1-1-2004

Handwritten character recognition using a gradient based feature extraction

Ravi Kiran Reddy Veligati
University of Nevada, Las Vegas

HANDWRITTEN CHARACTER RECOGNITION USING A GRADIENT BASED FEATURE EXTRACTION

by

Ravi Kiran Reddy Veligati

Bachelor of Engineering in Computer Science
Osmania University, India
2002

A thesis submitted in partial fulfillment
of the requirements for the

Master of Science Degree in Computer Science
Department of Electrical and Computer Science
Howard R. Hughes College of Engineering

Graduate College
University of Nevada, Las Vegas
December 2004

UNLV

Thesis Approval

The Graduate College
University of Nevada, Las Vegas

NOVEMBER 22nd , 2004

The Thesis prepared by

RAVI KIRAN REDDY VELIGATI

Entitled

HANDWRITTEN CHARACTER RECOGNITION USING A GRADIENT BASED FEATURE EXTRACTION.

is approved in partial fulfillment of the requirements for the degree of

M.S.C.S, MASTER OF SCIENCE IN COMPUTER SCIENCE

Examination Committee Chair

Dean of the Graduate College

Examination Committee Member

Examination Committee Member

Graduate College Faculty Representative

ABSTRACT

Handwritten Character Recognition using a gradient based feature extraction

by

Ravi Kiran Reddy Veligati

Dr. Evangelos Yfantis, Examination Committee chair
Professor of Computer Science
University of Nevada, Las Vegas

Handwriting Recognition is the task of transforming a language that is represented in its spatial form of graphical marks into its symbolic representation [1]. In Offline Handwriting Recognition, there are three steps: preprocessing of the image, segmentation of words into characters and recognition of the characters. In this thesis I implemented two methods for character recognition, which is the most important step in Offline Handwriting Recognition. The heart of character recognition is the features that are extracted from the character image. The accuracy of the classification of the character image depends on the quality of the features extracted from the image. The two methods presented in this thesis use two different types of features. One uses the connectivity features among various segments in a character image, and the other method uses the gradient feature at each pixel to construct the feature vectors. Both these methods are discussed in detail in the following chapters.

TABLE OF CONTENTS

ABSTRACT iii

LIST OF TABLES v

ACKNOWLEDGMENTS vi

CHAPTER 1 INTRODUCTION 1
1.1 Overview of Handwriting Recognition 1
1.2 Problem Statement 4

CHAPTER 2 LITERATURE REVIEW 5
2.1 Past Research Related to Handwriting Recognition 5

CHAPTER 3 CHARACTER RECOGNITION 14
3.1 Data Aquisition 14
3.2 Connectivity Features 16
3.2 Gradient Features 19
3.8 Classification 26

CHAPTER 4 EXPERIMENTS AND RESULTS 29
4.1 Results 29
4.2 Analysis of Results 30
4.2 Word Recognition 33

CHAPTER 5 CONCLUSION 34
5.1 Future Research 34

BIBLIOGRAPHY 35

VITA 38

LIST OF FIGURES

Figure 1 Steps involved in handwriting recognition 2
Figure 2 Middle line features of character 'v' (vertical direction) 7
Figure 3 Concave Features 8
Figure 4 Typical chain of stages for handwriting recognition 11
Figure 5 Input word image 13
Figure 6 Output character image 14
Figure 7 Fused character image 15
Figure 8 Character 'A' 15
Figure 9 Character 'A' after segmentation 15
Figure 10 Character segmentation 16
Figure 11 Sobel operator for horizontal component 19
Figure 12 Sobel operator for vertical component 19
Figure 13 The Sobel operators 19
Figure 14 Gradient magnitude 20
Figure 15 Gradient direction 20
Figure 16 (a) Original Image (b) Gradient magnitude (c) Gradient Directions 21
Figure 17 Partitioning of gradient map into 4 x4 grid 22
Figure 18 Quantization of gradient directions 23
Figure 19 Eucledean distance 26
Figure 20 Sample data 28
Figure 21 Experimental results 29
Figure 22 Ambigious handwritten characters 31

ACKNOWLEDGMENTS

First of all I would like to express my gratitude to Professor Evangelos A. Yfantis for supervising me in my thesis and providing the invaluable support and guidance without which this thesis would not have been completed.

I would like to thank Dr. Ajoy K. Datta, Dr. John T. Minor and Dr. Venkatesan Muthukumar for their direct and indirect contributions throughout this endeavor.

I would like to acknowledge the assistance provided by my associates in this project. Also, I would like to thank my coworkers Dimitri Papaioannou, Edwin So, Jason Hurt, Jae Adams, Magesh Kumar Padmanabhan and Meenakshi Sundar in this project.

Finally I would like to thank all my friends who were there to help and support me during the development of this thesis work.

CHAPTER 1

INTRODUCTION

1.1 Overview of Handwriting Recognition

Handwriting is a skill that is personal to individuals. Fundamental characteristics of handwriting are threefold. It consists of artificial graphical marks on a surface; its purpose is to communicate something; this purpose is achieved by virtue of the mark's conventional relation to language [9].

Handwriting recognition is the task of transforming a language represented in its spatial form of graphical marks into its symbolic representation. Handwriting interpretation is the task of determining the meaning of a body of handwriting, e.g., a handwritten address. Handwriting recognition and interpretation are processes whose objectives are to filter out the variations in the graphical representation, and determine the intended message.

The major steps in handwriting recognition are Preprocessing, Feature Extraction and Classification.

