Character Recognition with Minimum Edit Distance Method

Parul Vashist[1], K. Hema[2]

[1]I.T ,U.P.T.U, GNIOT, Greater Noida
Greater Noida,201308, North East Zone, India
pvashist2009@gmail.com

[2]E.I,U.P.T.U, GNIOT, Greater Noida
Greater Noida,201308, North East Zone, India
hemaeie@gmail.com

Abstract: ***Character Recognition is performed using Multilayer preceptor neural network. Number of hidden layer nodes is determined to achieve high performance back propagations network in the recognition of Characters. ICR software has a self-learning system referred to as a neural network, which automatically updates the recognition database for new handwriting patterns. It extends the usefulness of scanning devices for the purpose of document processing, from printed character recognition (a function of OCR) to hand-written matter recognition. We also compared ICR with OCR and OMR with the Edit distance method refers to the distance in which insertions and deletions have equal cost and replacements have twice the cost of an insertion.***

Keywords*:* Character Recognition, ICR, Feature Extraction, Edit Distance Method

1. Introduction

1.1 Neural Network

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true of ANNs as well.

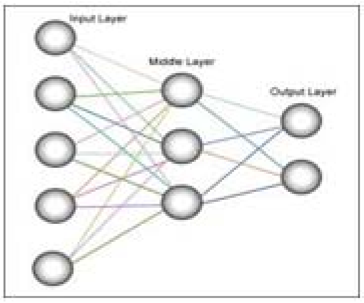

Figure 1: Neural Network

1.2 Why we use neural networks?

Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyses. This expert can then be used to provide projections given new situations of interest and answer "what if" questions.

Other advantages include:

1. Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience.
2. Self-Organization: An ANN can create its own organization or representation of the information it receives during learning time.
3. Real Time Operation: ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability.
4. Fault Tolerance via Redundant Information Coding: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.

1.3 Back Propagation

Back propagation, an abbreviation for "backward propagation of errors", is a common method of training artificial neural networks. From a desired output, the network learns from many inputs, similar to the way a child learns to identify a dog from examples of dogs.

It is a supervised learning method, and is a generalization of the delta rule. It requires a dataset of the desired output for many inputs, making up the training set. It is most useful for feed-forward networks (networks that have no feedback, or simply, that have no connections that loop). Back propagation requires that the activation function used by the artificial neurons (or "nodes") be differentiable. Back-propagation neural network is only practical in certain situations. Following are some guidelines on when you should use another approach:

- Can you write down a flow chart or a formula that accurately describes the problem? If so, then stick with a traditional programming method.
- Is there a simple piece of hardware or software that already does what you want? If so, then the development time for a NN might not be worth it.
- Do you want the functionality to "evolve" in a direction that is not pre-defined? If so, then consider using a Genetic Algorithm (that's another topic!).
- Do you have an easy way to generate a significant number of input/output examples of the desired behavior? If not, then you won't be able to train your NN to do anything.
- Is the problem is very "discrete"? Correct answer can be found in a look-up table of reasonable size? A look-up table is much simpler and more accurate.
- Is precise numeric output values required? NN's are not good at giving precise numeric answers.

2. Character Recognition

The procedure of handwritten English character recognition is as follows:

• Acquire the sample by scanning.
• Skeletionization and Normalization operations are performed.
• Apply Boundary Detection Feature Extraction technique.
• Neural network Classification.\
• Recognized Character

2.1 English Character

The English language consists of 26 characters (5 vowels, 21 consonants) and is written from left to right. A set of hand written English characters as shown in figure 2

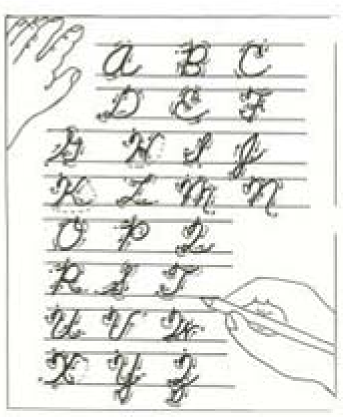

Figure 2: A Set of Handwritten English Characters

2.2 Scanning and Skeletonization

Handwritten characters are scanned and it has been converted into 1024 (32X32) binary pixels. The skeletonization process will be used to binary pixel image and the extra pixels which are not belonging to the backbone of the character has been deleted and the broad strokes has been reduced to thin lines .Skeletonization is illustrated in Figure 3.

