Off-line Handwriting Recognition:
A Review and Experiments

A.W.Senior

CUED/F-INFENG/TR 105

December 1992

Cambridge University Engineering Department
Trumpington Street
Cambridge CB2 1PZ
England

Email: aws@eng.cam.ac.uk

Off-line Handwriting Recognition: A Review and Experiments

Technical Report CUED/F-INFENG/TR 105

A.W.Senior
Cambridge University Engineering Department
Cambridge, England.

December 1992

Abstract

Computer handwriting recognition offers a new way of improving the human-computer interface and of integrating computers better into human society. A review of computer handwriting recognition aims and applications is presented, followed by a description of relevant psychological research. Previous researchers' approaches to the problems of on-line, off-line and isolated character handwriting recognition are described.

A complete system for automatic, off-line recognition of handwriting is detailed, which takes word images scanned from a handwritten page and produces word-level output. Normalisation and preprocessing methods are described and details of the recurrent error propagation network, Viterbi decoder and durational modelling used for recognition are given. Results are reported for experiments on a single-author cursive script database.

1 Introduction

1.1 General review

One of the wider aspirations of the field of artificial intelligence, if one forgets for the time being the longer-term goals of analysing and emulating human intelligence, is simply to enable computers to accomplish tasks which are natural to people. Thus computers should be better able to interact with people and act within the world in a less constrained manner than has previously been possible. These aims are reflected in the more modest attempt by the computer industry to make computers increasingly 'user friendly'. In this vein, computers have come out of laboratories and into homes and offices; we communicate with them using mice and keyboards rather than punched cards and toggle switches. Handwriting is a natural means of communication which nearly everyone learns at an early age. Thus it would provide an easy way of interacting with a computer requiring no training to use effectively, and a computer able to read handwriting would be able to process a host of data which at the moment is not accessible to computer manipulation.

After this argument, it seems surprising how little research has been carried out in recognition of handwriting by computer. One argument advanced is that the optimism in the speech recognition community about the abilities of machines which were about to arrive made people feel that other approaches were unnecessary. While some of the promises of speech recognition by machine have already been fulfilled, and researchers are still optimistic, some of the benefits have been slow to come, and people have thought again about what is required of human-computer interfaces. Though speech is a very convenient form of communication, it is not always the most practical. In noisy environments, those where silence is important, or where a large number of people must work with computers, it is clear that voice input is not the best solution. While computer professionals and secretaries would be loth to give up the convenience and speed of a keyboard, for computer

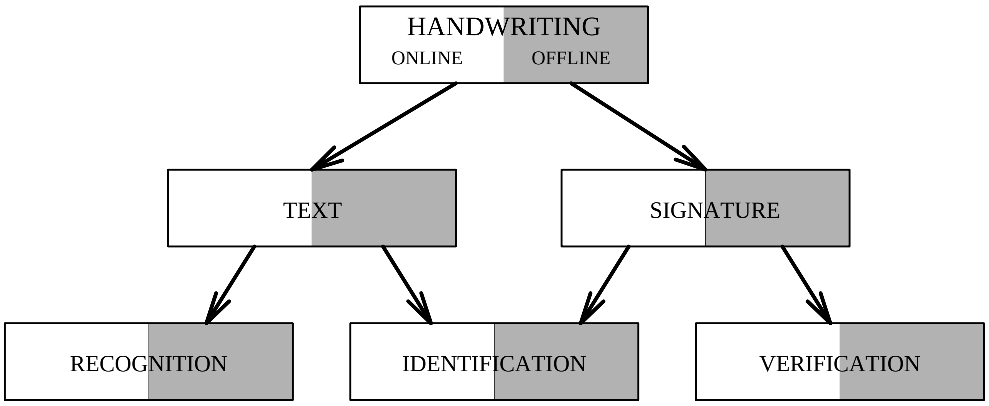


Figure 1: Subdivisions of machine handwriting recognition (after Plamondon & Lorette)

illiterates, portable or small-scale use, handwriting entry is clearly of practical value, leading to the growth in the last year or two of 'pen computing' [19]. As the Economist recently suggested, "Today's biggest prize in computer vision, however is text and handwriting.... A market probably worth billions for 'pen-based computers' awaits a solution." [9]

1.2 Classifications of the field

Having established the importance of automatic handwriting recognition in general, it is useful to examine the field more closely and to identify several areas of handwriting recognition with different applications and requiring different approaches.

On-line versus off-line

The major division of the field is between *on-line* and *off-line* systems. While there are a number of intermediate stages which could be distinguished, handwriting recognition systems are generally polarised between those receiving their data directly from some sort of pen device attached to the computer [15, 22, 78], and those which recognise handwriting already present on a piece of paper [69, 70] — a handwriting equivalent of the Optical Character Recognition (OCR) already widely used for reading printed matter. So far, the majority of systems have tackled the easier, on-line, problem where the time ordering of strokes is available as well as pen up/down information; overlapping strokes can easily be distinguished and stroke positions are accurately known. On the other hand, off-line systems have to cope with the vagaries of different pen types, wide strokes which frequently overlap and a complete lack of ordering information (though this is not *strictly* true as some stroke-ordering information can be determined off-line [75]). The growth of pen computing has seen much investment in on-line systems, but the difficulty of off-line recognition has so far deterred research. Although applications and techniques vary considerably, the general taxonomy of both off- and on-line handwriting analysis is similar as shown in figure 1 and described in the following section.

