Pattern Recognition Letters 14 (1993) 291-302
North-Holland

April 1993

PATREC 1112

Recognition of handwritten and machine-printed text for postal address interpretation

Sargur N. Srihari

Center of Excellence for Document Analysis and Recognition (CEDAR), State University of New York at Buffalo, Buffalo, NY 14228-2567, USA

Received 28 February 1992
Revised 24 November 1992

Abstract

Srihari, S.N., Recognition of handwritten and machine-printed text for postal address interpretation. Pattern Recognition Letters 14 (1993) 291-302.

Postal address interpretation is the task of assigning to letter mail pieces a delivery point encoding, e.g., ZIP+4 Code. The encoding is determined from images of destination addresses on mail piece faces: addresses that are handwritten, are of poor-quality machine printing, are incomplete or incorrect. This paper describes several recognition algorithms used in the interpretation of handwritten and machine-printed address text (digits/symbols/alphabets/words).

Keywords. OCR, address interpretation, handwriting recognition, neural networks.

1. Introduction

Mechanized mail processing performed by the United States Postal Service [UPS] requires the encoding of each mail piece by a string of digits. The first five digits of this code are referred to as the ZIP Code, and together with the next four as the ZIP + 4. Typically, the first three digits of the ZIP Code represent the state, city and postal sectional center facility [SCF] information, and the next two digits represent the post office. The four-digit add-on encodes mail down to the carrier level by including information such as a block face, post office box number, etc. Encoding mail to the carrier drop-off sequence, known as destination encoding, requires an additional two digits corresponding, for example, to the last two digits of the street address.

Correspondence to: S.N. Srihari, Center of Excellence for Document Analysis and Recognition (CEDAR), State University of New York at Buffalo, Buffalo, NY 14228-2567, USA.

The main classes of mail are letter mail (envelopes and postcards), flats (e.g., magazines) and parcels. In the case of non-pre-barcoded letter mail, the objective of address interpretation is to determine the destination code and spray it as a POSTNET bar-code, typically on the bottom right-hand corner of the face of the mail piece. The bar-code is utilized by subsequent bar-code reading machines.

The major operational steps in destination encoding are: image acquisition, locating the address field, reading the text, and determining the destination encoding. The major functional subsystems are: (i) locating the block of text corresponding to the destination address [11], and (ii) interpreting the destination address. The location task is separable from the interpretation task; for example, since it is possible to determine the spatial extent of the address text without fully recognizing and interpreting all parts of the address. The task of interpretation involves recognizing characters, words, punctuations, etc., i.e., mapping shapes to symbols, and interpreting the recognized symbols with the aid of directories. In the case of recognizing handwriting and poor quality machine-printing it is necessary to effectively use contextual information which is in the form of postal directories such as the ZIP + 4 file and the Delivery Sequence File.

It is necessary to separate machine-printed address interpretation from handwritten address interpretation due to several reasons: (i) handwritten words have more variability in shape than machine-printed words and thus require different techniques, e.g., in the machine-print case the process can begin by attempting to recognize all individual characters—an approach that is infeasible in the handwritten case where lexicons are necessary for word recognition, (ii) handwriting consists of elongated strokes whereas machine-print consists of regular blocks, and so the segmentation into lines and words is more complex than with machine-print, (iii) machine-printed addresses appear with more varied background than handwritten addresses, so a simpler image preprocessing, e.g., thresholding, suffices for handwriting, and (iv) there is significant advantage to be gained by utilizing gray-scale information in the recognition of machine-printed characters—an advantage not apparent with handwriting recognition.

2. Handwritten address interpretation

The goal of handwritten address interpretation [HWAI] is to assign a destination code to a block of handwritten address text [1]. The main task is to locate the ZIP Code or ZIP + 4 Code if present and to recognize its digits. Since the four digit add-on is usually absent, it is necessary to recognize street address information (which typically consists of a street number and a street name) to infer the four-digit add-on. If the five-digit ZIP Code is absent, incorrect, or ambiguous, then it is necessary to recognize city and state words. Finally, to overcome ambiguity in other recognition steps it may be necessary to recognize the personal or firm name.

The recognition of digit strings is usually more reliable than alphabetic words when no lexicon is available. This is because there are fewer digits than alphabets and digits touch each other less often. Thus a strategy is to first recognize digit string fields and then to utilize word lexicons derived from postal directories based on the recognized digit strings, so as to hypothesize the identity of alphabetic fields.

Our discussion of the HWAI task is divided into 5 sections: (i) digit string segmentation, (ii) digit recognition, (iii) character (alphabet) recognition, (iv) word recognition, and (v) control structure.

2.1. Digit string segmentation

The goal of digit string segmentation is to partition a previously extracted digit string image into regions, or snippets, each containing an isolated digit. The following are major phases: connected component analysis, estimation of the number of digits (obtained by involving a digit recognizer), and grouping and dissection.

The digit string segmentation procedure needs to: segment digits, estimate the number of digits, and handle exceptional cases. We use a recognition-aided iterative method [5]. A combination of statistical and structural techniques that utilize digit recognition results are used to segment digits and estimate the number of digits.

