Improving the Courtesy Amount Recognition Accuracy Rate Through Delimiters Identification

Cleber Zanchettin, George D. C. Cavalcanti, Eduardo F. A. Silva, Byron L. D. Bezerra, Rodrigo C. Dória, Juliano C. B. Rabelo

Improving the Courtesy Amount Recognition Accuracy Rate Through Delimiters Identification

Cleber Zanchettin[1,2], *George D. C. Cavalcanti*[1,2], *Eduardo F. A. Silva*[1], *Byron L. D. Bezerra*[1,2], *Rodrigo C. Dória*[1] *and Juliano C. B. Rabelo*[1]
AiLeader Technologies - Recife - Pernambuco - Brazil[1]
Federal University of Pernambuco (UFPE) - Centro de Informática (CIn)[2]
Recife - Pernambuco - Brazil
{cz,gdcc,efas,bldb,rcd,jcbr}@aileader.com.br, {cz,gdcc,bldb}@cin.ufpe.br

Abstract

This paper deals with automatic recognition of real bank checks. A new approach is proposed to read the numerical amount field from bank checks, considering the numeric value and the different delimiters existent in that field. The proposal combines different neural networks classifiers to perform the recognition. Experimental results have shown that this approach is robust and efficient for automatic recognition of real Brazilian bank checks.

Keywords: Automatic bank-check reading, Character and symbol recognition, courtesy amount recognition.

1. Introduction

Automatic check processing is a real banking industry interest, because great part of the checks are manually processed. This procedure involves high costs due to labor-consuming operation, since there is a high volume of this kind of documents.

A bank check image could be divided in the following fields: courtesy amount, legal amount, date, signature, MIRC CMC-7, account number, check number and other text fields, besides symbols and graphics, as shown in figure 1. The most important step to automatize the bank check reading process is the recognition of the courtesy amount field.

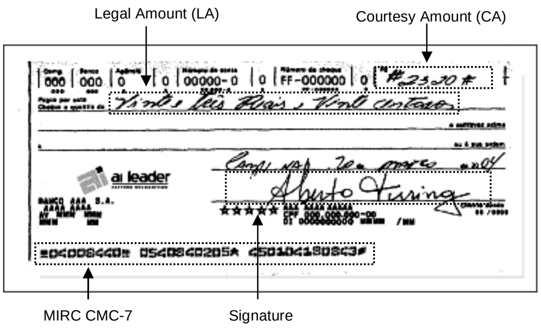

Figure 1. An example of an artificial check image. Some fields are highlighted.

A major characteristic of check amounts is the extensive use of delimiters as suffixes or prefixes of the amounts [8]. However, it is important to note that this is a cultural issue. While in Brazil the use of delimiters are very common, that does not hold true in countries like the United States. Palacios et al. have attested that, in average, 36% of the Brazilian checks contain delimiters. However, the real bank check database used in this work contains, approximately, 90% of the courtesy amount images with at least one delimiter. In their study, Lee et al [4] assumed that only digits, commas and periods were present in a numeric amount. Moreover, they assumed that detecting symbols like ”$”, ”#”, ”(” and ”)” in a digit amount is by no means trivial. Palacios et al [8], developed a Multi-Layer Perceptron classifier (MLP) with 11 outputs corresponding to the digits from 0 to 9, plus the symbol ”#”. However, other kinds of delimiters are commonly used.

In this paper, we investigate the problem of recognizing delimiters in courtesy amount (CA) fields and some alternatives are proposed to cope with that problem. To the best of our knowledge, this is the first work that deals specifically with the problem of courtesy amount delimiters recognition. And in some countries, like Brazil, telling delimiters from digits is an essential step to the correct classification of the courtesy amount field.

The remainder of the paper is organized in the following sections. In section 2 some aspects of the used database are discussed. Section 3 presents feature extraction and classification procedures. The experimental study and the conclusions are described in sections 4 and 5, respectively.