Author identification versus content determination

A second dichotomy in the field, orthogonal to the on-/off-line division is according to the information to be extracted from the handwriting. In both approaches, we may wish to determine the author of the writing, the words and meaning of what has been written, or both. However, these two requirements result in very different approaches, which diverge, and the processing involved may not overlap. Techniques also differ depending on whether the author is to be recognised from a signature or from a piece of text.

If we wish to determine the author of a piece of text or signature, the distinction is made between verifying that the author is the claimed author (for instance in security or banking applications) or merely deciding between a pool of known authors, for instance in an elementary writer-adaptation system. The former is the more useful, but of course the harder problem. Plamondon & Lorette [57] give an overview of handwriting systems, and a thorough review of signature verification systems.

Writer independence

The whole field of handwriting recognition is similar to the already well-developed subject of automatic speech recognition, which is often classified along the lines of speaker dependence, vocabulary size and isolated/continuous word. We find analogues to each of these in handwriting recognition.

Writer Dependence: Because of the enormous diversity of handwriting styles (as with spoken accents and idiolects) it is much more difficult to devise a system to recognise many peoples' handwriting than one which need only recognise that of a single author. Ways of overcoming the problem of multiple writers include a general recognition system which is able to adapt to the current writer, or having many systems, each recognising one style of handwriting (or one individual's handwriting) and a global system to select which should be used in any particular circumstance.

Vocabulary Size: The task of recognising words from a small lexicon is much easier than from a large lexicon (where words are more likely to be similar to each other). Thus an important criterion in assessing system performance is the size of the lexicon used. For most practical purposes a lexicon of 60,000 words is adequate, and for specific domains (such as reading cheque values in words or postal towns from envelopes) the vocabulary can be much smaller.

Isolated Characters: Segmentation of continuous speech into its component words has been found to be very difficult since in natural speech words run together with no silence between. For simpler tasks the recognition is made easier by forcing the speaker to pause between words. Similarly for handwriting it is hard to distinguish the boundaries between letters — the difference between '*ui*' and '*iu*' or '*vv*' and '*w*' is very slight. The task can be simplified by forcing the writer to separate letters, to write in capitals or to write clearly separated capitals in pre-printed boxes. For high accuracy data this may always be unavoidable since we already require these constraints to enable human readers to decipher official forms. We can place a similar constraint on cursive words, forcing the author to write each word in a separate box, but the word segmentation problem should prove less difficult, and less strict constraints could still ensure high accuracy segmentation of a page into its component words.

1.3 Applications

This section reviews some of the more important applications that may be envisaged for off-line handwriting recognition. Perhaps the largest immediate application is in using off-line techniques for on-line handwriting recognition. Since this involves throwing away the time information, at first this seems to be a poor method. However, by not relying on this, we free ourselves from a source of mis-information. For instance, in current on-line systems, an '*o*' written clockwise must be recognised differently from an '*o*' written anticlockwise, for in the time sequence information, they appear different. Someone who writes '*ana*' may subsequently return to extend the final *a*'s stroke to make the word read '*and*', but this change would be lost on a machine relying on the time-ordering of strokes. On the other hand, an off-line approach ignores these factors and simply looks at the final position of the strokes, reading the same information as is available to a human reader. Given this information, it may be seen that an off-line approach is also more likely to be extendable to a writer-independent reader, since one source of variation is eliminated. Conversely, it is precisely this sort of information which we need to preserve when creating an author verification system.

Cheques

One important commercial application for off-line cursive script is in the machine reading of bank cheques. Given a system with a limited vocabulary (about thirty-five words) but with high accuracy independent of author, one could imagine a system which would check that the amounts in figures and digits matched. Given a system that achieved high accuracy without a lexicon, one could also check that the payee corresponded to the account to be credited. (Of course, such a system might well include signature verification.) Given the number of cheques passing through the banking system each day, such a system, even if only able to confidently verify half of the cheques, would save much labour on a tedious and unpleasant job. In fact the project supported by the French post office has the goal of achieving a 0.01% error rate but permitting 50% of cheques to be rejected for manual sorting. Further, security would be improved since it is currently impossible to verify the signatures on cheques except those already singled out as suspect.