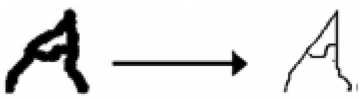

Figure 3: Skeletonization of an English Character

2.3 Normalization

There are lots of variations in handwritings of different persons. Therefore, after skeletonization process, normalization of characters is performed so that all characters could become in equal dimensions of matrix.

3. Character Recognition Types

3.1 OCR

OCR (Optical Character Recognition) also called Optical Character Reader is a system that provides a full alphanumeric recognition of printed or handwritten characters at electronic speed by simply scanning the form. More recently, the term Intelligent Character Recognition (ICR) has been used to describe the process of interpreting image data, in particular alphanumeric text.

Function of OCR Forms containing characters images can be scanned through scanner and then recognition engine of the OCR system interpret the images and turn images of handwritten or printed characters into ASCII data (machine-readable characters). Therefore, OCR allows us to quickly automate data capture from forms, eliminate keystrokes to reduce data entry costs and still maintain the high level of accuracy required in forms processing applications.

Features of OCR

The technology provides a complete form processing and +documents capture solution. Usually, OCR uses a modular architecture that is open, scalable and or flow controlled. It includes forms definition, scanning, image pre-processing, and recognition capabilities.

There are two basic types of core OCR algorithm, which may produce a ranked list of candidate characters. Matrix matching involves comparing an image to a stored glyph on a pixel-by-pixel basis; it is also known as "pattern matching" or "pattern recognition". This relies on the input glyph being correctly isolated from the rest of the image, and on the stored glyph being in a similar font and at the same scale. This technique works best with typewritten text and does not work well when new fonts are encountered. This is the technique the early physical photocell-based OCR implemented, rather directly.

Feature extraction decomposes glyphs into "features" like lines, closed loops, line direction, and line intersections. These are compared with an abstract vector-like representation of a character, which might reduce to one or more glyph prototypes. General techniques of feature detection in computer vision are applicable to this type of

OCR, which is commonly seen in "intelligent" handwriting recognition and indeed most modern OCR software. Nearest neighbor classifiers such as the k-nearest neighbors algorithm are used to compare image features with stored glyph features and choose the nearest match.

3.2 OMR

Optical mark recognition (also called optical mark reading and OMR) is the process of capturing human-marked data from document forms such as surveys and tests.

Many traditional OMR devices work with a dedicated scanner device that shines a beam of light onto the form paper. The contrasting reflectivity at predetermined positions on a page is then used to detect the marked areas because they reflect less light than the blank areas of the paper.

Some OMR devices use forms which are preprinted onto 'trans optic' paper and measure the amount of light which passes through the paper, thus a mark on either side of the paper will reduce the amount of light passing through the paper.

In contrast to the dedicated OMR device, desktop OMR software allows a user to create their own forms in a word processor and print them on a laser printer. The OMR software then works with a common desktop image scanner with a document feeder to process the forms once filled out.

OMR is generally distinguished from optical character recognition (OCR) by the fact that a complicated pattern recognition engine is not required. That is, the marks are constructed in such a way that there is little chance of not reading the marks correctly. This does require the image to have high contrast and an easily-recognizable or irrelevant shape. A related field to OMR and OCR is the recognition of barcodes such as the UPC bar code found on product packaging.

One of the most familiar applications of optical mark recognition is the use of 2 pencil (HB in Europe) bubble optical answer sheets in multiple choice question examinations. Students mark their answers, or other personal information, by darkening circles marked on a pre-printed sheet. Afterwards the sheet is automatically graded by a scanning machine. In the United States and most European countries, a horizontal or vertical 'tick' in a rectangular 'lozenge' is the most commonly used type of OMR form, the most familiar application being the UK National lottery form. Lozenge marks are a later technology and have the advantage of being easier to mark and easier to erase. The large 'bubble' marks are legacy technology from the very early OMR machines that were so insensitive a large mark was required for reliability. In most Asian countries, a special marker is used to fill in an optical answer sheet. Students likewise mark answers or other information via darkening circles marked on a pre-printed sheet. Then the sheet is automatically graded by a scanning machine.