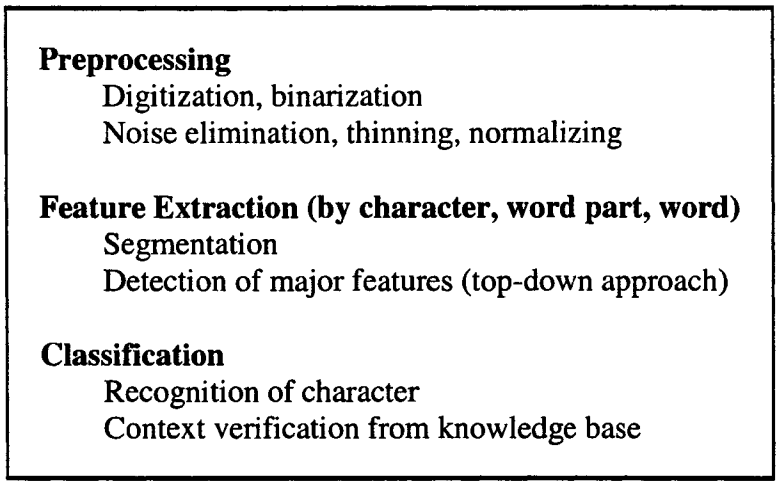


Figure 1: Steps involved in handwriting recognition

The initial stage of the process, the preprocessing steps, includes general image processing algorithms and also more application-specific algorithms such as straightening of slanted cursive writing, *geometric transformation* for straightening a skewed image, *thinning* the image to obtain a boundary line representation for that image, *normalizing* an image etc are some of the preprocessing steps required for various character recognition algorithms.

Segmentation is the process of dividing an image into regions, each susceptible to containing a single object or a group of objects of the same type. For instance, an object can be a character on a text page or a line segment in a drawing. A group of objects can represent a word or two line segments that touch each other [6].

Feature Extraction is the process of extracting a set of parameters that define the shape of the underlying character, as precisely and uniquely as possible. Another important feature of a parameterization method, which will provide the highest degree of

noise immunity and a good generalization capability for the resulting system, is the continuity of the representation. This means that similar objects must be mapped into similar representations. In summary, the three features that identify a good parameterization method are: precision, uniqueness and continuity [8].

Classification in character recognition is the process of analyzing the input feature vectors of the character image and classifying it to its corresponding character. There are several classification techniques like K-Nearest Neighbor, Neural Networks, Decision Tree classifiers etc.

On-line and Off-line Handwriting recognition

On-line recognition in general means that the recognition takes place as the user is writing. In the handwriting recognition jargon, *on-line* also refers to a special type of input data containing information about the pen trajectory as given, for instance, by a digitizing tablet. A typical application of on-line handwriting recognition is pen computer interfaces.

In contrast, Off-line recognition takes place sometime long after handwriting. The raw input data is most often a digitized image or 'pixel map', obtained, for instance, with a scanner. This is why "off-line" character recognition and Optical Character Recognition (OCR) are usually synonymous. Typical OCR applications include post office address readers and bank check readers. The recognizers must often process hundreds of characters per seconds [11].

1.2 Problem Statement

Character Recognition is one of the significant parts of a complete Handwriting recognition System. There are other problems like preprocessing the image document, segmentation into lines, segmentation of lines into words and words into characters. The characters are then recognized. Then once the characters are recognized, grouping them into words is done where contextual knowledge helps to improve the accuracy of the word recognition.

This thesis concentrates on the important part of character recognition. The goal is to develop a character recognition system which takes as input a character image and classifies it into one of the characters in the alphabet. To develop this system, preprocessing, feature extraction and classification algorithms are needed.

CHAPTER 2

LITERATURE REVIEW

2.1 Past Research in Character Recognition

Optical character recognition is a well studied field and a lot of research has been done on it. For each stage of the Character recognition system, various algorithms and approaches have been proposed and implemented.

Algorithms from Image processing, Statistics and Machine learning have been applied to handwriting recognition at each stage of the recognition system. In a study made by M. Sridhar and F. Kimura, "Segmentation based Cursive Handwriting Recognition", they developed a character recognition system where the features are based on chain code histograms.

Feature Extraction

Feature Extraction is one of the most important phases in a character recognition system. There are several types of features that are extracted.

Chain code feature

Local chain code histograms are used as a feature vector. The rectangular frame enclosing a character is first divided into 7 x 7 blocks. In each block, a chain code histogram of the character contour is calculated. The feature vector is composed of the local histograms. Since contour orientation is quantized to one of four possible values (0^{o}

or '-', 45 ° or '/', 90 ° or '|' or 135 ° or '\'), a histogram in each block has four components. After the histogram calculation, the 7 x 7 blocks are down sampled with a Gaussian filter into 4 x 4 blocks. Thus the feature vector has 64 elements when all the 16 blocks are included [14].

String distance measurement

In a study made by Sameer Singh, "Shape Detection Using Gradient Features for Handwritten Character Recognition", a feature extraction method known as String Distance Measurement is introduced which is based on the measurement of gradient change.

SDM quantifies the property of gradient change as the shape changes. If the image is in grey scale, it is converted into a binary image by assigning every pixel value equal to or greater than 128 a value 1 and all others a value 0. The source image is skeletonized and divided into nine segments containing equal number of pixels. For each segment, its *SDM* feature is calculated [15].

String distance (SD) represents the change in gradient as we traverse in a downward direction, which is summed up by considering any two adjacent rows at any one time. SD's are calculated for each of the nine segments.

Zhang ping and Chen Lihui, "A novel feature extraction method and hybrid tree classification", describe several global features of a numeral character which are described briefly in the following paragraphs.

Middle Line features

Middle line consists of a set of middle points between two neighboring strokes, in which the middle line can be established from horizontal direction or vertical direction, respectively.

The algorithm for extracting middle line features is very simple. The middle point of the adjacent two strokes needs to be written down by scanning the character image from left to right along the vertical direction. For example, the middle line of the character '*v*' is illustrated in the following figure. The symbol '#' represents extracted middle points which form the middle line [16].