The method is based on a row-partition of the image, where the set of rows in each partition is contiguous. The number of rows in a partition monotonically decreases and can be expressed as a function of the iteration number. The row-partition influences grouping properties of broken digits and word length estimation. As the number of rows per partition decreases, the input image is analyzed more finely. Multiple components that are near each other may be considered as one digit in earlier iterations, but are separated and identified correctly in later iterations.

The digit string segmentation procedure has a parameter with three settings—which allows it to: (a) estimate the number of digits, (b) force it to estimate 5 or 9 digits (for ZIP Code segmentation), or (c) force it to consider one specific word length. In the first two cases the number of digits is estimated as follows: in the first iteration, aspect ratio of the word image is used to estimate the number of digits; in subsequent iterations it is estimated using a constrained linear regression model. Since digits which are recognized with high confidence values are removed on each iteration, the effective contribution of the digits to the density of the word image can be recorded. Using this information, a least squares linear model is used, with image density as the independent variable and the number of digits recognized as the dependent variable, for the number of digits estimation. An image density of zero implies that all digits have been removed. Thus, setting density to zero in the least squares equation yields an estimate to the number of digits. The segmenter uses a digit recognizer based on the polynomial discriminant method.

A hierarchy of grouping types for digits has been empirically determined through an analysis of the type and frequency of broken digits. Broken digits are grouped using proximity relationship of digit strokes and the digit recognition information. That is, if two neighboring components satisfy proximal constraints and the components are recognized as a digit with high confidence value, the components are grouped.

If there are multiple-digit connected components they are identified in the last iteration, when each partition has a single row. Four basic splitting methods are applied in sequence inside an interval centered around the likely splitting column. The four splitting methods used are as follows. In the *histogram* method vertical density projections are calculated, and the column with minimal value is chosen for a vertical split. The *upper-lower contour* method utilizes the upper and lower contours of the components as determined by the first and last pixel in each column. A successful split will be made by connecting the valley to the peak with a straight line. The *upper contour* method keys on valleys in the upper contour. An upper contour valley and a local minimum within this valley are determined, and a modified hit-and-deflect splitting technique is applied, starting from the minimum point. The *lower contour* method performs

Table 1
Handwritten digit recognition performance Training set: 18,468 digit (br) Testing set: 2,711 digits (bs)

Recognizer (abbreviation)	Features (no., type)	Classification algorithm	Correct classification
P	Pixel pairs (1241, binary)	Linear discriminant	93.6%
M	Topological—arcs, caves, holes; Histogram-based—left most pixel pos. (26, integers and real)	Mixed structural/statistical	89.4%
S	Strokes, holes, contour profiles (6)	Rule-based (hierarchical)	86%
C	Contour features from chain code—arc, bay, curl, inlet, null, spur, stub, wedge (8)	Rule-based (130+rules, decision tree)	83%
Combination P, M, S, C	Classifier decision confidences (40, real)	Neural network[a] (20 hidden units)	96%
Cnn	Contour chain code (464, integers)	Neural network (80 hidden units)	93%
H	Histogram (72, real in range [0,1])	Neural network (40 hidden units)	95%
Gr	Gradient (192, binary)	Neural network (40 hidden units)	96.4%
Ga	Gabor coefficients (72, real)	Neural network (40, hidden units)	94%
Mo	Morphology (85, real in range [0,1])	Neural network (35 hidden units)	96%

[a] Trained on 6,432 bd digits

essentially the same operation as the upper contour splitter except that the lower contour splitter runs from a maximum in the peak in the lower contour.

The segmenter has a correct segmentation rate of 93% when the input is specified as a ZIP Code (5 or 9 digits), and 83% when the number of digits is unspecified.

2.2. *Digit recognition*

The goal of digit recognition is to assign a snippet into one of 10 categories, $0 \cdots 9$, or to reject it as being unrecognizable as a digit.

Handwritten digits have wide variability—from neat printing to broken and touching characters. It is desirable for the recognition method to have a high correct recognition rate, low error rate, and a low reject rate. A recognition algorithm has two components: features derived from the character image and the method of classification. We began by exploring several feature sets and classification methods as described in the literature.

Four digit recognizers were implemented: P—polynomial discriminant function [13], M—a mixed approach classifier [4], S—a stroke-based recognizer [8] and C—a structural contour-based chain-code classifier [3]. The classifiers were based on different methodologies: statistical, structural, and syntactic. The features employed by each recognizer are listed in Table 1. Each of the recognizers were trained on a set of 18,468 well-formed digits. They were tested on 2,711 automatically segmented digits (snippets). The test set contains certain digits, about 2%, that are of very poor quality for recognition.[1]

Recognizer combination

A design philosophy is that if several recognizers are independent and uncorrelated, then their decisions can be merged using a combination scheme so as to yield a higher performing method. Although the four recognizers P, M, S, and C are not close to the highest achievable performance, their combined performance is near the ceiling.