3.3 ICR

Intelligent Character Recognition (ICR) is the module of OCR that has the ability to turn images of hand written or printed characters into ASCII data. Sometimes OCR is known as ICR. Most ICR software has a self-learning system referred to as a neural network, which automatically updates the recognition database for new handwriting patterns. It extends the usefulness of scanning devices for the purpose of document processing, from printed character recognition (a function of OCR) to hand-written matter recognition. Because this process is involved in recognizing hand writing, accuracy levels may, in some circumstances, not be very good but can achieve 97%+ accuracy rates in reading handwriting in structured forms. Often to achieve these high recognition rates several read engines are used within the software and each is given elective voting rights to determine the true reading of characters. In numeric fields, engines which are designed to read numbers take preference, while in alpha fields, engines designed to read hand written letters have higher elective rights. When used in conjunction with a bespoke interface hub, hand-written data can be automatically populated into a back office system avoiding laborious manual keying and can be more accurate than traditional human data entry. An important development of ICR was the invention of Automated Forms Processing in 1993. This involved a three stage process of capturing the image of the form to be processed by ICR and preparing it to enable the ICR engine to give best results, then capturing the information using the ICR engine and finally processing the results to automatically validate the output from the ICR engine.

- Optical character recognition (OCR) - targets typewritten text, one glyph or character at a time.
- Optical word recognition - targets typewritten text, one word at a time (for languages that use a space as a word divider). (Usually just called "OCR".)
- Intelligent character recognition (ICR) - also targets handwritten print script or cursive text one glyph or character at a time, usually involving machine learning. Artificial neural networks can be made indifferent to both affine and non-linear transformations. Intelligent word recognition (IWR) - also targets handwritten print script or cursive text, one word at a time. This is especially useful for languages where glyphs are not separated in cursive script.

3.4 The Different of ICR, OCR and OMR

- ICR and OCR are recognition engines used with imaging; while OMR is a data collection technology that does not require a recognition engine. Therefore, basically OMR can not recognize hand-printed or machine-printed characters. However, in the OCR
- OMR is a data collection technology that does not require a recognition engine OMR cannot recognize hand-printed or machine-printed characters.

3.5 Comparisons between OMR and OCR/ ICR

1. Optical Mark Reader (OMR)

a. Forms

An OMR works with specialized document and contains timing tracks along one edge of the form to indicate scanner where to read for marks. It also contains form ID marks, which look like black boxes on the top or bottom of a form. The cut of the form is very precise and the bubbles on a form must be located in the same location on every form.

b. Storage

With OMR, the image of a document is not scanned and stored.

c. Accuracy

Considering that OMR is simpler than OCR, and if the forms and the system is designed properly, then OMR has accuracy more than of OCR.

2. OCR/ ICR

a. Forms

OCR/ ICR are more flexible since no timing tracks or block like form IDs required. In addition, the image can float on a page. The ICR/ OCR technology uses registration mark on the four-corners of a document, in the recognition of an image. Respondents place one character per box on this form. The color is very important because the use of drop color reduces the size of the scanner's output and enhances the accuracy.

b. Storage/ retrieval

If document needs to be electronically stored and maintained, then OCR/ ICR is needed. Because with OCR/ ICR technologies, images can be scanned, indexed, and written to optical media.

3.6 Feature Extraction

This scans the binary image until it finds the boundary. The searching follows according to the clockwise direction. For any foreground pixel p, the set of all foreground pixels connected to it is called connected component containing p. The pixel p and its 8-neighbors are shown in Figure 3. Once a white pixel is detected, it checks another new white pixel and so on. The tracing follows the boundary automatically. When the first pixel is found, the program will be assigned the coordinates of that position to indicate that this is an origin of the boundary. The new found pixel will be assigned as a new reference point and starts the eight-neighbor searching. In this way, the coordinates of the initial point are varied according tothe position. As the tracer moves along the boundary of the image, the corresponding coordinates will be stored in an array for the computation of Fourier Descriptors. During the boundary tracing process, the program will always check the condition whether the first coordinates of the boundary are equal to the last coordinates. Once it is obtained; means the whole boundary has been traced and boundary tracing process completes.