2 Brazilian checks difficulties

Brazilian bank checks have some characteristics that make the recognition process of the courtesy amount a challenging problem. These characteristics could be divided in two: i) one behavioral, because many bank account holders write delimiters, in the courtesy amount field, aiming to protect their checks against possible frauds, besides that, there is no roadmap to be followed in the courtesy amount fulfilling; ii) one associated to the physical aspect of the image that comes from the bank, such as security lines and some figures on the background.

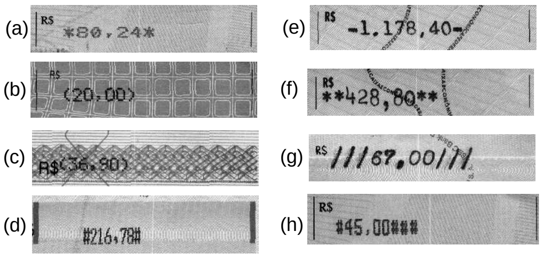

Figure 2. Some kinds of delimiters used on Brazilian bank checks.

In Figure 2, it is possible to observe some kinds of delimiters used in Brazilian bank checks. In general, it is common to see one delimiter at the beginning of the CA and the same one finishing the CA, like in Figure 2(a). However, variations are not prohibited, like the ones listed here: no use of delimiters; many delimiters at the beginning and none at the end, or vice-versa; different delimiters at the beginning and at the end of the CA. The most common printed delimiters are: "#", "*", "=", "(", ")" , "R", "$" and "X".

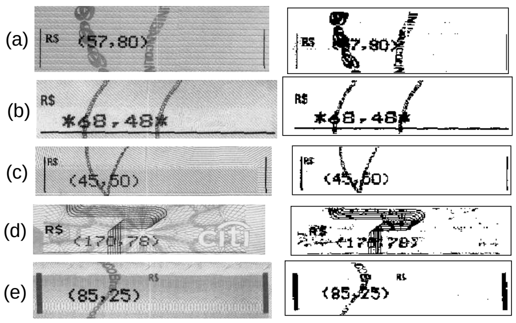

Figure 3. Some kinds of security lines of Brazilian bank checks. Gray-level images (left). Black and white images using Otsu's method (right).

Most of the Brazilian banks add a security line at the right side of the check. That is a sinuous and randomly-generated line, which crosses through many fields, such as courtesy amount, legal amount, payee's name, date and signature. The aim of these lines is to avoid falsifications. However, like it is shown in Figure 3, when one of these lines crosses through the courtesy amount, it compromises the performance of the recognition systems. Because after the binarization process, it is difficult to discriminate which part belongs to the security line and which part belongs to the courtesy amount symbols.

3 Methods

3.1 Feature Extraction

One of the most important step for the development of an optical character recognition engine is the choice of feature extraction techniques. These features must be as font-type-independent as possible for uniquely identify different characters. In our study, these characters are the numbers and delimiters.

For handwritten character recognition, directional, statistical and structural features have been used efficiently in many experiments [11, 6, 5, 7]. Usually, machine-printed character recognition is not so difficult as handwritten character recognition, because there are not many character variations for real use applications.

In this paper, our purpose is to classify delimiters or simply identify that one delimiter is not one number from 0 to 9. For this reason, we have chosen a feature set, also used for handwritten character recognition, that are able to classify characters into 2 groups: Group 1: 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9; and Group 2: #, *, =, (,) , R, $ and X.

This feature set is composed by 125 (2 + 78 + 45) different features from three different techniques: Geometric Moments, Concavity Measurements and Shape Representation of Profiles.

3.1.1 Geometric Moments

Geometric Moments have been used a lot in pattern recognition tasks [2, 6] due to its power of extracting invariant features. These invariant features are usually related to scale, orientation and translation. This is very important in handwritten recognition because one letter must be recognized, even when it is under translation, orientation or scale transformations.

Although writing in a bank check is a controlled task. It may be found distorted digitalized images. This distortions make character classification a more difficult task and for this reason moments may have important use.

Low order moment values described particular geometric features, such as object areas, center of mass coordinates and others [2]. In this work, we have chosen center of mass as an important feature for character separation. We have noted that some characters that are usually misclassified have different center of mass, as it happens to 5 and 9 characters.