Neural network classifier

Several approaches to combining decisions from multiple recognizers are possible. However, optimality is difficult to achieve due to practical limitations such as finite data, high dimensionality and unknown density function. Artificial neural-networks have been shown to approximate Bayesian decisions [9]. A neural network trained using back-propagation [6] provides an effective means of estimating the optimal decision, without dealing with those limitations directly.[2] Input consists of decisions from multiple classifiers, represented in the form of a vector of confidence values.

Recognizer C was subsequently re-implemented with a neural network classifier (Cnn) based on a feature vector of size 464 (derived from the same 8 features and considering 16 locations and 8 direc-

[1] The training and testing sets are referred to as br and bs digits in the CEDAR/USPS database.

[2] Our neural network classifier, named mlp, is a general multi-layer layer feed-forward network trained with backward error propagation. For each problem, the user provides a training file and specifies the configuration of the network. The number of input units is determined by the feature classes. There can be arbitrarily many hidden layers with arbitrarily many hidden units. All units in adjacent layers are fully connected. The activation functions used are standard sigmoid functions. During training, a target value of 1.0 is set for the desired output, and 0.0 for all other units. The algorithm seeks to minimize the sum of squared errors between the actual outputs and the targets. Both error term and momentum term are used in weight update. This describes a plain backprop trained network at the highest functional level.

Looking deeper into the implementation, several small modifications are found. The sigmoid function range, normally between 0 and 1, is shifted between -0.5 and 0.5. In addition, its derivative, originally $f(x)(1-f(x))$, is augmented by 0.1. These changes improve the rate of convergence. The learning rate and momentum can also be adjusted at run time. A few other options are available for improving the generalization of the network. The most frequently used is weight truncation. This is different from the weight decay proposed by Hinton. By limiting the difference in weight magnitudes, learning is distributed more evenly among all weights. The user can also specify an alternative maximum likelihood criterion function instead of the sum of squared errors. A tolerant range can be specified so that output greater than certain level will be considered as 1, and output below a certain level will be considered as 0. This scheme places more emphasis on getting the output units to respond with the 'right' sign than on approximating the exact target value. These options, in addition to different network configurations, provide some variability in how a classifier can be trained.

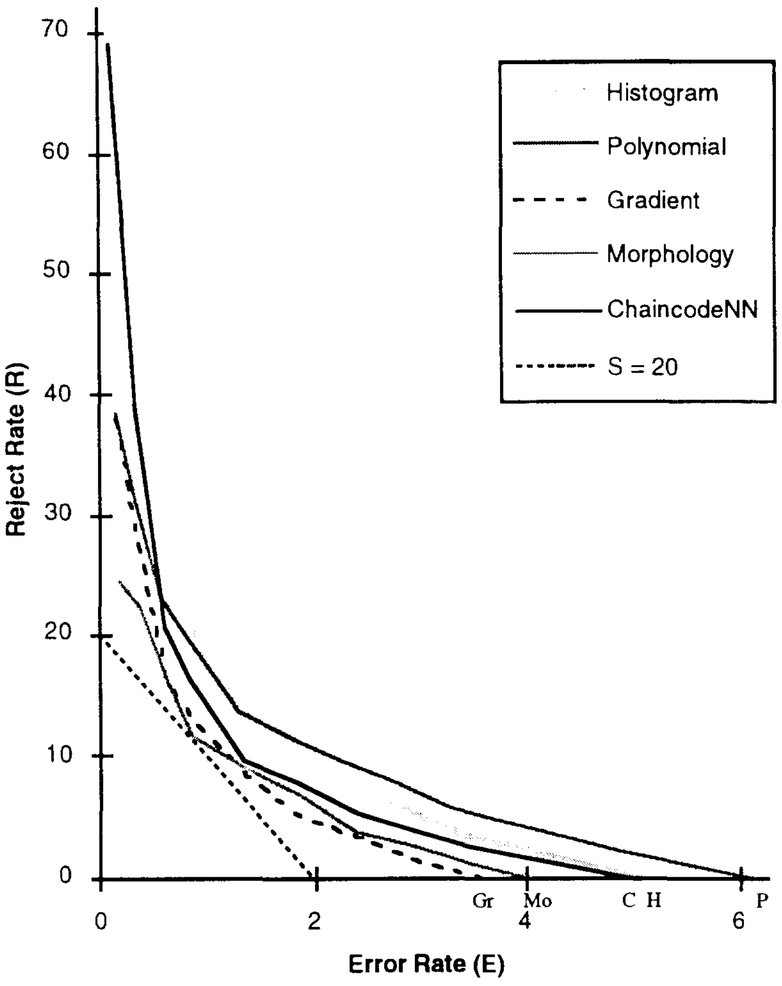

Figure 1. Handwritten digit recognition.

tions), and a resulting performance of 93%. The 10% improvement in performance between C and Cnn is due to somewhat different feature information and better tuning to the training set. When P and Cnn were combined, the resulting performance was 94%.

We further explored four other feature sets, each of which utilized a neural network classifier: H-histogram, Gr-gradient, Ga-gabor, and Mo-morphology. Recognizers Ga, Gr, and H were originally designed for gray-scale machine-printed character recognition (see Section 3.1). Classification in each case was based on a neural network trained using back propagation. The performance of each of these four recognizers is near the ceiling and hence their combination does not yield additional dividends.

