A Fuzzy System Specialized in Character Recognition

Jomar Fernandes Nascimento and Francisco José Gomes

Abstract -- **A new method for character recognition using Fuzzy Logic, Levenshtein Distance, Algorithms for Bit map Reduction and auto-learning techniques are presented. This method can recognize not only letters but based on the analyses of the character's morphology it is also possible to recognize numbers, symbols, Chinese characters. It was implemented and the results were stimulant, validating the applicability of Fuzzy Logic in applications where pattern selection is needed. The auto-learning tecniques shows itselves very important at the training process and forward utilization of the system, decreasing the error tax to very short values, without changing the recognition speed.**

Index Terms -- **Character recognition, Fuzzy Logic, Levenshtein Distance, Auto-learning**

I. INTRODUCTION

The subject of character recognition has been receiving considerable attention in recent years due the increasing dependence on computer data processing. Many methods were developed and used today in character recognition applications.

Some of then are: Statistical analyses [1], Neural Networks [2], among others. Each method is evaluated by two parameters: the error tax and the recognition speed whose measures qualify and determine it's kind of application.

The main purpose of this work is to develop a new character recognition method that presents the relation error tax versus recognition speed as better as the other existing methods. For that, some emerging technology's were joined to get the expected result.

The recognition is done starting with the analyses of the character and achievement of its main characteristics bound to its morphology and then by the comparison with information saved in the knowledge base the nearest one is chosen.

The selection process tends to be very subjective, because the characteristics are not the same. It is necessary to classify the similitude in groups and inside this group its degree of membership. This is achieved by the Levenshtein Distance and Fuzzy Logic in the evaluation and determination process.

In case of any mistake in the selection or no best case is found, the method permits the correction or the insertion of the correct or new information in the knowledge base, decreasing progressively the error tax.

This paper is organized in the following way: chapter II describes the technologies and methodologies used, chapter III describes the algorithms and chapter IV the implementation of the method. The results and conclusions are presented respectively in chapters V and VI.

The authors are with the Energy Departament of Engineering Faculty of Federal University of Juiz de Fora.
PO Box 422 - Phone (032) 229-3424
Juiz de Fora - MG - Brazil - ZC 36001-970

II. METHODOLOGY

A. FUZZY LOGIC

The Fuzzy logic was first broached by Lotfi A. Zadeh, a professor at the University of California at Berkeley, in a 1965 paper on fuzzy sets [3].

Zadeh introduced in 1973 the concept of a linguistic variable, or one whose values are words, and not numbers. Thus, the linguistic variable "size" might have the values "large," "not very large", "small", and so on. In combination with the notion of a fuzzy IF-THEN rule -- for example, "if pressure is very high, then volume is very low"--the concept paved the way for applying the theory to real tasks [4].

Fig. 1 shows the data flow on a fuzzy-system.

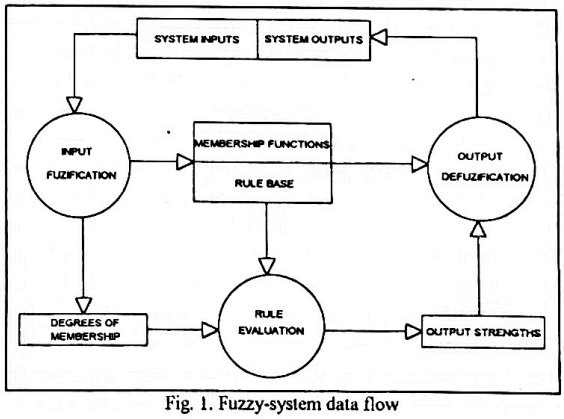

Fig. 1. Fuzzy-system data flow

B. LEVENSHTEIN DISTANCE

There are many applications where it is necessary to compare sequences of values and obtain a measure that defines the difference degree between then; voice recognition and speech processing, geological analyses, fingerprint analyses, gas chromatography among others [5]. For each of these diverse applications a number of sequence comparison algorithms are in use and most of them stem from the basic work of V. I. Levenshtein, published in Russian in 1965 [6].

Levenshtein introduced two measures of difference between two strings (sequence of values). One that is the smallest number of substitutions, insertions and deletions, required to transform one string to another. The other is similar, except that substitutions are not allowed. Both metrics have been called the "Levenshtein Distance." In this paper, we used only the first one.