From postcodes to addresses

Off-line systems capable of recognising isolated digits have already been created and installed in many post offices around the world, as part of automatic mail-sorting machines. Given a system to locate the postcode on an envelope [43, 53, 77] this can be read and used to direct mail automatically. Clearly certain countries such as the USA are at an advantage in having digit-only zip-codes and several researchers have already tackled this problem [16, 24–27, 32, 37, 44] with reasonable success. A further enhancement that can easily be imagined is to read the whole address and either (a) sort envelopes without postcodes, or (b) in countries with limited resolution in the postcode, sort the post to determine the destination more accurately. Mail sorting can be seen as an ideal application for handwriting recognition, since it has a wide variety of levels of difficulty (from isolated digits written at predetermined locations on an envelope, up to complete determination of an address without a postcode) and admits of a certain amount of error while allowing a large rejection rate. Some mail is already misrouted, so paper mail is considered fallible and errors are already tolerated. Since there will always be some addresses illegible or incomprehensible to a machine, we can afford to give a "don't know" answer and send the item to a bin for human sorting.

Other applications

Wright [78] describes a prototype for an order entry system, which would use a pen computer to record deliveries, storing the orders in machine readable form for subsequent downloading at a depot, or even transmission via a portable telephone. Given the expense and fragility of even the dumbest of electronic slates (which would require the recognition task to be carried out at the depot, with only stroke information recorded on the portable machine) it is not hard to imagine that a cheaper, more robust system could be created with the deliveries entered, as before, on paper and read *off-line* at a central computer. In many applications there is still an interest in keeping a paper copy if only because of legal reasons. The idea of electronic signature theft may make acceptance of some pen-based computing systems very slow.

A variety of other office document processing systems using off-line handwriting recognition can easily be envisaged. Already many companies use electronic document processing systems which manipulate the scanned images of documents rather than the documents themselves. This is clearly a very data-intensive task, but one way of reducing the data burden is to extract the information and store text in ASCII (or perhaps in a richer format recording style commands). Further possibilities exist in reading handwritten documents for the blind or in automatic reading of faxes. Orders could be processed and dispatched automatically and standard enquiries replied to without human intervention. Other faxes could be fed directly into an in-house electronic mail system, providing at the very least automatic notification of fax arrival by reading the cover sheet, if not the full text of the document.

Of course, the advantages of handwriting recognition are not restricted to English or the Roman alphabet, though these have probably attracted most research. In the literature there is a wide range of papers describing handwriting recognition in a multitude of languages. The basic problems of handwriting recognition are common to all languages, but the diversity of scripts means that very different approaches may be used to take advantage of special features to make recognition easier. For example, Chinese [40], Japanese [29, 49, 50] and Hebrew [12,39] have been studied. Govindan & Shivaprasad [21] cite many more.

1.4 Psychology

To study the machine recognition of handwriting, it is worthwhile considering the way that people read and write, to gain insights into techniques which may be useful.

Several authors have studied people's ability to read, and Edelman, Ullman & Flash [13] quote the following results for experiments conducted without context:

> "In comparison, people recognize correctly 96.8% of handprinted characters [Neisser and Weene 1960], 95.6% of discretized handwriting [Suen 1983] and about 72% of cursive strings (see [Edelman 1988] appendix 1)."

A large body of psychological data has been gathered on the processes involved in reading type, some of which is applicable to cursive script. Taylor & Taylor [72] and Rayner & Pollatsek [60] provide useful reviews of the psychology of reading. Most research so far has concentrated on reading isolated words out of context and uses tachistoscopes to flash a word up for a very short time followed and preceded by masks to inhibit iconic memory. Some would argue that this gives us little indication of the processes occurring in normal reading where many words are permanently visible, but results are hard to prove in such a natural environment with many variables.

One of the fundamental facts discovered by research is the importance of recognition of words as single entities and not as the conjunction of their component characters. Taylor & Taylor cite work by Kolers & Magee, whose experiments involved training subjects on inverted text. (That is, text whose letters are all upside-down.) They trained two groups — one to read words and one to name letters — then each group was switched to the other task. No evidence was found that learning one task improved performance in the other, thus one may conclude that "relatively fluent reading requires familiarity with the shapes of words, but not with the letters in those words."[p195]

Further evidence for reading by words rather than individual letters is given by the *word superiority effect*. This is the term used for the phenomenon that a letter is better recognised (more frequently recognised correctly when presentation time is short enough to induce errors) when presented as part of a word than when presented either on its own or surrounded by arbitrary characters in a non-word. (See descriptions of Reicher's experiments in Rayner & Pollatsek [60] p78.)

As will be seen later, many approaches to handwriting recognition rely on detecting significant features in the writing, such as the strokes which go to make up individual letters. Bouma [3] investigated the features which people use to recognise isolated characters and described a partitioning of the set of lower case letters which grouped letters found to be psychologically close (i.e. easily confused). Bouma's classification is shown in table 1.