The performance of the five neural network recognizers are also shown in Figure 1, with a reject option. If 10 rejects are deemed to be as costly as one error, then the objective is to find a recognizer that minimizes $S = 10E + R$ where S is the recognizer accuracy, R is the reject rate and E is the error rate. The error rate is defined as the proportion of those that are accepted (not rejected). The data points for the graph of each recognizer were obtained by first creating a file of recognition results for all test images. The file was sorted by decreasing recognition confidence. A given error rate is obtained by starting from the top and including as many errors as permitted and rejecting the remainder. Recognizer M achieved the lowest S value ($S = 20$).

2.3. *Character recognition*

Handwritten character recognition is more complex than digit recognition, since: (i) there are more classes of (upper and lower case versions of 26 alphabets), and (ii) there are two types of character shapes: those printed in isolation and those that are present in cursive writing.

We implemented one recognizer that uses structural features extracted using morphological operators and a neural network classifier. The feature vector consists of 85 real values: 4-direction concaves, strokes (horizontal, vertical and diagonal), end-points, cross-points (all extracted using morphological operators), and three moment features. The feature set is size-independent (requiring no size normalization). A training set of 10,134 lower case and 11,134 upper case characters were obtained from (non-postal NIST data. An additional 9,166 segmented cursive characters were obtained from postal images. The recognizer was tested on 877 lower case, 1,296 upper case, and 615 cursive characters. The results are shown in Table 2.

2.4. *Word recognition*

The handwritten word recognition [HWWR] procedure accepts a word image as input, as well as a lexicon, i.e., a list of possible identities of the word (which may be derived from knowing the ZIP Code and a postal directory). Its goal is to provide an ordering of words in a lexicon, ordered according to the level of match with the word image. We are exploring several approaches to the HWWR task, two of which are described here: the hidden Markov model approach and the hypothesis generation approach.

Table 2
Handwritten character recognition performance
Training set: 10,134 lower case (NIST)
11,134 uppercase (NIST)
9,166 segmented cursive (postal)
Testing set: 877 lower case (NIST)
1,296 upper case (NIST)
615 cursive (postal)

Data	Top choice	Top 2 choices	Top 3 choices
Upper case	93%	97%	98%
Lower case	85%	94%	97%
Cursive	63%	73%	78%

2.4.1. *Hidden Markov model*

This method is based on first dividing the word into segments and then recognizing the segment string. Recognition is based on the hidden Markov model (HMM), where the observed segments are referred to as symbols and the characters giving rise to the symbols as hidden states. The Viterbi algorithm and a dictionary are used in the final recognition.

The segmentation algorithm, which is based on mathematical morphology, translates the 2-D image into a 1-D sequence of sub-character symbols. This sequence of symbols is modeled by a continuous density variable duration hidden Markov model. Thirty-five features are selected to capture shape information from the character symbols onto the feature space. While each character symbol is modeled as a mixture Gaussian distribution, the linguistic constraint is modeled as a Markov chain. The variable duration state is used to take care of the segmentation ambiguity among the consecutive characters. A modified Viterbi algorithm, which provides a number of globally best paths, is adapted by incorporating the duration probabilities for the variable duration state sequence. The general string editing method is used at the post-processing stage to rank the lexicon to give the final interpretation of the image.

2.3.2. *Hypothesis generation and reduction*

The Hypothesis Generation and Reduction (HGR) method uses three constraints to hypothesize and rank all plausible interpretations of the word image: character shapes embedded in the word, character transitions allowed in the dictionary, and spatial compatibility of character pairs. First, whole character shapes embedded in the word are isolated and recognized by the segmentation and the character recognition system. The word is scanned for stroke configurations corresponding to possible characters. This produces many word interpretations which must be ranked and reduced in number. This is done using the allowable character transitions of the

Table 3
Handwritten word recognition performance

Classifier	Lexicon size 10	100	1,000
HMM	90%	69%	45%
HGR	89%	82%	69%
Combination	93%	86%	80%

language and implied spatial context (compatibility) of character pairs. The lexicon allows ranking of word hypotheses using character diagram and trigram tables. String matching of the best hypotheses with the lexicon produces the final interpretation.

The approach is based on segmentation and recognition of individual cursive characters in the word. Since most handwriting contains predictable flaws, the segmentation process is actually a search that generates a series of segmentations of the word. Potential characters that result from segmentation are passed to character recognition which ranks each segment with respect to each of the 26 characters. The system keeps track of the rank of each character as segmentation proceeds. Depending on the desired depth-of-search, the top *n* characters and their respective segmentations are chosen and the segmentation process continues from that segmentation point. This result is an *n*-way branch at each character candidate segmentation point in the word. The final result is a series of strings whcih represent the characters detected from each segmentation sequence of the word. Since many segmentations result in words that contain improbable (and illegal) sequences of characters, most strings are rejected. The remaining strings are passed to a matching algorithm for dictionary matching and ranking.