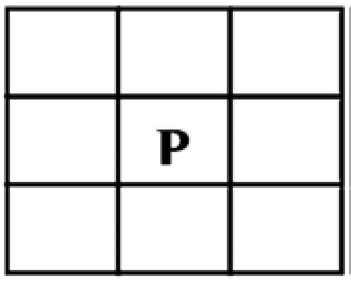

Figure 4: Pixel P and Its 8-neighbor

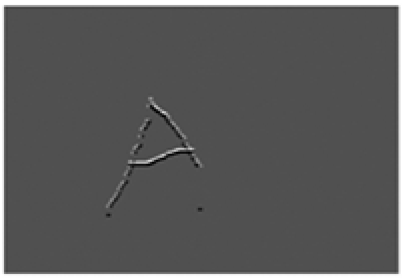

Figure 5: Boundary Tracing of a Character

4. Edit Distance Method

The **edit distance** between two strings of characters generally refers to the Levenshtein distance. "The term 'edit distance' is sometimes used to refer to the distance in which insertions and deletions have equal cost and replacements have twice the cost of an insertion". The edit distance is a common similarity measure between two strings. It is defined as the minimum number of insertions, deletions or substitutions of single terminal needed to transform other of the strings into the other one. The edit distance models were studied in two contexts, for string matching and for sequence similarity.

The edit distance is a common similarity measure between two strings. It is defined as the minimum number of insertions, deletions or substitutions of single terminal needed to transform other of the strings into the other one. This distance is a key importance in several fields, such as bioinformatics, Text processing consequently computational problems. Given a string S1 = [a 1 a 2 a 3... an] and S 2 = [b 1 b2 b3... b m] as the minimal cost of transforming S1 into S2 using the three operations insert, delete, and substitution, where only unit cost operations are considered in edit distance. The cost of elementary editing operations is given by some scoring function which induces a metrical on strings. The similarity of two strings is the minimum number of edit distance. DNA sequence can be seen as a pair of reverse complementary repeats in a string that are separated by a number of Nucleotide. The complementary relation on nucleotides (A T C G) means that A is complementary to T and C is complementary to G.

Given two strings of size m, n and set of operations replace (R), insert (I) and delete (D) all at equal cost. Find minimum number of edits (operations) required to convert one string into another.

Edit distances are measures of string similarity. A simple edit distance, which counts the minimal number of single-character insertions, deletions, and replacements needed to transform one string into another, could be returns 0 if *a* and *b* are identical, and 1 otherwise [Hall and Dowling, 1980].

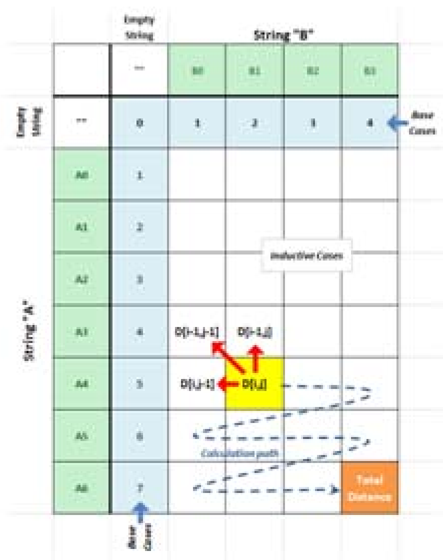

Figure 6: Distance Using Dynamic Programming:

Given two character strings $S_1$and S_2, the *edit distance* between them is the minimum number of *edit operations* required to transform S_1 into S_2 . Most commonly, the edit operations allowed for this purpose are: (i) insert a character into a string; (ii) delete a character from a string and (iii) replace a character of a string by another character; for these operations, edit distance is sometimes known as *Levenshtein distance* . For example, the edit distance between cat and dog is 3. In fact, the notion of edit distance can be generalized to allowing different weights for different kinds of edit operations, for instance a higher weight may be placed on replacing the character s by the character p, than on replacing it by the character a (the latter being closer to s on the keyboard). Setting weights in this way depending on the likelihood of letters substituting for each other is very effective in practice However; the remainder of our treatment here will focus on the case in which all edit operations have the same weight.

It is well-known how to compute the (weighted) edit distance between two strings in time $O(|S_1|*|S_2|)$ where Si denotes the length of a string $S_{i.}$ The idea is to use the dynamic programming algorithm, where the characters in $S_1$and S_2, are given in array form. The algorithm fills the (integer) entries in a matrix m whose two dimensions equal the lengths of the two strings whose edit distances is being computed; the (i,j) entry of the matrix will hold (after the algorithm is executed) the edit distance between the strings consisting of the first i characters of S_1 and the first j characters of S_2. The central dynamic programming where the three quantities whose minimum is taken correspond to substituting a character in S_1, inserting a character in $S_1$and inserting a character in S_2. Programming where the three quantities whose minimum is taken correspond to substituting a character in S_1, inserting a character in $S_1$and inserting a character in S_2 .