Short				Tall		Projecting
1	2	3	4	5	6	7
a s z x	e o c	r v w	n m u	d h k b	t i l f	g j p q y

Table 1: Bouma Shapes

Using these classes, words can be grouped according to their shape, so '*leg*' would become 627, but so also would '*fop*' which is seen to be similar in shape. Taylor & Taylor used these Bouma shapes for a study on the text of their own book, and found that the Bouma shape uniquely specified 6953 out of 7848 words in the sample. Table 2 summarises these results. The words of the book are classified according to each of four shape description techniques, and the number of

Figure 2: An Ambiguous Word

words of each shape are counted. Thus the outer contour[1] is enough to specify 2304 of the words uniquely, but there are 91 shapes shared by ten or more words each. Similar work can be found in Haber & Haber [23] who also give a decision tree which might be used to distinguish the letters of the Helvetica font by observing a limited set of features.

Table 2: Discrimination of words using word shape measures

Shape type	Number of Words sharing the same shape									
	1	2	3	4	5	6	7	8	9	10+
Outer Contour	2304	452	209	103	64	48	40	19	16	91
Outer Contour +initial letter	4088	694	248	113	55	35	21	17	11	23
Bouma Shape	6953	323	50	15	4	2	1			
Bouma Shape + initial letter	7431	185	14	1						

Taylor & Taylor propose a reading mechanism with three paths:

Whole-word process This is a rapid process taking perhaps 50-100ms which is based only on the pattern of the word as a whole, or the first half-dozen letters of longer words.

Letter-based process From 50ms after a word is presented, the individual letter identities are becoming available. (This could be understood as a progressive increase in the frequency of the filter used as suggested in work by Marr [42]). Outer letters are identified first, and may be used to adjust the first hypothesis of the whole-word process, or to generate a new hypothesis. These authors also hypothesise that word units (prefixes and suffixes) may be recognised as single units.

Scan-parse process This process is the slowest and uses the letter identities to produce a phonetic version of the written word, which can be used as additional evidence for the word identity.

It is interesting to note the work by Yamadori [79] and Sasanuma [66] which shows that damage to certain areas of the brains of Japanese readers can severely impair reading of Kana (phonogram) script whereas Kanji (ideogram) script is much less affected. This shows that different brain pathways must be used for the two script types and indicates that the mechanism of reading is more complex than it would at first appear.

A further fact which is evident from psychological studies, but which can be trivially verified by looking at figure 2, is the importance of context. The word in the figure could equally well be identified as '*clump*' or '*dump*' or even '*jump*', and it is only from the meaning or the grammar

[1] Just using the three short/tall/projecting categories.

of surrounding words that the two can be distinguished. To implement this discrimination in an automatic system, some language model must be introduced, to determine which words are likely given the preceding word sequence. Such language models vary in complexity, from simple word pair grammars (which state which pairs of words may be found adjacent) through to probabilistic models indicating the probability of a particular part of speech in the next word position.

A further hope of psychological research is to investigate the link between reading and writing. Several authors see learning to write and learning to read as inseparable processes and have produced models of handwriting generation to be used as a tool for handwriting recognition. This approach may be applicable beyond the field of psychology by yielding useful, human-like methods for handwriting recognition.

1.5 Optical Character Recognition

Off-line handwriting recognition has much in common with optical character recognition — the reading of print by computer. This application received much attention during the 1980s and is largely solved, with commercial packages available for microcomputers which can successfully read type in a variety of fonts and in a certain amount of noise [48]. The reason why the success of OCR has not carried over into handwriting recognition is the great variability in handwriting. For type, all letters 'a' are produced from a single archetype, and thus are very similar on the page, only being corrupted by a relatively small amount of noise in forms such as blurring, merging and slight positional variations. The process of handwriting is much more variable in all of these processes and suffers from variations due to other effects such as co-articulation — the influence of one letter on another. Also, with type, the symbols are usually distinct (except certain ligatures, as 'fi', which can be learnt as a separate symbol) so the problem of segmentation is not present.

As a consequence of this the techniques used are inadequate when presented with the greater variability in handwriting, and we find relatively little help in the OCR literature.

2 Existing Handwriting Recognition Systems

This section reviews some of the other handwriting systems which have been detailed in print. To do this it is convenient to classify them, as described already, into on-line, isolated character and off-line cursive script systems. These three approaches share the same basic structure in that most systems can be broken down into preprocessing, feature-extraction and recognition modules, but because of the different data available and the different demands imposed by applications, different approaches are taken to each of these three problems.

2.1 On-line systems

This area has seen perhaps the largest commercial interest so there are a number of systems on the market accompanying the new 'pen computers' [73]. Naturally most of the work behind these is unpublished, but there is nevertheless a large body of work in print.

One approach which has been taken to this problem is that of Morasso et al. [45, 46, 74]. In the first paper [74] they describe the preprocessing required for an on-line system. Subsequently they describe [45] a Kohonen self-organising network which is used to classify strokes into similar forms and build what the authors refer to as a 'graphotopic map'. The pen strokes are effectively vector quantised by this process, though the authors present no justification for a two-dimensional feature map in the space of pen strokes, on which a Kohonen net depends. Stroke information from pairs of letters is fed into a backpropagation network which is trained to identify the digraph. A 97% accuracy was achieved with the training set. Finally, [46] a second set of Kohonen networks is created, grouping consecutive strokes into characters and creating an 'allograph' lexicon of possible stroke sequences which are labelled with their character names. No results are presented in this paper.