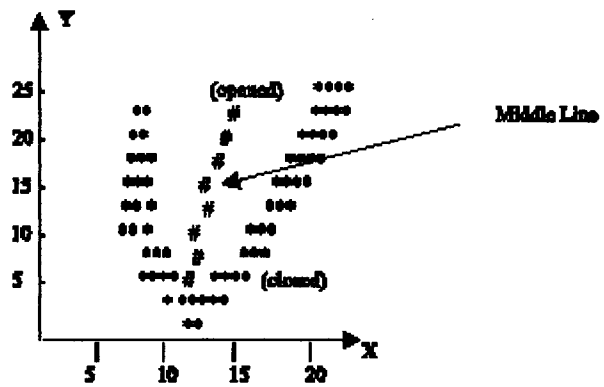


Figure 2: Middle line features of character 'v' (vertical direction)

Concave Features

Concave feature describes the concavity in characters outer profiles from the top/bottom/left/right direction point of view. For example, left concave feature is shown in Figure 2.

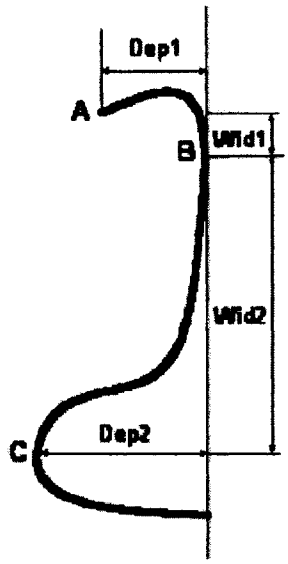


Figure 3: Concave features

From the Character's left profile A-B-C, A and C are two most outer edge points, the B is a most interior point. The middle point of between point A and point C is assigned as the concave point. Some parameters are defined as

Dep1 = | A – B | , Dep2 = | C- B|,

Depth: D = min(Dep1, Dep2),

Width: W = Wid1 + Wid2,

Concavity: C = D/W

We can take a threshold, for example C > 0.3, to take the concave feature into consideration; otherwise it is invalid and hence ignored. The number and position of each concave feature in each profile can be featured [15].

Width Feature

A normalized character is divided into four equal sub-regions vertically (along the direction of the character height). The maximum width of each sub-region is calculated

$m1$, m 2, m 3, m 4, respectively. In order to systematically quantify the width of any sub-region $x \in (m1, m2, m3, m4)$, a scaling function $f(x)$ is calculated and used as width feature.

$$f(x) = \text{int}(\alpha^*(x - \text{Mmin})/(\text{Mmax} - \text{Mmin})),$$

where Mmax = max{mi}, Mmin = min {mi}, i = 1,..,4; ά is a scale factor and selected as 3 which ensures the width feature of each sub-region can be encoded by 2 bits[15].

Point Features

Several other features like end points, branch points and cross points can be defined [17]. These features are easily extractable and can be encoded as point features.

Classifiers

Classification is the process of analyzing the input features and assigning the input sample, a class based on those input features.

Neural Network Classifiers

Neural networks classifiers (NN) have been used extensively in character recognition [18,19]. These networks can be used as a combined feature extractor and classifier, where the inputs are scaled or sub–sampled input image, or as a "pure" classifier where the inputs are extracted features. One problem of using neural networks in character recognition is that it is difficult to analyze and fully understand the decision–making process.

Gader et al. [20] describe an algorithm for hand printed word recognition that uses two 27–output–4–layer backpropagation networks, one for uppercase and the other for lowercase characters, to account for the cavity features and other two 27–output–4–layer backpropagation networks to account for the direction–value features. Cavities are

computed using mathematical morphology and yields up to a 105–dimensional vector. Direction–value features are derived from zones using boundary and skeletal pixels, and yield up to a 60–dimensional feature vector. Further, the outputs of the networks are combined to optimize recognition accuracy. Recognition rates of 94% and 82% were achieved for uppercase and lowercase characters respectively.

Hidden Markov Model

An HMM is a doubly stochastic process with an underlying Markov process that is not observable, but can only be observed through another set of stochastic processes which are produced by the Markov process. Let $O = (o_1 \ldots\ldots o_T)$ be the sequence of observations produced by a Markov state sequence $Q = \{q_1 \ldots\ldots q_t\}$, where each observation o_t is from the set of M observation symbols $V = \{v_k; 1 \leq k \leq M\}$ and each state q_t is from the set of N states $S=\{s_i; 1 \leq i \leq N\}$.

Thus HMM can be characterized by:

$\prod = \{\pi_i\}$, where $\pi_i = P(q_1 = s_i)$ is the initial state probability;

$A=\{a_{ij}\}$, where $a_{ij} = P(q_{t+1} = s_j \mid q_t = s_i)$ is the state transition probability;

$\Gamma = \{\gamma_j\}$, where $\gamma_j = P(q_T = s_j)$ is the last state probability;

$B = \{b_j(k)\}$, where $b_j(k)=P(o_t=v_k|q_t=j)$ is the symbol probability;

and they satisfy the probability constraints.

The HMM study consists of three basic problems: scoring problem, training problem and recognition problem. The recognition problem is finding the optimal state sequence of an HMM. A Viterbi algorithm [21], which is a dynamic programming procedure, can solve the recognition problem.

K nearest neighbor Classifier

Nearest Neighbor classifiers are based on learning by analogy. The training samples are described by *n*-dimensional numeric attributes. Each sample represents a point in the *n*-dimensional space. In this way all of the training samples are stored in an *n*-dimensional pattern space. When given an unknown sample, a *k-nearest neighbor classifier* searches the pattern space for the *k* training samples that are closest to the unknown sample. These *k* training samples are the *k* "nearest neighbors" of the unknown sample [13].

Didier Guillevic and Ching Y. Suen used a modified K-Nearest Neighbor algorithm to implement a method for recognizing unconstrained words belonging to a small static lexicon [23].

General Handwriting Recognition system

Finally a general handwriting Recognition system is shown below in Figure 4.