3.1.2 Concavity Measurements

The basic idea of concavity measurements is to identify different types of concavities one character has [7]. To do this, from each white pixel in the character image, the system searches in the 4-freeman directions until it reaches a black pixel or until it reaches image limits. There are four main elements of concavities to be found: i) no-concavity (no black pixel is found in four directions), ii) two-directions concavities (two black pixels are found in two consecutive directions), iii) three-direction concavities (three black pixels are found in three consecutive directions) and iv) closed-loops (black pixels are found in four directions). More details about the technique can be found in the Oliveira et al.'s article [7].

3.1.3 Shape Representation of Profiles

This feature is usually classified as a Structural Feature and they are insensitive to character shape variation [5]. The profile shape of character mainly reflects concavities and convexities [5]. We calculate them in our work using object boundary and corresponding convex hull. To extract profile shape features, the left and right apparent boundary are obtained and their convex hull are computed. The profile shape is represented as follows:

$$Dl(y) = Pl(y) - Cl(y). \tag{1}$$

$$Dr(y) = Cr(y) - Pr(y). \tag{2}$$

Where $Pl(y)$ and $Pr(y)$ are the left profile and the right profile respectively, and $Cl(y)$ and $Cr(y)$ the corresponding convex hull. $Cr(y)$, $Dl(y)$ and $Dr(y)$ are used as structural features for recognition and the total number of features is 45. The complete feature techniques can be found in [5].

3.2 Auto-associative Neural Networks

Vasconcelos et al. [12] studied the reliability of feed-forward neural networks with respect to the rejection of patterns not belonging to the defined training classes. In special, the following nets were investigated: the MLP network, the MLP with Gaussian activation function (GMLP) and the Radial Basis Function (RBF). It can been proven that neural networks based on radial basis functions can also provide closed separation surfaces and, consequently, appear to be more adequate than MLPs based on sigmoidal or radial basis functions.

Gori et al. [1] also investigated the ability of multilayer perceptrons in the creation of bounded domains in the pattern space and, in particular, they related that analysis to applications of pattern verification. They have proven that, regardless of the function used in the processing units, architectures with less units than inputs in the first hidden layer can not yield closed separation surfaces. When using more hidden units than inputs, they have also proven that an MLP can either create open or closed surfaces. Moreover, no choice of the sigmoidal function in the neurons can transform open separation surfaces into closed separation surfaces, and deciding whether or not they are open is NP-hard.

There are alternative approaches to pattern verification using neural networks which do not present the problems pointed out above. For instance, in MLPs used as auto-associators [3], the weights are adjusted so as to copy the inputs to the outputs, which can profitably be used for designing pattern verification systems. For each pattern, the verification criterion is based on the input/output Euclidean distance, that is, given a threshold t, a pattern x is accepted if and only if $|f(x) - x| \leq t$. The basic idea is that only the patterns of the class used for training the auto-associator are likely to be reproduced with "enough" approximation at the output. It has been shown that in this case the separation surfaces are always closed [1].

The threshold t is defined by a circle in Figure 4. It represents the decision area of this kind of neural network. That figure is a graphical representation of a "#" verifier. The main idea is to create a classifier that recognizes only the pattern of the class that it was previously trained for (in this case, only "#" patterns). All other patterns, like digits "1", "4" and "7", and other delimiters must be rejected. In this verification scheme, two types of errors are possible: false negative and false positive. The first kind of error occurs when a symbol is classified as an impostor being a genuine pattern. This is expressed by the pattern "#" that is outside the circle. On the other hand, a false positive can happen when a symbol is incorrectly classified as a genuine pattern. In this case, like shown in the referenced figure, digit "4" is said to belong to the "#" class.