We can define the Levenshtein Distance between two sequences of values as a recursive relation resulting in a weighted sum. At each stage, represented by cell M[i,j], the minimum distance between partial strings A[1..i] and B[1..j] is the three-way minimum of the accumulated distance in predecessor stages plus the cost of moving from those neighboring cells over the current cell -- the cost of deleting, inserting, or substituting the corresponding characters.

$$D(a_i, b_j) = \min \begin{cases} D(a_{i-1}, b_j) + w(a[i],\phi) \\ D(a_{i-1}, b_{j-1}) + w(a[i],b[j]) \\ D(a_i, b_{j-1}) + w(\phi,b[j]) \end{cases}$$

where:

$D(a_i, b_i)$	is the string distance from string a of length i to string b of length j,
$w(a[i],\phi)$	is the cost of deleting character a[i],
$w(a[i],b[j])$	is the cost of substituting a[i] with b[j], and
$w(\phi,b[j])$	is the cost of inserting character b[j].

Let's see an example of the comparison between the sequences (001234554) with (012345678):

001234554	Delete the first 0
01234554	Substitute the second 5 for 6
01234564	Substitute the second 4 for 7
01234567_	Insert an 8 at the last position
012345678	

With four operations we can transform the first sequence (001234554) in the second one (01234567), so the Levenshtein Distance between then is 4 (four).

III. ALGORITHMS

A. FULL VIEW OF THE METHOD

Fig. 2 shows the method algorithm. It is an implementation of finding the best case problem, and fuzzy logic tells when the best case is achieved.

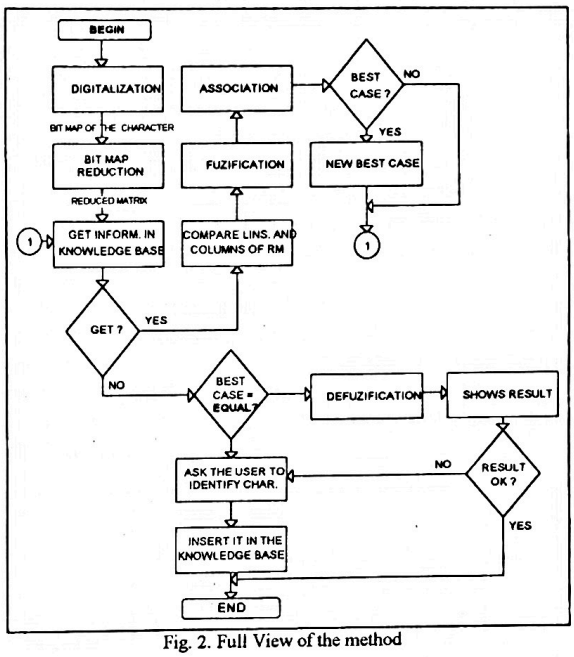

Fig. 2. Full View of the method

B. LOOSELESS BIT MAP REDUCTION

When we make an analyze of a character as an input of the system, it usually is provided from a digitizer or a scanner with many pixels. Many of these pixels have redundant information and must be eliminated. There are some pixels that result from then introduction of noise in the process. The elimination of these pixels (the redundant ones and the noise) must be done in way that the loose of information is the minimal as possible.

We achieve these results analyzing the superficial distribution of the pixel, dividing the image in quadrants, lines and columns. Relationing the distribution of the pixels in each quadrant, lines and column with the whole image, we can localize the main pixels' concentrations and with this information compose a new image, smaller than the first one that can be translated in a little matrix of order 16 that we called Reduction Matrix.

This matrix will represent the character in the knowledge base. The advantage of using the Reduction Matrix instead of the full image itself is that it requires less computational effort and occupies less storage

space in the computer permitting the system to have a large amount of information without requiring so many system resources.

The reduction matrix can be expressed as:

$$U(i,j)=\frac{\sum_{y=i*SliceY}^{(i+1)*SliceY}\sum_{x=j*SliceX}^{(j+1)*SliceX}\varphi(x,y)}{\sum_{0}^{LenY}\sum_{0}^{LenX}\varphi(x,y)}\cdot 100\%$$

$$M_{(i,j)}=\begin{cases}0, \text{ if } U(i,j) > FatF\\ 1, \text{ if } U(i,j) <= FatF\end{cases}$$

where:

- U(i,j) Percentual distribution of the pixels in the quadrant (i, j) relationed with the full image
- $M_{(i,j)}$ is the element of line i and column j of the reduction matrix,
- φ(x,y) Binary function, return 1 if a pixel exists in the coordinates (x, y), 0 on the other hand,
- LenX Width and Height in pixels of the character
- LenY image,
- SliceX Width and height of the quadrant being
- SliceY analyzed,
- FatF Filter factor. Through this factor it's possible to eliminate aleatory noise in the reduction process. Higher values eliminate concentrate noise in regions and lower values eliminate noise disperse in the bit map.