2.4.3. *Combination*

Ranked outputs of several uncorrelated HWWR algorithms can be combined so as to improve overall performance. Two HWWR algorithms were considered for combination. Both were tested on images of 94 handwritten street names involving different dictionary sizes. The results for the top choice are shown in Table 3.

2.5. *HWAI control structure*

The HWAI control structure invokes procedures for: *preprocessing* (thresholding, underline removal, etc.) *line segmentation*, locating the fields of the address (separating the address lines), *parsing* (assigning functional categories to the fields, e.g., city, state, ZIP, P.O. Box, etc.), *digit recognition* (recognizing the numeric fields of the address, e.g., ZIP Code and street numbers), recognizing other fields (state abbreviation, state name, and street name), and looking up postal directories to determine the destination code. The control structure needs to overcome several sources of uncertainty: the ZIP Code is often on the last line (90% of cases), but not always; underlines make segmentation of lines and digits difficult; poorly formed addresses without a ZIP Code may be incorrectly identified as having a ZIP Code; the segmentation of a ZIP Code into digits is difficult when digits touch, have connecting ligatures or have overhangs between digits; the recognition of digits needs to be able to handle a variety of writing styles and artifacts due to segmentation.

Parsing is aided by doing a first pass of recognition—where a digit/character recognizer is applied to each connected component. Ambiguity in recognizing handwritten characters and words is overcome by using contextual information, such as a lexicon derived from a directory that relates ZIP Codes with street addresses (the ZIP + 4 directory).

As an example, in a certain handwritten address the procedure locates and recognizes the five-digit ZIP Code to be 14222. The HWAI system locates and recognizes the street number as 703. Looking into the ZIP + 4 directory for 14222 it is found that there are only four ZIP + 4 entries with 703 as the street number. These are:

14222-1658	703 W Ferry St
14222-1416	703 Auburn Ave
14222-1220	703 W Delavan Ave # 1
14222-1449	703 Lafayette Ave.

This narrows down the lexicon of street words to about ten. The HWWR procecure is invoked to determine the street name to be 'W Ferry St' thereby resulting in the ZIP+4 encoding of 14222-1658.

Such a system developed at CEDAR has a performance of 75% accept rate, i.e., determining either 5- or 9-digit ZIP Code, with a 1.5% error rate [1]. The destination encode rate is 30% with a 4% error rate.

3. Machine-printed address interpretation

There are three aspects to machine-printed address interpretation [MPAI]: (i) character recognition, (ii) word recognition, and (iii) use of contextual information (postal directories) in address interpretation.

3.1. Character recognition

Current OCR technology can reliably recognize cleanly typed machine-print. The areas where performance needs improvement are in handling addresses with smear, broken characters, touching characters, poor contrast, textured backgrounds, etc. Machine-printed character recognition [MPCR] using binary images has a very large literature; a recent survey has been made by us [16]. We describe three methods here, P, F, and Ga, each of which was explored from the viewpoint of recognizing segmented gray-scale data. Gray-scale character recognition has previously received insufficient attention and it is was felt that this was a critical area for consideration.

Recognizer P is based on polynomial discriminant functions [13]. Our implementation uses products of certain pixel pairs as features and a linear classifier. We achieved a correct recognition rate of 89% on binary alphanumeric characters and 92% on gray-scale characters extracted from images of letter mail captured with a flat-bed scanner thereby demonstrating the value of gray-scale imagery.

Recognizer F is based on fuzzy template-matching, which uses the idea that missing object-pixels may be recovered from their neighbors. The key idea is to distinguish between different degrees of mismatch. In template-matching, two pixels having different values have a difference of 1. In fuzzy template-matching, the difference of two pixels depends also on their neighbors. To contrast the performance of F with that of conventional template matching, a training set of 30,280 binary characters, extracted from postal images, were used. Using a different test set, the top choice performance of the fuzzy matcher was 87% while the performance of binary template-matching was 80%.

Recognizer Ga is based on Gabor functions. Gabor functions maximize the joint resolution in the spatial frequency information domain. They also resemble striate cortical cell profiles. A set of selected Gabor functions act as spatially localized, oriented feature extractors. Least square solutions of those function coefficients provide compact representation of original images. We divide a 32×32 image into 3×3 overlapping regions of size 12×12. Four Gabor functions are fitted to each region and their least square solutions are sought. Since each coefficient is complex, this results in 4×2 features for each 3×3 region, or 72 features altogether. Classification is accomplished by a neural network trained on those coefficients, with 40 hidden units for digits and 100 hidden units for alphanumerics.

Performance of P, F, and Ga using 300 ppi gray-scale images are given in Table 4. In the alphanumeric case each recognizer used 62 classes and performance was measured with upper and lower cases folded together (to yield 36 classes). The three methods were combined using logistic regression [7]—which is equivalent to a neural network with no hidden units and using a weight constraint (this was necessary due to the large dimensionality of the feature space for a conventional neural network).