Burr [10] treats only isolated characters and compares these letters to a set of 26 templates. Having deduced the most likely sequence of letters by dynamic programming, Burr's system performs a dictionary lookup using word roots and suffixes. Wright [78] gives a very coarse Freeman coding of the stroke sequence presented and looks this up in a table of previously encountered word codes from which he can deduce the word written. Nag, Wong & Fallside [51] encode the pen strokes using distance/angle (s, θ) coordinates and then use hidden Markov models to learn the statistical properties of the stroke data.

Following the insights of psychological investigations, many of the other approaches taken to on-line handwriting recognition are feature-based and involve identifying useful features in the handwriting [6]. Brown [8] gives a review of previous feature-based approaches, describing other researchers' feature sets before setting out his own. A recurring problem is that of segmentation — at which points to divide up the stroke information to decompose it into meaningful features. It has been found that the local maxima of curvature are natural segmentation points and these tend to be at minima of pen speed, which are easily determined. Kadirkamanathan [31] concentrates on scale-space filtering to identify these segments in noisy data.

2.2 Isolated characters or digits

Suen, Berthod and Mori [71] in 1980 provided a good review of handwriting recognition, concentrating on isolated character recognition — which had been the focus of research until then. They describe a variety of feature based approaches and divide these into global features (Templates or transformations such as Fourier, Walsh or Hadamard); point distributions (zoning, moments, n-tuples, characteristic loci and crossings and distances) and geometrical or topological features. The latter was, and has remained, the most popular technique, and involves separate detectors for each of several types of features such as loops, curves, straight sections, endpoints, angles and intersections. For instance, Impedovo, Dimauro and Pirlo [27] use cross-points, end-points and bend-points as their features, coding these as to their location in three horizontal and three vertical zones within each character. The encoded characters are then identified using a decision tree classifier. Elliman & Banks [14] also use features (end-point, junction, curve and loop) each of which is associated with a numerical quantity, such as curvature or length, before being decoded in a neural network (either backpropagation or adaptive feedback classifier).

Nellis & Stonham [52] and Hepp [24] use similar sets of global morphological features created by separately examining the left, right, top and bottom edges of each character. Both sets of authors feed this information into a neural network (Aleksander's discriminator and backpropagation respectively).

LeCun et al. [37] and Fukushima [18] take the approach of feeding a normalized bitmap image of the character to be recognised into their networks (multi-layered perceptron and neocognitron respectively). Both these networks are constructed from layers of identical feature detectors, which become more specialised and less location specific deeper in the network, until the outputs of the final layer are characters, independent of location in the image.

Yet another approach is taken by Idan & Chevallier [26] who also use bitmap images, but carry out the recognition task with a Kohonen [33] unsupervised network. Hinton, Williams & Revow [25] take a generation approach in which they generate spline models of digits and space Gaussian 'ink generators' along these. Then, by using elastic matching and determining the probability that each pixel was generated by an ink generator or their noise model, they find the best-fit model for each test digit presented. While not a practical system, this approach is interesting for its theoretical grounding and shares much with Edelman, Ullman & Flash [13]. Other works include Bienenstock [2] and Kimura & Shridhar [32].

2.3 Off-line cursive script

A brief review of several off-line cursive script recognisers is given by Simon [69], though this area of handwriting recognition is less thoroughly researched than those described above.

Papers by Božinović & Srihari [4, 70] describe what is perhaps the most significant work so far in the field of off-line cursive script recognition. In common with most of the on-line studies, they use a feature-based approach, searching a representation of the input image for useful features (such as hooks, cusps, straight line segments).

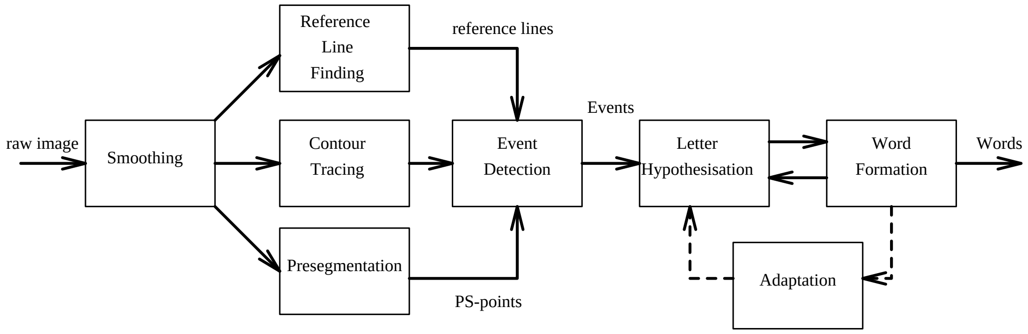


Figure 3: Schematic of Božinović & Srihari's System

Unusually, these authors generate their data by tracing handwritten words using an on-line graphics tablet, rejecting the time information and then creating a bitmap image. This image is then smoothed and slant corrected (the problem of slope correction is not addressed). To the resultant cleaned image, they apply three procedures in parallel (see figure 3): finding the baselines, contour tracing and presegmentation. Contour tracing involves the Freeman [17] coding of the image border, then processing this structure to create a tree describing the image topology.[2]