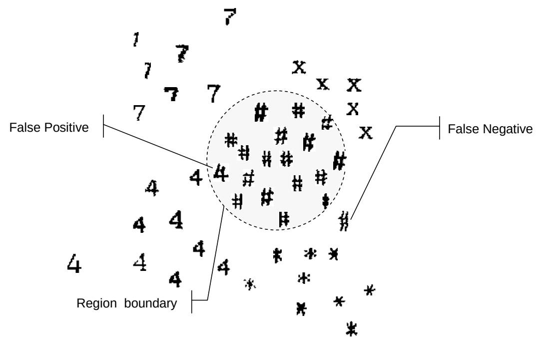

Figure 4. Auto-associative neural network decision area.

It is important to observe that some errors do not influence the performance of the system, like misclassifying a "#" as a "*" or a "X". In that situation, the system will change one delimiter by another, but the courtesy amount value does not depend on any specific delimiter. However, when a digit (e.g., "4") is misclassified as another digit (e.g., "7"), a false positive for the courtesy amount recognition will be generated.

4 Experimental Results

4.1 Database

The bank checks in Brazil can be fulfilled by handwriting or through some printer device. The real Brazilian check database used contains a total of 10,871 images, from which approximately 89% are cursive handwriting. In the experiments done in this article, 1,146 courtesy amount printed images were used. In Table 1, it is possible to observe the number of images used to test the system per class - 10 digits (0-9) and 8 delimiters("#", "(", ")", "*", "=", "R" , "$" and "X"), as shown in Figure 5. From this total of printed checks, only 9.7% do not have delimiters in the courtesy amount field. That is an important piece of information, because it shows the importance of discriminating delimiters from digits for the application in question.

4.2 Neural Network Training Methodology

The database for training and test the artificial neural network was divided as follows: 50% of the patterns from each class were randomly assigned to the training set, 25% were assigned to the validation set, and 25% were re-

Table 1. Digits and Symbols used to test the system.

Symbol	Quantity	Percentage (%)
0	2500	7.2558
1	2500	7.2558
2	2500	7.2558
3	2500	7.2558
4	2500	7.2558
5	2500	7.2558
6	2390	6.9366
7	2135	6.1965
8	2200	6.3851
9	2078	6.0311
#	1920	5.5725
(	2349	6.8176
)	2342	6.7973
*	2500	7.2558
=	372	1.0797
R	406	1.1783
$	537	1.5586
X	226	0.6559
Total	34455	100

served to the network test, as suggested by Proben1 [9]. The patterns were normalized to the range [-1, +1], and the processing units were implemented by hyperbolic tangent activation function. To the auto-associative neural networks gaussian activation functions were used. The neural network contains all possible feedforward connections between adjacent layers, and no connection between non-adjacent layers. In training, different topologies were tested to find the best neural network architecture to the problem.

The training algorithm used is a version of the Resilient backpropagation method described in [10]. For each topology, 10 runs were performed with 30 different and random weight initializations.

Considering N_C classes in the database, the *true class of the pattern x* from the training set P_t is defined as:

$$\gamma(x) \in \{1, 2, ..., N_C\}, \forall x \in P_t \tag{3}$$

In the neural network training the *winner-takes-all* classification rule was used. For this reason, the number of output units is equal to the number of classes (N_C).

Being $o_k(x)$ the output value of the output unit k for the pattern x, the *class assigned to pattern* x is defined as:

$$\phi(x) = arg \max o_k(x), \forall x \in P_t, k \in \{1, 2, ..., N_C\} \tag{4}$$

The *network error for the pattern x* is defined as follows:

$$\varepsilon(x) \equiv \begin{cases} 1, & if \quad \phi(x) \neq \gamma(x), \\ 0, & if \quad \phi(x) = \gamma(x). \end{cases} \tag{5}$$

Therefore, the classification error for the training set P_t, which represents the percentage of incorrectly classified training patterns, can be defined as:

$$E(P_t) \equiv \frac{100}{\#P_t} \sum_{x_e P_t} \varepsilon(x) \tag{6}$$

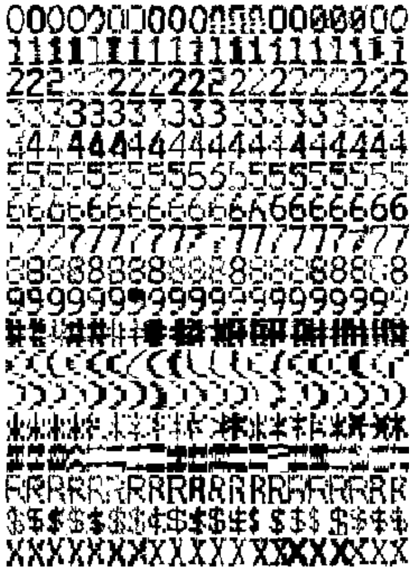

Figure 5. Some samples of digits and delimiters of the symbol database.

where $\#P_t$ is the number of patterns in the set P_t.