Two other recognizers, Gr and H, were also tested. Recognizer Gr uses gradient magnitude and direction as features. The gradients are obtained by convolving with a Sobel edge operator and thresholding the results. The image is divided into 4×4 regions. With 12 directions (corresponding to $0°$, $30°$, $60°$, ...) we get 192 binary features. Classification is done with 101 hidden units for digits (average of 192 and 10) and with 127 hidden units for alphanumerics (average of 192 and 62). Recognizer H uses 288 features derived from ver-

Table 4
Machine-printed character recognition performance
Training set: alphanumerics = 47,565; digits = 10,926
Testing set: alphanumerics = 26,357; digits = 5,420

Recognition method	Features (no., type)	Classifier	Digit recognition	Alphanumeric recognition
P	Pixel pairs (989, real)	Linear	99%	91.7%
F	Fuzzy pixels	Template match	96%	89.0%
Ga	Gabor coeffts (72, real)	Neural network (40 hu, 100 hu)	98%	92.9%
Combination (P, F, Ga)	Decisions, confidences (196 real)	Logistic	—	93.2%
Gr	Gradient (192, binary)	Neural network (101 hu, 127 hu)	—	94.5%
H	Histogram (72, real)	Neural network	—	93%

tical and horizontal histograms. The performance of H, P, Ga, and Gr with the reject option for alphabets and numerics are given in Figures 2 and 3 respectively.

3.2. *Word recognition*

A typical address consists of 4 lines, each with four 5-letter words. Thus it consists of 80 letters.

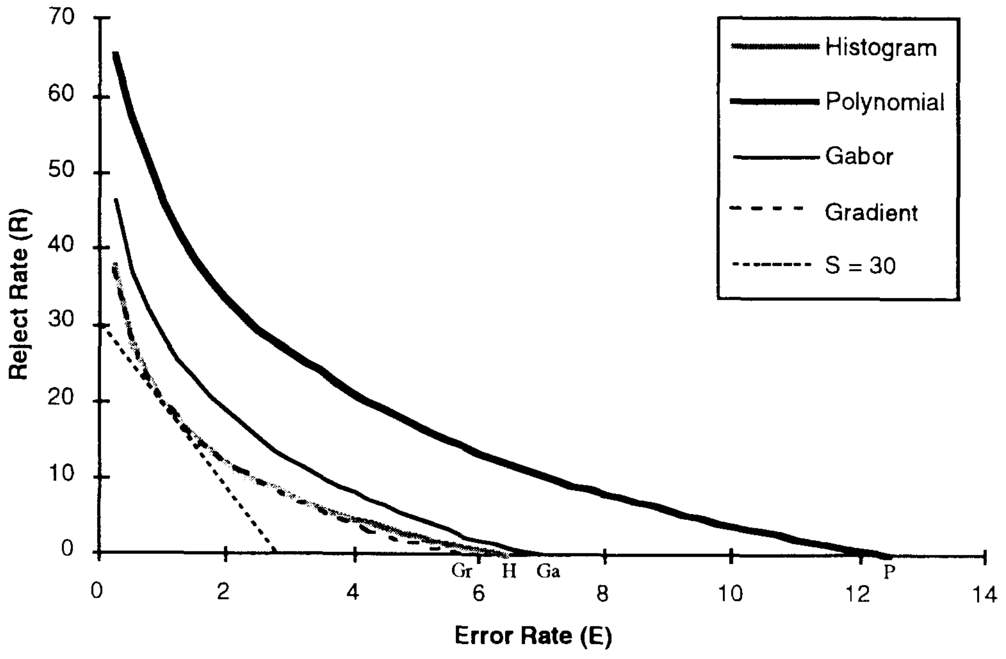

Figure 2. Machine-printed alphanumeric recognition.

Assuming a 95% correct character recognition rate yields 4 character errors or only a 75% correct word recognition rate. In order to achieve a significantly higher word recognition rate it is necessary to go beyond the character level and utilize word context.

A machine-printed word recognition [MPWR] algorithm associates a word image with one or more entries in a lexicon of possible word identities. Knowledge about the context of characters that occur within words in the lexicon has been used to improve the matching process. For example, it is wel known that the identity of a character is constrained by the identity of its predecessor within a word. There are many such constraints and a number of techniques have been developed to utilize them. Each approach has its particular strengths and is applicable to a subset of the word recognition problem.

Methods for MPWR are divided into three generic classes: character recognition-based, segmentation-based, and word shape analysis. Character-based and segmentation-based methods are analytical methods whereas word shape analysis is a holistic method. Each approach uses image information in recognition in a different way. In the first approach, contextual constraints are used only in the last stage, that is, the postprocessing stage after the character classes are decided. In the second approach, contextual information is used before class decisions are made, but after feature extraction. In the third approach, word context information is used directly in feature computation and matching.