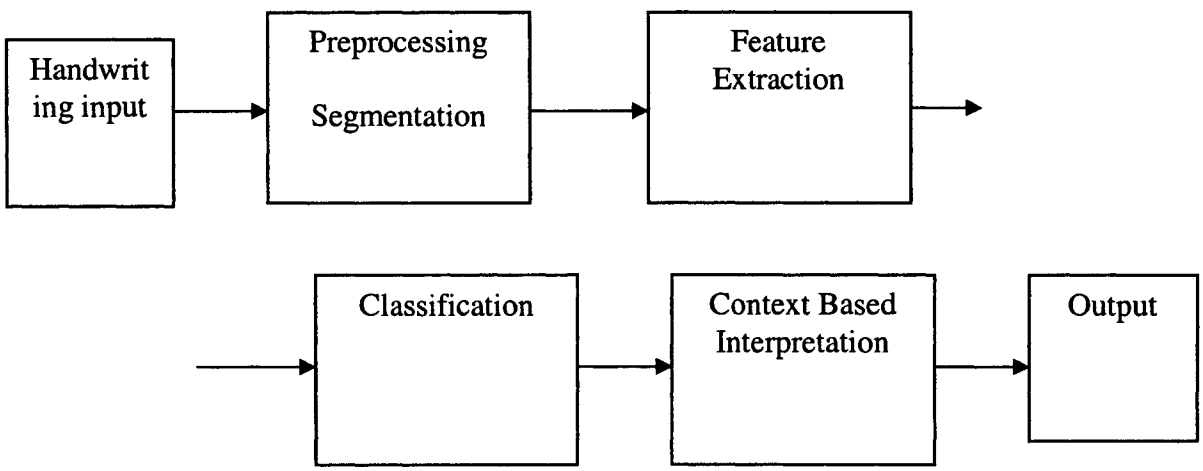


Figure 4: Typical chain of stages for handwriting recognition

In most applications the classification of patterns is only part of the overall recognition system, embedded in a sophisticated network of operations such as Preprocessing, Segmentation, Feature Extraction and Knowledge based processing [24].

As an example, consider the input to be a postal mail address to be recognized. The first steps are scanning and preprocessing of the address image, resulting in a two-value raster image with dark text and white background pixels. Segmentation is based on a geometric analysis in order to identify the address block as an arrangement of lines, and then to divide these lines into single classifiable objects, characters and digits.

Feature Extraction is highly important for every classification problem. A good choice of features is often half the solution. In feature extraction the domain dependent knowledge of the human system designer comes into play, since the task is to describe the relevant properties of the objects to be recognized as precisely as possible with a fixed number, as small as possible, of feature variables[24].

There exists close connection between feature extraction and classification, since the efforts going into feature extraction and classification can be exchanged to a certain degree.

After classification, the possibly ambiguous estimations of the Classifier are processed further, making use of contextual constraints, such as dictionaries, statistics and application dependent knowledge.

CHAPTER 3

CHARACTER RECOGNITION

3.1 Data Acquisition

The data for the implementation of this thesis was obtained from "IAM Database – A Database for Off-line Handwriting Recognition Research" [4]. Unfortunately this database did not have a collection of character images. This database consisted of a form database, sentence database and word database.

A simple algorithm to extract character images from the word database has been implemented. This algorithm takes as input a word image and then it segments the word image into several segmented components. Once the components are segmented, the first segmented component is stored, which is a character image.

Figure 5: Input word image

Figure 6: Output character image

A word database of 13817 words has been processed by the above algorithm and a character database of 1408 capital letters has been obtained.

Algorithm1

Input: A set of word images.

Step 1: For each word image check if the first letter it starts with is a capital letter using the word reference.

Step 2: If the given word starts with a capital letter then get all the connected components in the word image, otherwise ignore the image.

Step 3: Store the first connected component in a separate directory, starting with the first letter of the word from which it is extracted.

Output: A set of character images.

Some of the character images in the output may be two or more characters that are fused together. This happens if the first letter in the word image is connected to the second letter and so on.

Figure 7: Fused character image

In this case the word image "After" was written such that the first two letters A and F are connected. Hence when the algorithm is applied, the result looks like the above figure. Such character images should be discarded.

3.2 Connectivity Features

The Connectivity Features method for character recognition uses the segmentation strategy to recognize the character. The main strategy in this is to segment each character into a 3*3 matrix format that has a total of nine segments. Then the distribution of pixels in these segments is used to find out the features and distinguish between each character [10].

This strategy is used because capital letters in English have distinct distributions, which is automatically followed even when it is handwritten. Each character is segmented as shown in the figures.

Figure 8: Character 'A'

Figure 9: Character 'A' after segmentation

The first image is the actual handwritten character, 'A' which is segmented into nine parts as shown in the second part of the figure. They are segmented into 9 equal parts by dividing them using 1/3rd of their length.

These segments are then used to recognize the character, based on the distribution of the pixels. The top three segments are 1, 2, and 3 from left to right, similarly 4, 5, 6 and 7, 8, 9.

1	2	3
4	5	6
7	8	9

Figure 10: Character segmentation

The basic methodology that the heuristic method follows is based on the connections between the above mentioned segments in each alphabet. The heuristic method checks for connections between the segments first by each column, then by each row. A 1 is assigned if there is a connection between the segments, a 0 signifies that the segments are not connected. Each character has some distinct feature that makes this heuristic method efficient. For example: see the character 'A' in the figure. In general, the segments 4-5 and 5-6 are connected while checking the alphabet horizontally. Vertically, 4-7 and 6-9 are connected. This is valid for all different ways of writing A. This set of connections will be present. These set of connections are unique for A, and are written into a binary format. In the same way, all the letters can be recognized based on

their individual connection codes. This method of recognition can be bolstered further by running some heuristic methods which check for those segments that are empty in each character. Continuing with the same example of A, the segments that are most likely to be empty are 1, 3 and 8. These zero pixel segment combinations are also unique for each letter [10].

Character A

As explained in the above mentioned example, the character A is recognized based on its connections between the segments, and also by the combinations of its zero pixel segments. The connections in A as mentioned above are between the segments 4-5, 5-6, 4-7 and 6-9. These connections are assigned as 1. In the same way, 5-8 is a segment connection that is never connected. The unconnected segment pairs are assigned 0s. This sequence of 1s and 0s generates a binary format which is unique for this particular character. For example, consider four different types of writing A. The heuristic method generates the following binary sequences for each of the four cases.