In the experiments, the neural network training stops if:(1) the GL_5 criterion defined in Proben1 [9] is satisfied twice (based on the classification error, to avoid initial oscillations in validation errors); (2) the training progress criterion defined in Proben1 [9], with $P_5(t) < 0.1$ is satisfied; or (3) the maximum number of 5,000 iterations is reached. The GL_5 criterion is a good approach for avoiding overfitting to the training set. The classification error for the validation set P_v is given by $E(P_v)$, which is calculated according to the Equation 6. In this way, denoting by $V(k)$ the classification error $E(P_v)$ at iteration $i = kI_T$, $k = 1, 2, ..., \frac{I_{max}}{I_T}$ the *generalization loss* parameter (GL) is defined as the relative increase of the validation error over the minimum-so-far (in percent):

$$GL(k) \equiv \left(\frac{V(k)}{\min_{j \leq k} V(j)} - 1 \right) \tag{7}$$

The GL_5 criterion stops the execution when the parameter GL becomes higher than 5% [9].

4.3 MLP (digits)

The first experiment aimed to evaluate the accuracy rate of a MLP when classifying the symbols presented in Table 1. A MLP neural network with 10 outputs was constructed, where each one of the outputs corresponds to the digits from "0" to "9".

The neural network was trained with only 10 classes, the digits from 0 to 9. Thus, knowing that the test will be done with delimiters too, a threshold to reject some spurious patterns is needed. After some preliminar experiments, the threshold was defined as 0.7. With this configuration the neural network achieved the following rates: 67.54% of accuracy rate, 19.44% of rejection and 13.02% of incorrect classifications. Great part of those errors can be attributed to delimiters that were classified as digits, for example: 751 occurrences of the delimiter "(" were classified as class "1" or "7"; 1,276 occurrences of the delimiter "$*$" were incorrectly classified as the digits "4", "8" or "9"; and the delimiter "R" was misclassified 253 times as the digit "9". Only those three cases represent, approximately, fifty percent of the incorrect classification rate.

Testing that network only with the digits from "0" to "9", the rates are much better: 97.76% of correct classification, 0.39% of false positive and 1.84% of rejection. These results show that the neural network was not prepared to treat with spurious patterns, even when a threshold is adjusted.

Applying this MLP to the database of courtesy amount images, an accuracy rate of 37.78% was achieved. In other words, a total of 433 courtesy amount images were correctly classified from a universe of 1,146 images.

4.4 MLP (digits plus delimiters)

The results found with the neural network in the previous section, when the neural networks were trained only with the digits from 0 to 9 were not good. The next experiment aims to minimize the percentage of errors through increasing the number of outputs to the total number of symbols, i.e, 18 (ten digits and eight delimiters).

With this architecture, the neural network achieved the following rates over de symbols listed in Table 1: 97.79% of accuracy rate, 1.91% of rejection and 0.30% of incorrect classification. From the total rejected images, almost 30% were due to the digit 8 and the delimiter *. Once again the threshold adopted was 0.7.

When this neural network was used to classify the courtesy amount images, the accuracy rates were much better than the previous neural network (the one using only 10 outputs), as it achieved 63.87% of accuracy rate (732 images from 1,146 were correctly classified).