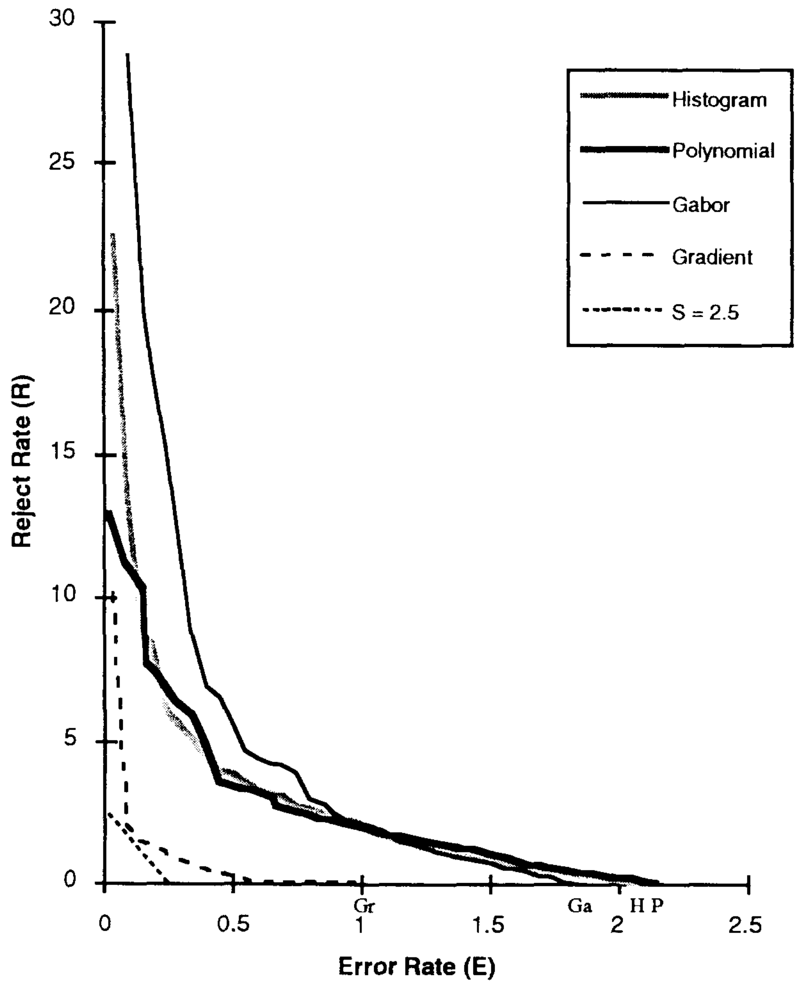

Figure 3. Machine-printed digit recognition.

3.2.1. *Character recognition-based*

In this approach, individual characters in a word are first isolated and each is recognized using MPCR. Character decisions are then postprocessed with a dictionary of allowable words to correct character recognition errors. These methods are suitable for cases where a reliable segmentation can be obtained and the segmented characters are not deformed by normalization. It is an appropriate strategy for shorter words which are easier to segment, and where little word-level context can be utilized.

3.2.2. *Segmentation-based*

An alternative to character-based techniques is to defer decisions about character identity and to perform segmentation-based word recognition. This is suitable for images where the characters can be easily extracted but are difficult to recognize in isolation. In this approach, feature descriptions of the extracted characters are assembled and matched to a similar representation of the words in a lexicon. These techniques effectively focus word-level knowledge on the recognition process and are suitable for situations where characters can be segmented and better recognized together with other characters in the word.

3.2.3 *Whole-word-based*

A third type of MPWR algorithm determines features from the whole word shape and uses this description to calculate a group of words in a dictionary that match the input. Such a method is suitable for images that are difficult to segment into characters, or when characters are distorted after extraction and normalization in isolation.

3.2.4. *Combination*

A method to integrate several MPWR algorithms has been developed [7]. Reliability is achieved by making use of multiple classifiers with uncorrelated errors at various stages. Final decisions are derived by applying a group consensus function that measures agreement among parallel classifiers.

The utility of this approach was demonstrated by an application to machine-printed words of a wide range of font styles and quality. The database is a collection of images of machine-printed postal words obtained from live mail. They were captured on a postal scanner at roughly 200 ppi and binarized. Lexicons derived with domain knowledge are expected to contain thousands of words. Early results show that the integrated algorithm performs significantly better than each of the three approaches.

The method has been applied to a database of postal word images scanned on an MLOCR at 200 ppi. In many cases the image quality was very poor. A typical lexicon in the experiment contained 500 words. A testing set of 1710 images resulted in a 96% correct rate within the top ten choices. The resulting system can be applied to recognize words in various positions in an address block image so that contextual address information can be utilized to improve the currently achievable level of sort.

3.3. *MPAI control structure*

Given the image of an address block, the control structure for MPAI first calls the segmentation/parsing routine. This routine divides the address block into words and assigns categories to each word such as ZIP Codes (z5), street numbers (sn), city word (cw), etc. The parsing routine that assigns these categories may produce several configurations for each address block [12]. A configuration is an assignment of word types to every segmented word in the address block. Currently the control structure looks at the top seven configurations for an address block. The configurations are tried sequentially. Since not every word in the address block is important to the control structure, it checks for duplicate configurations of the important words. Duplicate configurations are then ignored.