A = 010101010010

A = 101101010010

A= 011101010010

A = 110101010010

From the above sequences, it can be observed that the sequence from the 5th digit onwards is always the same, for any kind of A. This characteristic sequence can be used to recognize the letter [10].

Algorithm 2

Input: The character image

Step1: Preprocess the input image. Binarize the input character image if it is in grey scale.

Step 2: Perform the thinning operation on the binarized image.

Step 3: Segment the image into a 3x3 matrix.

Step 4: Find Connectivity between segments adjacent to each other in horizontal and vertical directions. Connectivity between segments is represented by a '1' and no connectivity is represented by a '0'.

Output: A binary feature vector of length 12 for the input character.

3.3 Gradient Features

Since the accuracy of the Character Recognition Classifier depends on the quality of the character features provided to it, we need to have a robust and accurate feature extraction algorithm.

The *Gradient* computes the magnitude and direction of the greatest change in intensity in a small neighborhood of each pixel [5]. Gradients are computed by means of the *Sobel operator* [3]. The sobel templates used to compute the horizontal (X) and vertical (Y) components of the gradient are shown in the following figures.

-1	0	1
-2	0	2
-1	0	1

X

Figure 11: Sobel operator for horizontal component

1	2	1
0	0	0
-1	-2	-1

Y

Figure 12: Sobel operator for vertical component

Given an input image I() of size D_1 x D_2, each pixel neighborhood is convolved with these templates to determine the X and Y components S_x and S_y, respectively. That is,

$$S_x(i,j) = I(i-1,j+1) + 2I(i,j+1) + I(i+1,j+1) - I(i-1,j-1) - 2I(i,j-1) - I(i+1,j-1),$$

$$S_y(i,j) = I(i-1,j-1) + 2I(i-1,j) + I(i-1,j+1) - I(i+1,j-1) - 2I(i+1,j) - I(i+1,j+1),$$

Figure 13: The Sobel operators

Here i, j range over the image rows (D1) and columns (D2), respectively. The gradient magnitude is then calculated as

$$r(i,j) = \sqrt{S_x^2(i,j) + S_y^2(i,j)}.$$

Figure 14: Gradient magnitude

The Gradient Direction is calculated as

$$\theta(i,j) = \tan^{-1} \frac{S_y(i,j)}{S_x(i,j)}.$$

Figure 15: Gradient direction

An example of the gradient magnitude and direction is depicted in the following figure.

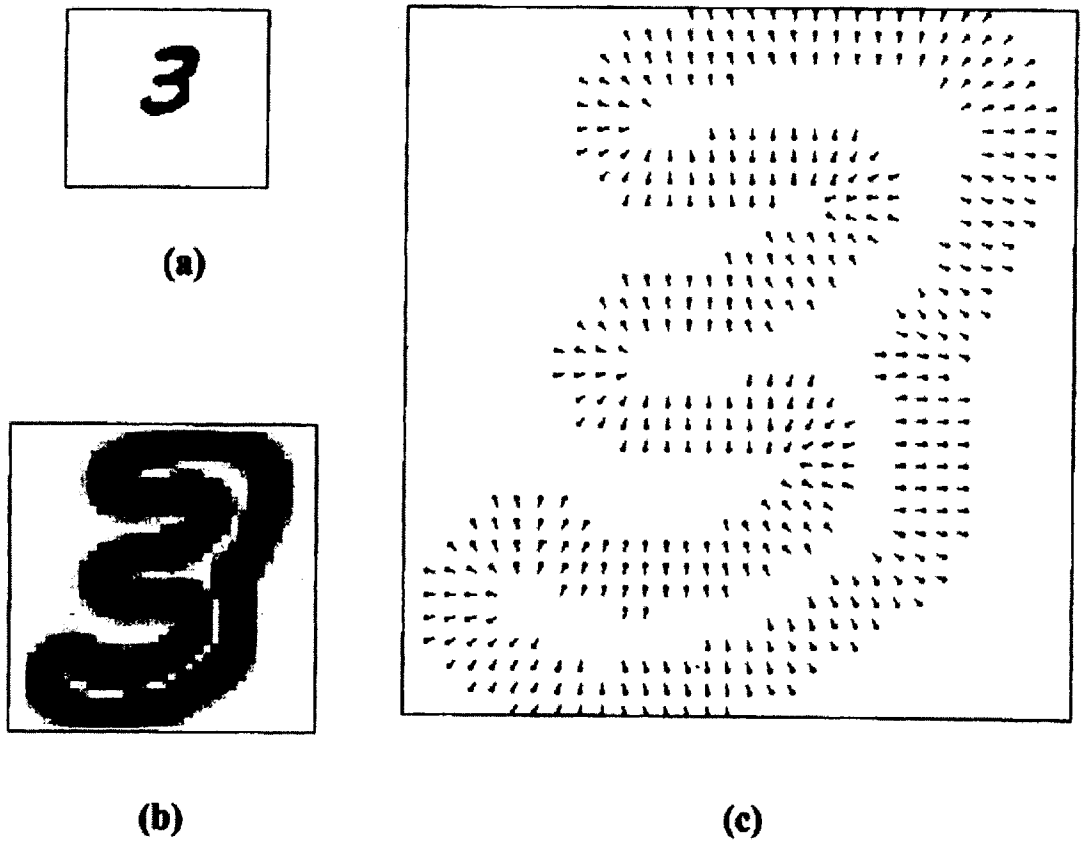


Figure 16: (a) Original Image (b) Gradient magnitude (c) Gradient Directions

The white space in Fig. 16(b) indicates that the gradient magnitude at that pixel is zero. The arrows indicate gradient direction at pixels with positive gradient magnitude in Figure 16(c) [5]. We notice from Figure 16(c) that the Gradient direction varies along with the contour of the character image.

Feature Computation

The features are extracted from the gradient map of the character image. There are two ideas that are incorporated in these feature sets:

a) Partitioning of the Gradient map into regions and

b) Quantization of Gradient directions.