Fig. 3 and 4 shows a bit map reduction of a print and a handprinted character. They show the bit map, the reduced bitmap and the reduction matrix.

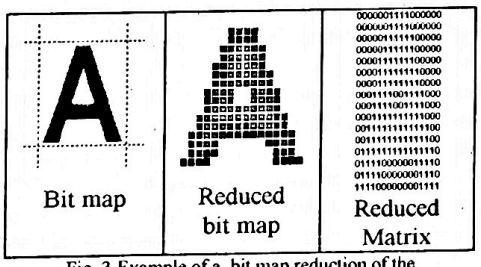

Fig. 3 Example of a bit map reduction of the uppercase letter A (Font Times New Roman)

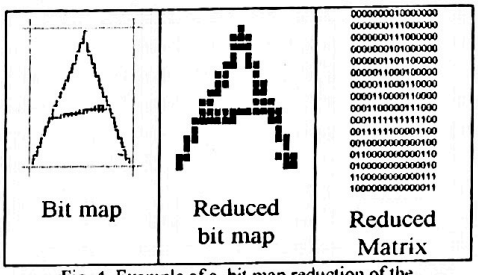

Fig. 4. Example of a bit map reduction of the uppercase letter A (Hand printed)

C. FUZZY SETS AND RULE BASE

The Fuzzy Logic was used in the comparison of two characters, to aply the same subjective as the human reasoning.

It is a SISO system (single input single output). The input is the Levenshtein Distance of each column and each line of the reduction matrix of the character being recognized and the character on the knowledge base.

The Fuzzy Sets used in the input fuzzification is showed in Fig. 5.

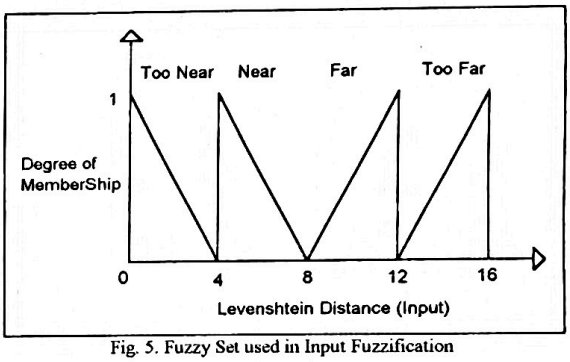

Fig. 5. Fuzzy Set used in Input Fuzzification

The rule base is showed in Fig. 6. Each entry is associated with the last one in order to produce a final output: the similitude degree of the two characters being compared.

	Too Far	Far	Near	Too Near
Too Far	D	D	D	D
Far	D	D	D	S
Near	D	D	S	S
Too Near	D	S	S	E

Fig. 6. Rule Base-D=Diferent, S=Similar, E=Equal

The rule base tends to reinforce the more favorable result. In other words, if the comparison of the lines and columns tend to be "Near" or "Too Near," the output tends to be "Similar" or "Equal," with a degree of membership that depends on the degrees of membership

of each entry associated. On the other hand if the comparison of the lines and columns tends to be "Far" or "Too Far," the output tends to be "Different."

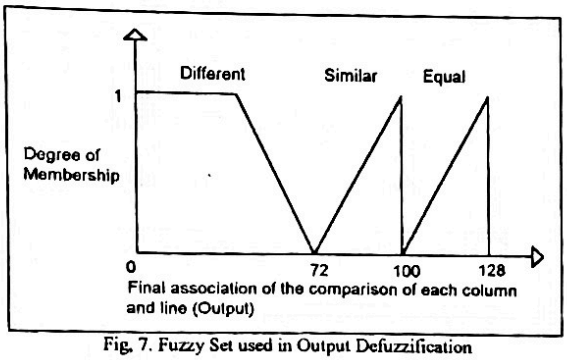

Fig. 7. Fuzzy Set used in Output Defuzzification

The output Fuzzy Set is Showed in Fig. 7. It is possible to see that modifing the clusters the system becames more or less selective.

For the definition of the Fuzzy Sets and rule base presented here, we used heuristic modeling, based on tests and results obtained from the system for each Fuzzy set we try.

D. AUTO-LEARNING TECNIQUES

To get a constant increase in the hit tax, it became necessary to include an auto-learning module to make possible the addition of information and correction on the knowledge base.