Dynamic Programming Algorithm for Edit distance method:

```
EDITDISTANCE($s_1, s_2$)
1  int $m[i, j] = 0$
2  for $i \leftarrow 1$ to $|s_1|$
3  do $m[i, 0] = i$
4  for $j \leftarrow 1$ to $|s_2|$
5  do $m[0, j] = j$
6  for $i \leftarrow 1$ to $|s_1|$
7  do for $j \leftarrow 1$ to $|s_2|$
8      do $m[i, j] = \min\{m[i-1, j-1] + \text{if } (s_1[i] = s_2[j]) \text{ then } 0 \text{ else } 1\text{fi},$
9                         $m[i-1, j] + 1,$
10                        $m[i, j-1] + 1\}$
11 return $m[|s_1|, |s_2|]$
```

5. Conclusion

Neural network based character recognition focuses on the font and different styles of hand writing during processing which are improved in ICR. Automatic updating, automatic checking of information against databases, cost reduction are its special features. Spatial distribution of the pixel values characterizes each character.. OCR, ICR and OMR are linked to each other with a thread of recognizing an input, to bring out a useful output. OMR apart from being speedy and accurate, it is an easy process with no typing errors, misconceptions involved. However, machine prints got ambitious, the need for hand-written text recognition got more serious and hence ICR was born. Edit distance method normalizes the value of length of the corresponding edit path.

References

[1] Aha, D. W., Kibler, D. & Albert, M. K. (1991) Instance based learning algorithms. Machine Learning 6(1), 37–66.

[2] Angluin, D. & Valiant, L. G. (1979) Fast probabilistic algorithms for Hamiltonian circuits and matchings. Journal of Computer and System Sciences 18, 155–193.

[3] Anthony, M. & Shawe-Taylor, J. (1993) A result of Vapnik with applications. Discret Applied Mathematics 47, 207–217.

[4] Verma B.K, "Handwritten Hindi Character Recognition Using Multilayer Perceptron and Radial Basis Function Neural Network", IEEE International Conference on Neural Network, 4, pp. 2111-2115, 1995.

[5] Sutha.J, Ramraj.N, "Neural Network Based Offline Tamil Handwritten Character Recognition System", IEEE International Conference on Computational Intelligence and Multimedia Application, 2007, 2, 13-15, Dec.2007, Page(s): 446-450, 2007.

[6] Yuelong Li Jinping Li Li Meng, "Character Recognition Based on Hierarchical RBF Neural Networks" Intelligent Systems Design and Applications, 2006. ISDA '06. Sixth International Conference, 1, On Page(s): 127-132, 2006.

[7] N.Kato, M.Suzuki, and S.Omachi, "A Handwritten Character Recognition System Using Directional Element Feature and Asymmetric Mahalanobis

Distance", IEEE Trans. on PatternAnalysis and Machine Intelligence, 21, No.3,

[8] pp. 258-262, 1999.

[9] C.C. Tappert, C.J. Suen and T. Wakahara, "The State of the Art in Outline Handwriting Recognition," IEEE Trans. on Pattern Analysis and Machine Intelligence, PAMI-12, No.8, pp.707-808, 1990.

[10] D.S. Yeung, "A Neural Network Recognition System for Handwritten Chinese Character Using Structure Approach," Proceeding of the World Congress on Computational Intelligence, 7, pp. 4353-4358, Orlando, USA, June 1994.

[11] D.Y. Lee, "Handwritten Digit Recognition Using K Nearest-neighbor, Radial Basis Function and

[12] Backpropagation Neural Networks,"IEEE Neural Computation, 3, Page(s) 440- 449.

[13] Y.Y. Chung, and M.T. Wong, "Handwritten Character Recognition by Fourier Descriptors and Neural Network", Proceedings of IEEE TENCON – Speech and Image Technologies for C.

[14] H. Almualim and S. Yamaguchi, "A Method for

[15] Recognition of Arabic Cursive Handwriting," IEEE Transon Pattern and Machine Intelligence, 9, No 5, pp.715-722, Sept. 1987.

[16] I.S.I. Abuhaiba and S.A. Mahmoud, Recognition of Handwritten Cursive Arabic Characters," IEEE Transaction on PA&MI, 16, No 6, pp. 664-672, June 1994