The gradient map is partitioned to find the local contour variations. That is, each image, of size $D_1 \times D_2$ is divided into $N_1 \times N_2$ parts, by subdividing along the height and width of the image. Each partition has dimension $(D_1/N_1) \times (D_2/N_2)$ approximately. In Figure 17 the gradient map of character '0' is partitioned into a 4 x 4 grid.

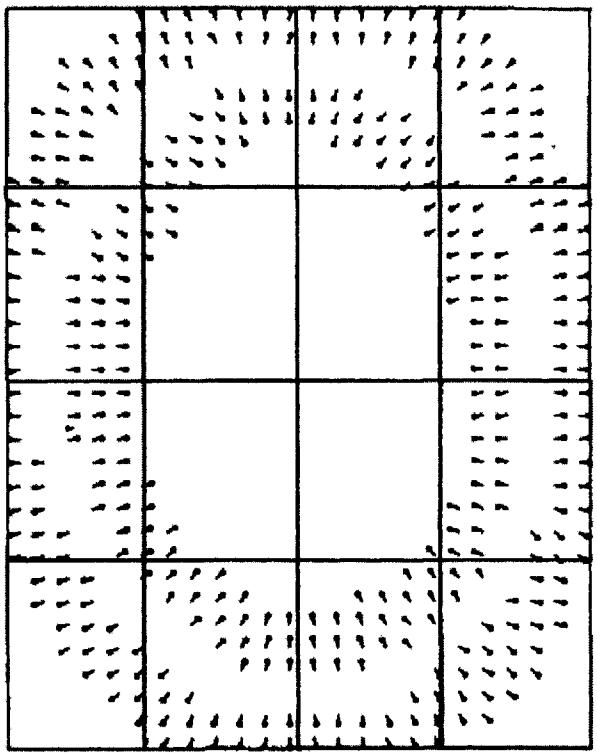


Figure 17: Partitioning of gradient map into 4 x4 grid

Quantizing gradient directions into a small number, **K**, of ranges aids in the generation of a fixed number of features. In this implementation, the gradient directions are quantized into 12 ranges, 0°-30°, 30°-60°, 60°-90°..... 330°-360°, respectively. An

An Example of quantizing the actual gradient directions at a pixel into 12 bins is

shown in Figure 18. After such a quantization, gradient directions are in integers in the range [1, 12].

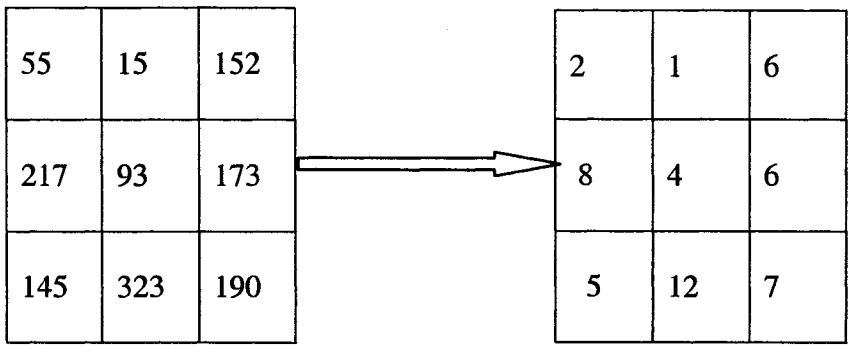


Figure18: Quantization of gradient directions: map [0, 30] to 1, [30, 60] to 2 and so on until [330, 360] to 12.

Partitioning the gradient map and gradient direction quantization results in a normalized feature space. Partitioning of gradient directions in each region into **K** bins is done even when it appears that none of the characters will have a particular range (a, b) in a certain region.

A feature vector F_i is computed in each partition, where $1 <= i <= N_1 \times N_2$. Feature vectors from all partitions are concatenated to form the complete feature vector, F.

The features indicate the presence or absence of pixels in a gradient range, in each partition. Gradient directions of all pixels are quantized to the K ranges. Exactly K features are determined in each partition corresponding to the K direction bins. Hence a total of $N_1 \times N_2 \times K$ features are computed over the whole character image.

In each partition, increment the k^{th} bin whenever a pixel with gradient direction in the k^{th} range is found, where $1<=k<=K$. In the feature vector F_i, element $F_i(k)$ is *ON* if the k^{th} bin in the i^{th} partition has a non-zero entry, i.e. there exists a pixel in the i^{th} partition with direction in the k^{th} range.

Algorithm 3

Input: The character image of size $D_1 \times D_2$.

Step 1: Preprocess the input image. Binarize the input character image if it is in grey scale. Initialize the feature vector.

Step 2: At each pixel in the image find the gradient magnitude and direction using the sobel operators. Quantize the gradient direction into K bins. Enclose the Binary image and divide into a $N_1 \times N_2$ grid.

Step 3: Initialize the Feature Vector F to 0. The length of the feature vector is $N_1 \times N_2 \times K$.

Step 3: In each partition, increment the k^{th} bin if a pixel has gradient direction in the k^{th} range. $(1<=k<=K)$.

Step 4: Set the value of $F_i(k)$ in the Feature vector to 1 if the k^{th} bin in i^{th} partition is non zero.

Step 5: Concatenate feature vectors from all the partitions to form the final feature vector F.

Output: Feature Vector for the input character image of size $N_1 \times N_2 \times K$.

Computation Complexity

Gradient computation requires convolution of the original image of size $D_1 \times D_2$ with the sobel X and Y operators. As this involves multiplication by integers (1, 2, -2, -1) and addition, few arithmetic operations are needed at each pixel. A cumulative sum of gradient magnitudes and direction quantization can be maintained along with gradient computation at each pixel. All these operations can be implemented in linear time, that is $O(N) = O(D_1D_2)$. Here $N = D_1D_2$ defines the length of the input data.

Extraction of the binary features requires an additional O(N) time, i.e. one pass through the gradient map.