Each time the system can't identify the character it asks the user to identify it and then it saves the information in the knowledge base.

When the system makes a mistake in the identification process, the user can correct the assertion and the information is saved in the knowledge base.

This is especially interesting in applications where the user inserts information in the system through an optic pen (pen computers). The system learns the way the user writes and after a small training time the error tax tends to zero.

The concept of auto-learning was implemented using a data base of patterns. Each entry in this data base contains the reduction matrix and the identification of the character it corresponds to.

Fig. 8 shows the record definition for each entry in the knowledge base.

Character Identification
Reduction Matriz
Pixel Distribution per Line
Pixel Distribution per Column

Fig. 8. Record Definition in the Knoledge Base

There are several ways to access the information stored in the knowledge base and we choose the simplest one: sequential search. This kind of access method is preferable because the entries in the knowledge base are small when measured in bytes and can be searched quickly.

IV. IMPLEMENTATION

This method was implemented to obtain an integrated environment specialized in pattern recognition and making also possible to analyze the characters and its properties, its superficial distribution of pixels by quadrant, lines and columns and its reduced bit map.

We used a G.U.I. interface to make it easy to use and it was implemented entirely in C++ programming language and works on any computer and operational system that supports this language. Fig. 9 shows its main screen.

To digitize the characters, we use a GENIUS 300 DPI scanner and to test the system we used a 486DX 33MHz with 4MBytes of memory.

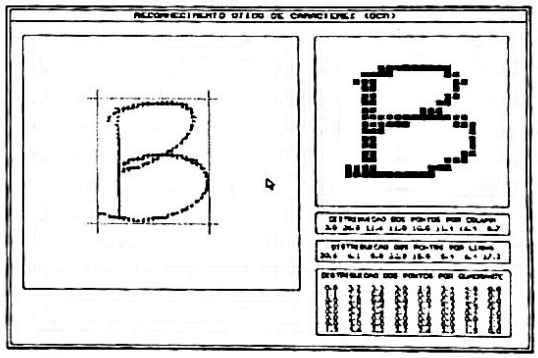

Fig. 9. Main screen of the System.

V. RESULTS

To test the method and the system that was developed we made two kinds of tests: printed character recognition and hand printed character recognition with the auto-learning module on and off.

In the two sequences of tests, we introduced in the system one sample of each character of the alphabet. Usually we will use several samples, but our main purpose was to see the method resolution capability.

Fig. 10 and 11 shows the results of the test when auto-learning is OFF.

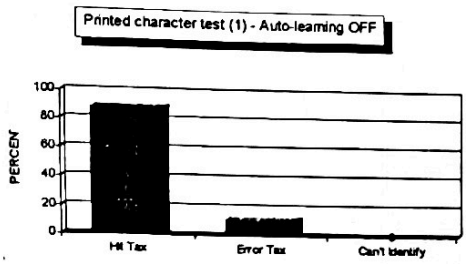

Fig. 10. Printed test results - Auto-learning OFF

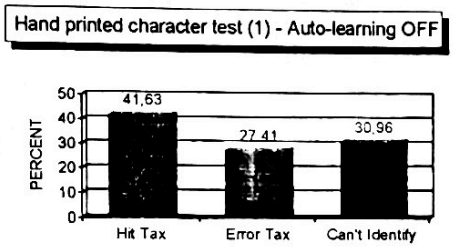

Fig. 11. Hand Printed test results - Auto-learning OFF

Fig. 12 and 13 show the results when the auto-learning module is ON. It is possible to see that the Hit Tax is increasing nearly the same rate that the error tax is decreasing.

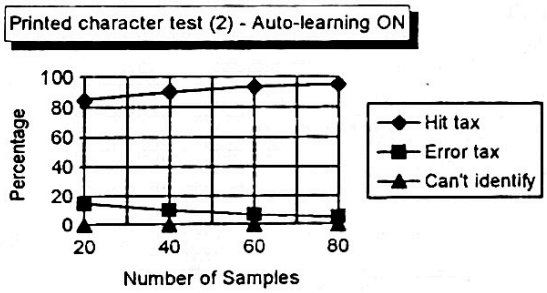

Fig. 12. Printed test results - Auto-learning ON

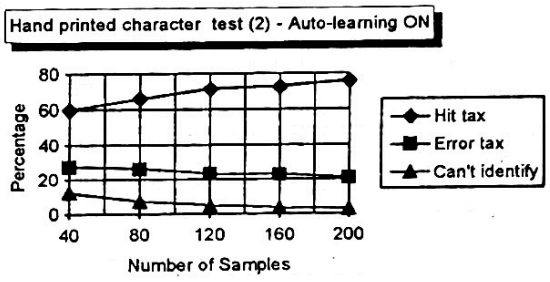

Fig. 13. Hand Printed test results - Auto-learning ON

The system presents an excelent performance in the relation between Hit Tax versus Recognition Speed. Although it isn't presented any information on the recognition speed, the system showed no measurable delay for the recognition.