The segmentation task is an important problem in cursive script recognition, for it is difficult to decide, *a priori*, where one character ends and another begins. It is possible to sidestep this problem by treating words as a whole, or to confront it, and concentrate on finding the boundaries between characters. These authors choose an intermediate approach and seek presegmentation points — where the gaps between letters *might* lie — by a complex heuristic. A single character may straddle a presegmentation point, but characters are always bounded by these PSPs. The word image is then further processed by a series of event detectors which identify such features as dots, curves, strokes, loops and cusps (16 in all). From the presegmentation and event set, letter hypotheses are constructed based upon learnt statistics of letters.

Words are hypothesised via a stack method, where the most likely prefixes are stored and expanded until the word end is reached. After the first iteration of this procedure, the stack contains all the hypotheses for the first letter in order of likelihood. The top (most likely) hypothesis is then expanded by looking at what letters could follow. The resultant two-letter sequences are put onto the stack to be expanded when they are the most likely sequences. At the end of the word, the lexically correct word that is highest on the stack is chosen as the best match.

The authors conducted a number of experiments, using different writers and different lexica (780 and 7800 words). In the first experiment, non-slanting, non-sloping writing was used, and 78% recognition was obtained on the small lexicon, 48% on the large. With a 'mixed training scheme', results of 54-72% correct were obtained for three writers.

Edelman, Ullman & Flash [13] have also developed a handwriting reader (although they too collect their data using a digitising tablet, their techniques are applicable to off-line recognition) which relies on the alignment of letter prototypes. Here, anchor points (e.g. endpoints; turning points at top, bottom, left or right of characters) are found in the test word and these points are used to match the word against a set of prototype curves, coded as splines, which can be composed into lower-case characters. Using a lexicon, these authors obtained an 81% recognition rate on the training set and 47-53% on test sets by various authors. The authors estimate that their system has a 78.5% character recognition rate, and that it has the potential to achieve near-human performance. The training set sizes used were very small, but a large lexicon of 30,000 words was used.

[2]Use of the topology of a word at first seems to be a good technique for script recognition, but closer examination shows that this is not a good choice of invariant — vertical lines can often become loops, '*t*' strokes can often join up with other letters to create an enclosed area, and normally closed letters like '*a*' and '*o*' can often be left open in normal handwriting.

A problem where useful results can be obtained from real data through a task of limited vocabulary is that posed by the French post office. The task here is to recognise amounts written (in words) on postal cheques. Several authors [36, 38, 47, 54, 56] have attempted this problem and their results are as follows. Moreau, Plessis, Bougeois & Plagnaud [47] use the amount written in words as a check on the value determined from the digits and with 60% rejection achieve 0.2% error rate in recognising the total amount of the cheque, using the highly constrained grammar associated with the task. Paquet & Lecourtier [54] achieve 60% correct on the 50% of words which are well-segmented and Lecolinet & Crettez' [36] result is 53% words correct. Leroux, Salome & Badard's [38] system gets 62% words correct. The system described in Simon [69] achieves 86% correct on the test set (using a 25 word vocabulary). Tests using other authors' scripts (separately trained) give results between 47 and 91%.

3 An Error Propagation Network Handwriting Recognition System

In this section a complete off-line handwriting recognition system is described. The system is explained in sections and the whole system is shown in the block diagram of figure 4. First, the database on which the system is to operate is described, followed by a description of the preprocessing techniques needed to transform the data into a more usable form. Next, the recurrent network architecture is detailed and variations to the conventional multi-layered perceptron are explained. Finally the Viterbi decoder which produces word-level output is presented, with details of the enhanced durational modelling used. Earlier work on the same system was presented in less depth in Senior & Fallside [67].