Hence it can be seen that total feature extraction time is linear in the input size and involves a few simple arithmetic operations at each pixel.

3.4 Classification

The method of nearest neighbor, proposed by T. M. Cover and P. E. Hart [2], has become a very efficient nonparametric approach to classification. Partly because of its perfect mathematical theory, the NN method develops into several variations [12].

Nearest Neighbor classifiers are based on learning by analogy. The training samples are described by *n*-dimensional numeric attributes. Each sample represents a point in the *n*-dimensional space. In this way all of the training samples are stored in an *n*-dimensional pattern space. When given an unknown sample, a *k-nearest neighbor classifier* searches the pattern space for the *k* training samples that are closest to the unknown sample. These *k* training samples are the *k* "nearest neighbors" of the unknown sample [13].

Closeness is defined in terms of Euclidean distance, where Euclidean distance between two points, $X = (x_1, x_2, \ldots, x_n)$ and $Y = (y_1, y_2, \ldots, y_n)$ is

$$D(X,Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Figure 19: Euclidean distance

The unknown sample is assigned to the most common class among the k nearest neighbors. When $k = 1$, the unknown sample is assigned the class of the training sample that is closest to it in pattern space.

Nearest neighbor classifiers are *instance-based* or *lazy learners* in that they store all of the training samples and do not build a classifier until a new (unlabeled) sample needs to be classified. This contrasts with *eager learning* methods, such as decision tree induction and back propagation, which construct a generalization model before receiving new samples to classify.

As expected, lazy learning methods are faster at training than eager methods, but slower at classification since all computation is delayed to that time. Unlike decision tree induction and back propagation, nearest neighbor classifiers assign equal weight to each attribute. This may cause confusion when there are many irrelevant attributes in the data.

Both the feature vectors obtained using the Connectivity features and the gradient features algorithms are binary feature vectors so there is no confusion regarding weights of the attributes in the feature vector.

Character Recognition

By combining all the algorithms that have been discussed earlier we can arrive at a character recognition algorithm as follows.

Algorithm 4

Input: The character image which needs to be classified.

Step 1: Preprocess the character image by converting it into a binary image if it is not in binary format.

Step 2: Extract the features using either *algorithm 1* (connectivity features) or *algorithm 2* (gradient features).

Step 3: Use the K nearest neighbor algorithm to classify the character image based on the features extracted from the character.

Output: The class of the input character.

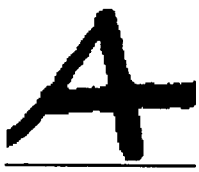
CHAPTER 4

EXPERIMENTS AND RESULTS

4.1 Results

The algorithms that were described in the previous chapter were implemented using Java on Eclipse platform. Both the algorithms were tested using both Handwritten and Machine printed characters.

Machine printed



Handwritten

Figure 20: Sample data

For machine printed characters the accuracy was 96% for a test sample of size 100 including letters from A to J.

For Handwritten characters an accuracy of 82.20% was achieved for a character database of size 834 capital letter characters.

An Experiment has been done by mixing Handwritten Characters and Machine printed characters in both the training and test data sets over a sample of 1034 characters and achieved an accuracy of 84.62%.

EXPERIMENTAL RESULTS

Train Data	Test Data	Connectivity Algorithm	Gradient Algorithm	Comments
407	427	59.25%	82.20%	Handwritten characters, A to Z
507	527	56.73%	84.62%	Mixed Characters, A to Z.
144	118	69.16%	90.00%	Handwritten characters, A, B, C, D, E
100	100	74.00%	96.00%	Printed characters different font sizes 12,14

Figure 21: Experimental results

4.2 Analysis of Results

The contrast in the results can be explained by observing the contrast between both the algorithms that are used for the character recognition.

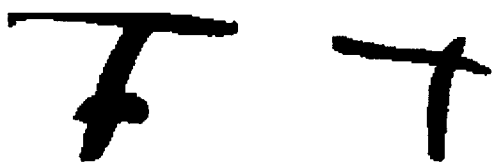
We can observe from the above results that the gradient features outperform the connectivity features. This can be explained as follows.

The results were obtained from a very natural handwritten character dataset which can be very ambiguous and contain some inconsistency in their general representation. Some of the examples of the dataset used in the recognition are

i) U ii) V

i) H ii) M

i) O ii)G



i) F ii) T

i) E ii) C

Figure 22: Ambiguous handwritten characters

The above sample characters explain the difficulty in recognizing the handwritten characters. The algorithm should capture the uniqueness of the character to the maximum extent possible in order to differentiate between other similar representations.

In a better situation where the characters are represented reasonably well, the character recognition goes well beyond 90%. This can be explained by the accuracy achieved on a Machine printed character set (96%) where each character is clearly and consistently very different from all other characters.

But in practice Handwriting always has some ambiguities and inconsistencies compared to the actual symbolic representation of the characters.

The gradient features method is motivated by the need for structural analysis, particularly in localized contour analysis. The Gradient representation is based on gradient magnitude and direction and it is rich in information. In addition the computation is based on a few simple operations at each pixel.

The algorithm is also very robust as it can be applied to both alphabet character as well as digits, handwritten characters as well as machine printed characters. Since the features are extracted after enclosing the character into a grid we do not have to normalize the character to a particular size.

4.3 Word Recognition

The accuracy of the word recognition can be improved if we have a good character recognition system. We understand that in Handwriting recognition the text image is segmented into words, and the words are segmented into characters. Once the characters are recognized, they are grouped into words and the words are grouped into sentences.

Once the characters are recognized and when they are being grouped into words, we can use the contextual knowledge and probability to improve the accuracy of the word recognition.

For example if we have a word 'RUSE' to be recognized and the character recognizer gave the output as 'RVSE', we clearly see that the recognizer has misclassified 'U' as 'V'. We can have a dictionary look up and see that there is no word spelled 'RVSE' and there is high probability that the given word is 'RUSE' instead.