4.5 Neural Combination Architecture

The correct classification probability of all symbols in a courtesy amount is given by the product of the probability of each one of the symbols, i.e., $\prod_{i=1}^{n} P_i$, where n is the number of symbols in the courtesy amount and P_i is the correct classification probability of a specific symbol. In average, a courtesy amount is formed by 6 symbols (numbers and delimiters). Thus, based on the MLP trained in the previous section, MLP (digits plus delimiters), that has achieved an individual correct recognition rate of 97.79%, this will generate a much smaller probability of courtesy amount correct classification, in the case: 87.45%. An error will happen when at least one of the six symbols of the CA is incorrected classified. This will give a false positive rate of 1.62%. The remaining 10.93% come from rejected patterns.

In fact, despite of the correct classification probability around 87% for six digits, the actual rate achieved by the MLP (digits and delimiters) in the classification was, approximately, 64%. Aiming to improve this classification rate, a structure to confirm each recognized digit was developed, as shown in Figure 6.

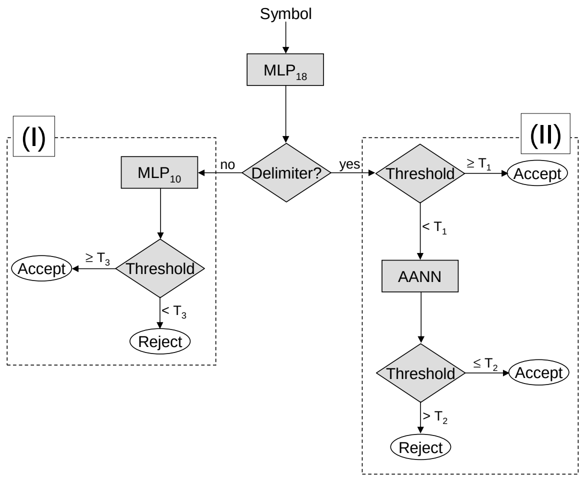

Figure 6. Block diagram of the proposed neural system architecture. (I) Confirmation structure for digits from 0 to 9. (II) Confirmation structure for delimiters. MLP$_{10}$ - MLP for digits only. MLP$_{18}$ - MLP for digits and delimiters. AANN - Auto-associative Neural Network.

The main idea was to create confirmation structures: one for the digits and another one to the delimiters. First, a segmented symbol is given to the MLP$_{18}$ - MLP (digits and delimiters). That neural network is the root of the architecture because it has achieved a good rate in the task of discriminating digits and delimiters. After that, depending on the class of the symbol (if it is a digit or a delimiter), it is given as input to one of the two confirmation structures.

If MLP$_{18}$ recognizes the pattern as a delimiter, and if the threshold from this neural network is greater than a predefined value T_1, the delimiter is accepted. Otherwise it is passed through a confirmation neural network, which is defined as an auto-associative neural network, $AANN$, like in Figure 6. There is one of those neural networks for each one of the delimiters. It is a one-class classifier and has the advantage of being specialized in verifying if the delimiter is the same classified by the MLP$_{18}$. Based on the output of an AANN it is possible to calculate how distant is the pattern from its class. Thus, the delimiter is accepted if its distance is smaller than a predefined threshold T_2, and rejected otherwise.

On the other hand, if MLP$_{18}$ recognizes the pattern as a digit, it is given as input to the MLP$_{10}$ (MLP digits), and the answers of the neural networks, MLP$_{18}$ and MLP$_{10}$, are combined using the product rule. After that, if the joint threshold is smaller than a predefined threshold T_3 the digit is rejected, what causes the rejection of the whole CA. Otherwise, the digit is accepted and the CA recognition continues with the classification of the next symbol, if it exists.

The results of the architecture showed in Figure 6 using the symbols database were similar to those achieved when only MLP_{18} was used. The rates are: 97.71% of correct classification, 0.33% of false positives and 1.96% of rejection. When a CA is correctly classified, it means all symbols in that CA were correctly classified. Analyzing the results of the CA images, the proposed architecture achieved 67.89% of accuracy rate. That result is, approximately, 4 percentile points higher than the result found using only MLP_{18}.