Fig. 14 shows some samples used in the hand printed test.

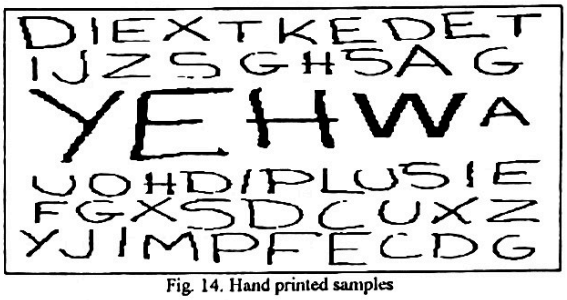

Fig. 14. Hand printed samples

VI. CONCLUSIONS

One of the purposes of this work was to validate de behavior of Fuzzy Logic when applied to character recognition specially pattern selection and the results were stimulants. A more concise mathematical model and a MISO system (Multiple Input Single Output) would conduct to better results.

The application of the Levenshtein Distance, implemented in a partial way show itself very important in the determination of the degrees of difference between two lines or columns of the reduced matrix.

We note that the size of the character had no influence in the recognition process because in the bit map reduction, we used the percentual superficial distribution of pixels.

The results were very good when we look at the Relation Hit tax and Recognition Speed, as showed in the figures 10, 11, 12 and 13 at chapter V, but it did not tend to be definitive. They must be analyzed under a qualitative aspect and future enhancements and studies are needed.

VI. ACKNOWLEDGMENT

The authors wish to acknowledge the support for this project by CAPES for the financial support.

VII. REFERENCES

[1] Al-Yousefi, H.; S.S. Udpa, "Recognition of Arabic Characters", Transactions on Systems, Man and Cybernetics, vol. 14, N° 8, 1992.

[2] Stefan Knerr; Léon Personnaz; Gérard Dreyfus, "Handwritten Digit Recognition by Neural Networks with Single-Layer Training", IEEE Transactions on Neural Networks, vol. 3, N° 6, 1992.

[3] L. A. Zadeh, "Fuzzy Sets", Inform. Contr., vol. 8, p. 338, 1965.
[4] Daniel G. Schwartz, "Fuzzy Logic Flowers in Japan", IEEE Spectrum, vol. 29, Nº 7, p. 32, 1992.
[5] Ray Valdés, "Finding String Distances", Dr. Dobb's Journal, Nº 187, p.56, 1992.
[6] V. I. Levenshtein, "Binay Codes Capable of Correcting Deletions, Insertions and Reversals", Doklady Akadaemii Nauk SSR 163, p 845, 165.
[7] R. Colin Johnson, "What is Cognitive Computing ?", Dr. Dobb's Journal, Nº 197, p. 18, 1993.
[8] Shinichi Tamura; Seihaku Higuchi; Kokichi Tanaka, " Pattern Classification Based on Fuzzy Relations ", Transactions on Systems, Man and Cybernetics, vol 1, Nº 1, 1971.
[9] Ron Avitzur, "Your Own Handprinting Recognition Engine ", Dr. Dobb's Journal, Nº 187, p. 32, 1992.
[10] Greg Viot, "Fuzzy Logic in C ", Dr. Dobb's Journal, Nº 197, p. 40, 1993.
[11] Ogê Marques Filho, "Reconhecimento de Caracteres Utilizando Correlação com Máscaras Binárias", 8º Congresso Brasileiro de Automática, 1990.
[12] Michio Sugeno and Takahiro Yasukawa, "A Fuzzy-Logic-Based Approach to Qualitative Modeling", IEEE Transactions on Fuzzy Systems, Vol.1, Nº 1, February 1993.
[13] Yousuf Saifullah and Michael T. Manry, "Classification-Based Segmentation of ZIP Codes", IEEE Transactions on Systems, Man and Cybernetics, Vol.23, Nº 5 September/October 1993.
[14] Earl Cox, "Adaptive Fuzzy Systems", IEEE Spectrum , Vol.30, Nº 2, February 1993.