A constraint satisfaction solver is used to handle most address types, including PO Box, rural route and the more common street addresses [2]. The first two types are fairly easy to process since they usually involve only city, state, ZIP Code, box number and possibly route number. Street address processing is more complex. The street address processing routine makes use of our digit recognition routine (ZIP codes, street numbers, secondary numbers) as well as word recognition (city words, street words, street suffixes, street prefixes and postdirectionals) and ZIP + 4 query routines. The goal here is to identify a configuration of values from the recognition results that is consistent with the ZIP + 4 database. Regardless of the address type, the problem of address interpretation is regarded as a constraint satisfaction problem and solved using an intelligent backtracking routine.

The MPAI system has been tested on a 1200 address block image set. These were images that were captured by a current postal OCR whose performance on these images was 28% destination encoding with a 15% error rate. The performance of the MPAI system described here was 82% with 17% error rate.

4. Summary and conclusion

We have described several recognition components necessary for handwritten and machine-printed address interpretation. With handwriting, digit string segmentation performance is 93% for known number of digits (83% with unknown number), digit recognition performance is 96%, word recognition performance is 90% for small lexicons (10–100 entries), and destination encoding is 30% (with 4% error rate). With machine print, digit recognition performance is 99%, alphabet recognition performance is 94%, word recognition performance is 96% for 500-word lexicon, and destination encoding is 82% (with 17% error rate). Reaching the goals of 50% encoding for handwriting and 90% encoding for machine-print with low error rates (<4%) require further improvements in recognition module performance and in utilization of context.

Acknowledgement

Several individuals at CEDAR contributed experimental results described here. In particular, Dar-Shyang Lee was responsible for several of the recognition algorithms.

References

[1] Cohen E., J.J. Hull and S.N. Srihari (1991) Understanding handwritten text in a structured environment: determining ZIP Codes from addresses. *Internat. J. Pattern Recognition and Artificial Intelligence*, 5 (1 & 2) 221–264.

[2] Cullen, P.B., J.J. Hull, and S.N. Srihari (1992). A constraint satisfaction approach to the resolution of uncertainty in image interpretation. *Proc. Eighth Conf. on AI Applications*, March 1992, 127–133.

[3] Duerr, B., W. Haettich, H. Tropf, and G. Winkler (1980). A combination of statistical and syntactical pattern recognition applied to classification of unconstrained handwritten numerals. *Pattern Recognition* 12 (3), 189–199.

[4] D'Amato, D., L. Pintsov, H. Koay, D. Stone, J. Tan, K. Tuttle, and D. Buck (1982). High speed pattern recognition system for alphanumeric handprinted characters. *Proc. IEEE-CS Conf. Pattern Recognition Image Processing*, Las Vegas, 165–170.

[5] Fenrich, R. (1992). Segmentation of automatically located handwritten numeric strings. In: S. Impedovo and J.C. Simon, (Eds.), *From Pixels to Features, vol. III*, North-Holland. Amsterdam, 47–59. (1992).

[6] Hinton, G. (1992) How neural networks learn from experience. *Scientific American* 267 (3), 145–151.

[7] Ho, T.K. (1992) A theory of Multi-classifier Systems and its Application to Visual Word Recognition. Ph.D. Thesis, Department of Computer Science, SUNY at Buffalo.

[8] Kuan, C.-C., J.J. Hull, and S.N. Srihari (1991) Method and Apparatus for Handwritten Character Recognition. United States Patent, No. 5, 058, 182 (October 15, 1991).

[9] Lee, D.-S., S.N. Srihari and R. Gaborski (1991). Bayesian and neural network pattern recognition: a theoretical connection and empirical results with handwritten characters. In: I.K. Sethi and A.K. Jain, (Eds.) *Artificial Neural Networks and Statistical Pattern Recognition: Old and New Connections*. Elsevier Science Publishers, Amsterdam, 89–108.

[10] Palumbo, P.W., and S.N. Srihari (1986). Text parsing using spatial information for recognizing addresses in mail pieces. *Proc. Internat. Conf. on Pattern Recognition*, Paris, France, 1068–1070.

[11] Palumbo, P.W., S.N. Srihari, J. Soh, R. Sridhar and V. Demjanenko (1992). Postal address block location in real time, *IEEE Computer*, (July 1992) 34–42.

[12] Prussak, M., and J.J. Hull (1991). A multi-level pattern matching method for text image parsing. *Proc. IEEE-CS Applications of Artificial Intelligence Conf.*, Miami Beach, FL, 183–189.

[13] Schurmann, J. (1978). A multifont word recognition system for postal address reading. *IEEE Trans. Comput.* 27 (8), 721–732.

[14] Srihari, S.N. (1992). High-performance reading machines. *Proc. IEEE* 80 (7), 27–35.

[15] Srihari, S.N., E. Cohen, J.J. Hull and L. Kuan (1989). A system to locate and recognize ZIP codes in handwritten addresses. *Internat. J. Research and Engineering—Postal Applications*, 37–45.

[16] Srihari, S.N. and J.J. Hull (1992). Character recognition. In: S.C. Shapiro, Ed., *Encyclopedia of Artificial Intelligence*, 2nd edition. Wiley, New York, 138–150.