It was observed that in 146 incorrect classified CA images there are problems that need to be fixed before classification, such as: i) incorrect binarization procedure (72 images); ii) deformation of the CA digits caused by the security line (63 images); and, iii) problems in high-level segmentation, in which the CA area in the checks was incorrectly found (11 images). Discarding those images from the database, the accuracy rate of the system in classifying CA images would be improved to 77.80% of correct classification.

Table 2. Accuracy rates of the architectures.

	Symbol Recognition			CA Recognition
	Accur.	Error	Rejec.	Accuracy
MLP_{10}	67.54	13.02	19.44	37.78
MLP_{18}	97.79	0.30	1.91	63.87
System	97.71	0.33	1.96	67.89

Table 2 summarizes the results found with the tested architectures. It is possible to observe that MLP_{18} and the system architecture, described in Figure 6, have similar rates in the task of classifying digits and delimiters. However, due to its confirmation structures, the system achieved better rates than MLP_{18} in the recognition of courtesy amount fields.

5 Conclusions

An important phase of the systems that deal with courtesy amount images is to discriminate digits from delimiters. This task is crucial to achieve good results because: i) most of the printed courtesy amount in Brazilian checks use delimiters (in our database, more than 90% of them have delimiters); ii) many delimiters may be misclassified as digits from ”0” to ”9”. Some classes in particular presented similarities, like those formed by the digits ”7” and ”4”, and the delimiters ”#” and ”*”. In the same way, the delimiters ”(” and ”)” are commonly misclassified as the digits ”1” and ”7”.

The proposed neural architecture to recognize courtesy amount achieved good recognition rates through the use of two confirmation structures: one that aims to confirm if the preliminary class of the digit is correct and another that uses an auto-associative neural network to confirm the class of a delimiter. Another factor that makes our experiments more reliable comes from the fact that the database used was not artificially produced in a laboratory. The system was tested over a set of real Brazilian bank check images.

The performance of the proposed system could be improved by the construction of a context analyzer. It would be responsible for establishing some rules, like the one that does not allow a delimiter to be rounded by two digits. Future work needs to consider the context and other features of real bank checks.

References

[1] M. Gori and F. Scarselli. Are Multilayer Perceptrons Adequate for Pattern Recognition and Verification? *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 20(11):1121–1132, 1998.

[2] M. Hu. Visual pattern recognition by moment invariants. *IEEE Transactions on Information Theory*, 8(2):179–187, 1962.

[3] M. Kramer. Autoassociative Neural Networks. *Computers Chem Engng*, 16(4):313–328, 1992.

[4] L. Lee, M. Lizárraga, N. Gomes, and A. Koerich. A prototype for brazilian bankcheck recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 11(4):549–569, 1997.

[5] C. Liu, Y. Liu, and R. Dai. Preprocessing and statistical/structural feature extraction for handwritten numeral recognition. Progress of Handwriting Recognition, A.C. Downton and S. Impedovo eds., World Scientific, 1997.

[6] C. Liu, K. Nakashima, H. Sako, and H. Fujisawa. Handwritten digit recognition: benchmarking of state-of-the-art techniques. *Pattern Recognition*, 36(10):2271–2285, 2003.

[7] L. Oliveira, R. Sabourin, F. Bortolozzi, and C. Suen. Automatic recognition of handwritten numerical strings: A recognition and verification strategy. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 24(11):1438–1454, 2002.

[8] R. Palacios, A. Gupta, and P. Wang. Feedback-based architecture for reading courtesy amounts on checks. *Journal of Electronic Imaging*, 12(1):194–202, 2003.

[9] L. Prechelt. Proben1: A set of neural network benchmark problems and benchmarking rules. Technical Report 21/94, 1994.

[10] M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In *Proceedings of the IEEE International Conference on Neural Networks*, pages 586–591, San Francisco, CA, 1993.

[11] M. Shridhar and A. Badreldin. Recognition of isolated and simply connected handwritten numerals. *Pattern Recognition*, 19(1):1–12, 1986.

[12] G. Vasconcelos, M. Fairhurst, and D. Bisset. Investigating feedforward neural networks with respect to the rejection of spurious patterns. *Pattern Recognition Letters*, 16:207–212, 1995.