on Pattern Analysis and Machine Intelligence, 14(9):869-885, 1992.
24. P.S.P.Wang, and Y.Y.Zhang, "A fast and flexible thinning algorithm", IEEE Transactions on Computers, 38(5), 1989
25. L. Lam and C.Y. Suen, "An Evaluation of Parallel Thinning Algorithms for Character Recognition", in IEEE Transactions on Pattern Analysis and Machine Intelligence, 17(9):914-919, 1995.
26. M. V. Nagendraprasad, P. S. P. Wang, and A. Gupta "Algorithms for Thinning and Rethickening Digital Patterns" Digital Signal Processing, 3, 97-102 , 1993.

27. R.C. Carrasco, and M.L. Forcada, "A note on the Nagendraprasad-Wang-Gupta thinning algorithm", Pattern Recognition Letters, Vol 16, pp. 539-541, 1995.
28. I. Guyon and P.S.P. Wang. Advances in Pattern Recognition Systems using Neural Network Technologies. Ed World Scientific. ISBN 981-02-144-8. 1993
29. A.S. Atukorale, P.N. Suganthan, "Combining Classifiers Based on Confidence Values", Proceedings of the 5th International Conference on Document Analysis and Recognition, pp.37-40, 1999.
30. C.Y. Suen, J. Kim, K. Kim, Q. Xu, and L. Lam, "Handwriting Recognition – The Last Frontiers", Proceedings of the International Conference on Pattern Recognition, vol. 4:1-10, 2000.
31. Wasserman, P.D., "Advanced Methods in Neural Computing", Van Nostrand Reinhold, New York, 1993.
32. M. Prundaru and I. Prundaru, "Syntactic Handwritten Numeral Recognition by Multi-Dimensional Gramars", Proceedings of SPIE: The International Society for Optical Engineering, Vol 3365, pp. 302-309, 1998.
33. A. Gupta, M.V. Nagendraprasad, P.S.P. Wang, "System and method for character recognition with normalization", U.S. Patent No. 5633954, 1997.

Photo and Bibliography

Rafael Palacios received his degree in Industrial Engineering in 1990 and his PhD in electrical engineering in 1998, from University Pontificia Comillas (UPCO), Madrid, Spain. He has developed research for many companies in projects related to failure detection, image processing and other fields. He was visiting scientist of Sloan school of management in Massachusetts Institute of Technology (MIT) during 2001 and 2002. He is currently an assistant professor in the department of computer science at UPCO. His interests include failure detection and diagnosis, digital image processing, web programming and computer security.

Amar Gupta is Co-Director of the Productivity from Information Technology (PROFIT) Initiative at MIT. He has been at MIT since 1979 and his current areas of interest are: Knowledge Acquisition; Knowledge Discovery and Data Mining; Knowledge Management; and Knowledge Dissemination. He is presently Associate Editor of ACM Transactions of Information Technology (TOIT), and also of Information Resources Management Journal (IRMJ). He is one of the Co-Inventors of a new approach for reading handwritten numbers, and is currently trying to encourage the use of electronic techniques for nationwide paper-less check processing. He serves as an advisor to a number of corporations and international organizations.

Patrick S. Wang is IAPR Fellow, tenured full professor of computer science at Northeastern University, research consultant at MIT Sloan School, and adjunct faculty of computer science at Harvard University Extension School. He received his Ph.D. in C.S. from Oregon State University, M.S. in I.C.S. from Georgia Tech, M.S.E.E.from Taiwan University and B.S.E.E. from Chiao-Tung University. He was faculty at Oregon and Boston University, and senior researcher at Southern Bell, GTE Labs and Wang Labs. Dr. Wang was elected Otto-Von-Guericke Distinguished Guest Professor of Magdeburg University near Berlin, Germany, and serves as Honorary Advisor Professor for Xiamen University, and Guangxi Normal University, Guilin, China, since 2001.