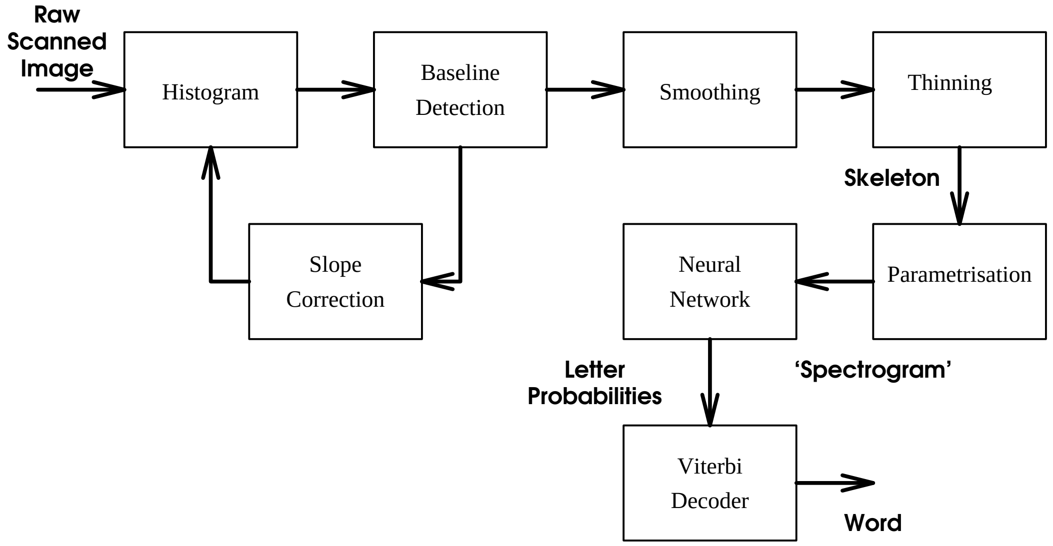


Figure 4: Schematic of the recognition system

3.1 Image acquisition and corpus choice

Words are written by a single author on an A4 sheet, each surrounded by white space to facilitate word extraction. These sheets were then scanned at 300 dpi resolution, in 8 bits to produce one file per page. These files were processed to extract each word into a separate file. Figure 5 shows one such word. All files are stored in TIFF [1] format.

The corpus chosen was the set of numbers written out as words ('one' to 'nineteen', tens from 'twenty' to 'hundred', plus 'thousand', 'million' and 'zero'). These words were chosen because they form a corpus useful for an application such as cheque verification, but the small vocabulary enables a reasonable study to be made in a short time and facilitates data collection. Ten exemplars of each of these words were taken: three to serve as a training set and four as test data, (a test set of 124 images), plus a further three to be used as a validation set (see section 3.4).

Subsequently, the data set has been increased in size and vocabulary by the collection of transcripts of the LOB (Lancaster–Oslo/Bergen) corpus [30]. This is an extensive corpus of modern English collected from a wide variety of sources, such as newspapers, novels and non-fiction books. The corpus as a whole contains a million words with a vocabulary of around 15,000 words. Writing out sentences from this corpus will give larger data sets permitting more significant results and laying the foundations for future work considering the addition of a grammatical analysis after the word recogniser, based on work already conducted on the LOB corpus, Kuhn & de Mori [34] for instance. The preliminary LOB handwritten database contains 768 training words, 219 validation words and 330 test words written by a single author, though these data sets are still too small for reasonable results to be expected. The overall vocabulary is 550 words.

3.2 Preprocessing

From the original scanned image, containing 8MB of information, all that is ultimately desired is the identity of the words on the page, which is in the order of bytes. One way of looking at recognition is as a process of information filtering with the ultimate aim of deriving this word information. In order to process the data effectively with a recognition technique such as a connectionist network, they must be reduced in number and transformed into a form more appropriate than a grey scale image, by the application of certain preprocessing techniques (the importance of data representation is discussed in [42]). In the same manner as filters, cepstra, Mel scale binning and vector quantisation are used to encode speech before attempting recognition, useful, invariant information must be extracted from the written words while discarding the vast majority of redundant variation.

A number of operations are involved in this sifting process. Each operation is described below, and figure 4 illustrates the whole process.

Histograms and baseline detection

Normalisation is of prime concern in preprocessing handwritten words. To normalise an image, it must be scaled so that letters are a standard height, and sheared so that letters are oriented vertically on a horizontal base. The character height is determined by finding the intuitively important lines which are shown running along the top and bottom of lower case letters in figure 5 — the half and base lines respectively (using the terminology of Brocklehust and Kenward [7]), with a centre line between the two. With these lines, the ascenders and descenders which are used by human readers in determining word shape (section 1.4) can also be identified.

The heuristic used for base line detection consists of the following steps:

1 Calculate the vertical density histogram by counting the number of black pixels in each horizontal line in the image. (Vertical and horizontal density histograms are shown on the right and bottom edges of figure 5.)
2 Reject the part of an image likely to be a descender.
3 Find the lowest remaining pixel in each vertical scan line.
4 Find the line of best fit through these points.
5 Reject all outliers, and calculate the new line of best fit. This is now considered to be the base line of the character.

Slope and slant correction

Given the estimate of the base line, from section 3.2, the writing can be straightened to make the baseline horizontal. This straightening is carried out by application of a shear transform parallel to the y axis. (See figure 6b.) Next, the base line and the half line can be recalculated, using the

Figure 5: Histograms and half, centre and base lines.

knowledge that the baseline is now horizontal, by scanning the histogram above and below the peak value, to find the 40% points of the slope. Slope correction can be carried out on whole lines to remove rotation in the scanned image or skewed writing, and then carried out on individual words to remove local tranformations.