Similarly contextual knowledge can be applied when recognizing documents related to special areas. If a medical document is being recognized and the word 'EAR' is recognized as 'CAR', then by applying domain knowledge there is high probability that the 'E' has been misclassified as 'C'.

CHAPTER 5

CONCLUSION

Character Recognition is a very extensive system with several stages from preprocessing to classification and several algorithms involved at each stage. We have described a robust algorithm for extracting features from character images, which is the essence of character recognition. The quality of these features is demonstrated by comparing it with the connectivity features algorithm. These gradient features result in high recognition rates in both handwritten as well as machine printed characters. The K nearest neighbor algorithm was used for classification.

5.1 Future work

There are several other classification algorithms that can be used for character recognition. More experiments can be done by implementing learning vector quantizers, Bayesian or artificial neural networks classifiers. A larger or more extensive data sets can be used to evaluate the performance of the method.

A facility for unclassifiablity of a character can be incorporated in the classification algorithm if it finds the features are too difficult to assign to a certain class. There can also be several classification algorithms, and a voting scheme can be built where the class with the maximum number of votes is the output of the recognition.

BIBLIOGRAPHY

[1] R. Plamondon and S.N. Srihari, "On-line and Off-Line Handwriting Recognition: A Comprehensive Survey", IEEE Transaction on Pattern Analysis and Machine Intelligence, VOL 22, NO. 1, JANUARY 2000.

[2] T. M. Cover and P. E. Hart (1967), "Nearest Neighbor Pattern Classification", IEEE Transactions on Information Theory, Volume IT-13(1), pp.21-27, 1967.

[3] W.K. Pratt, Digital Image Processing, John Wiley (1991).

[4] Matthias Zimmermann, "The Homepage of the IAM Database A Database for Off-line Handwriting Recognition Research", http://www.iam.unibe.ch /~zimmerma/iamdb/iamdb1.html#1.

[5] G. Srikantan, S.W. Lam and S.N. Srihari, "Gradient based Contour Encoding for Character Recognition", Pattern Recognition, Vol. 29, No. 7, pp. 1147 1160, 1996..

[6] Thien M. Ha and H. Bunke, "Image Processing Methods for Document Image Analysis", Handbook of Character Recognition and Document Image Analysis, pp. 1-47 Eds. H. Bunke and P.S.P. Wang.

[7] Scott Teresi, "Handwriting Recognition Using a Neural Network Character Classifier", May 1998.

[8] J.C. Perez, E. Vidal and L. Sanchez, "Simple and Effective feature Extraction for Optical character recognition".

[9] C.F. Coulmas, "The Writing Systems of the World", Blackwell, 1980.

[10] Pavan Kumar Devineni, "Handwriting Recognition" Masters thesis, University of Nevada, Las Vegas, August 2004.

[11] I. Guyon, M. Schenkel and J. Denker, "Overview and Synthesis of On-line cursive handwriting recognition techniques", Handbook of Character Recognition and Document Image Analysis, pp. 1-47 Eds. H. Bunke and P.S.P. Wang.

[12] Yu Jiangsheng, "Method of *k*-Nearest Neighbors", September 3, 2002.

[13] Jiawei Han and Micheline Kamber, "Data Mining Concepts and Techniques", Morgan Kaufmann, 2001.

[14] M. Sridhar and F. Kimura, "Segmentation based Cursive Handwriting Recognition", Handbook of Character Recognition and Document Image Analysis, pp. 1-47 Eds. H. Bunke and P.S.P. Wang.

[15] Sameer Singh, "Shape Detection Using Gradient Features for Handwritten Character Recognition", University of Plymouth, UK.

[16] Zhang ping and Chen Lihui, " A novel feature extraction method and hybrid tree classification", Pattern Recognition Letters 23 (2002) 45 – 56.

[17] A. Amin, Humoud Al-Sadoum and S. Fischer, 1996, "Hand-printed Arabic character recognition system using an artificial network", Pattern Recognition 29 (4), 663-675.

[18] M. Blumenstein and B. Verma, "Neural–based solutions for the segmentationand recognition of difficult handwritten words from a benchmark database", Proc. 5th

International Conference on Document Analysis and Recognition, pages 281–284, Bangalore, India, 1999.

[19] P. D. Gader, M. A. Mohamed, and J. H. Chiang, "Handwritten word recognition with character and inter–character neural networks", IEEE Transactions on Systems, Man and Cybernetics – Part B, 27:158–164, 1994.

[20] P. Gader, M. Whalen, M. Ganzberger, and D. Hepp, "Handprinted word recognition on a NIST data set", Machine Vision and Applications, 8:31–41, 1995.

[21] A. L. Koerich, R Sabourin, C. Y. Suen and A. El-Yacoubi, "A Syntax-Directed Level Building Algorithm for Large Vocabulary Handwritten Word Recognition".

[22] Alessandro L. Koerich, "Large vocabulary off-line handwritten word recognition", August 19, 2002.

[23] Dilder Guillevic and Ching Y. Suen, "Cursive Script Recognition applied to the Processing of Bank Cheques", CENPARMI, GM-606.

[24] U. Kressel and J. Schurmann, "Pattern Classification techniques based on function approximation", Handbook of Character Recognition and Document Image Analysis, pp. 1-47 Eds. H. Bunke and P.S.P. Wang.

VITA

Graduate College
University of Nevada, Las Vegas

Ravi Kiran Reddy Veligati

Local Address:
4217 Grove Cir,
Apt # 3,
Las Vegas, NV, 89119

Degrees:
Bachelor of Engineering, Computer Science, 2000
Chaitanya Bharthi Institute of Technology, Osmania University, Hyderabad, India

Thesis Title: Handwritten Character Recognition using a Gradient based feature extraction.

Thesis Examination Committee:
Chairperson, Dr. Evangelos A.Yfantis, Ph. D.
Committee Member, Dr. Ajoy K.Datta, Ph. D.
Committee Member, Dr. John T. Minor, Ph.D.
Graduate Faculty Representative, Dr. Venkatesan Muthukumar, Ph.D.