Božinović and Srihari [4] detail a complex method for letter slant correction (slant is the tilting of tall strokes away from the vertical). This involves isolating areas of the text which are near-vertical strokes and estimating the slant of each by calculating the centroids of the upper and lower halves. These slant estimates are averaged, and a shear carried out to remove the slant. This procedure has been found to be very sensitive to the thickness of the writing and is unreliable when the writing is thinner than expected. Figure 6d shows a slant-corrected word.

Smoothing and thinning

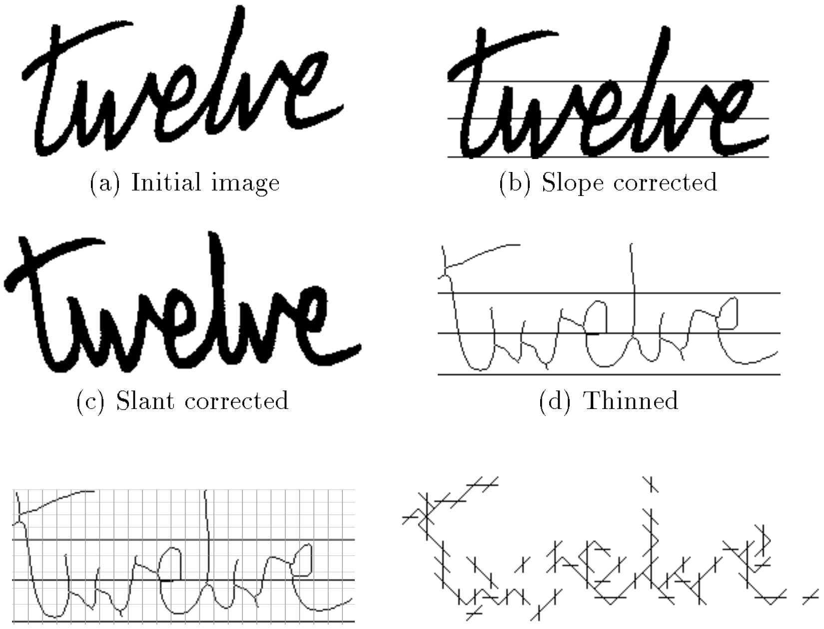
To remove noise from the image, either from scanning defects, or from applying transforms to discrete images, it is useful to smooth the image. This is carried out by convolution with a 2-dimensional Gaussian filter. It has been found that there is negligible noise on a scanned image when using a black fibre-tip pen on plain white paper, but degradation from this ideal situation is possible from a large number of sources (such as paper quality, age and condition; pen or pencil type; poor illumination when using a camera rather than a flat-bed scanner.)

Having normalised and smoothed the image, it is thresholded to leave every pixel black or white. Next an iterative thinning algorithm is applied to reduce the lines in the writing to one-pixel width so that the strokes can be followed later (see figure 6b). The algorithm used initially was that due to Gonzalez and Wintz [20] but it was found that this did not give one-pixel wide lines, so Davies' [11] p153 algorithm was applied when the first procedure had terminated.

3.3 Parametrisation

Now that the image has been reduced to a standard form, which highlights invariants of the words and suppresses spurious variations, the preprocessed image needs to be parametrised in an appropriate manner for input to the network which is to carry out the recognition process. The scheme chosen for this parametrisation is described below.

The area covered by the word is first divided into rectangles. (See figure 6e.) There are sixteen horizontal rows and a variable number of vertical frames of a width proportional to the



(a) Initial image (b) Slope corrected

(c) Slant corrected (d) Thinned

(e) Boxes for parametrisation (f) The parametrised data

Figure 6: Successive stages in the preprocessing.

row height. Thus long words have more frames than short words, but a given character will always occupy approximately the same number. For each of these rectangles, four bins are allocated to represent different line angles (vertical, horizontal, and the lines 45 degrees from these). Given this framework, the lines of the skeleton image are 'coarse coded':

Working from left to right in the thinned image, each stroke (a one-pixel wide line in the skeleton between junctions or end points) is divided into equal length segments (the length being determined by the character size) for which the centroid position and angle are calculated. The box associated with this segment's (x, y, θ) values is now 'filled'. Segments which are not perfectly aligned with the angles of the bins contribute to the two bins representing the closest orientations.

Figure 6f shows the input pattern schematically. Each line represents a full bin and its position and orientation correspond roughly to the position and orientation of the section of skeleton which gave rise to it. Because of the coarse coding, some line segments contribute to two bins and this is seen on the '*l*' stroke which is between the vertical and 45 degrees so both these lines are shown in the corresponding boxes in figure 6f.

Labelling and segmentation

The training data for the networks must be labelled to allow the network to associate input patterns with the correct letter outputs, one per frame. To do this, a segmentation scheme is required. The scheme used initially is an 'equal length' scheme, where each letter in any word is assumed (though this is clearly inaccurate) to occupy the same number of frames of input. Thus in an n letter word which takes k frames, the first $\frac{k}{n}$ frames are labelled with the first letter of the word. A further segmentation scheme has been implemented to allow 'bootstrapping' after the network has learnt an approximate input-output mapping. Here the network uses its knowledge of letter lengths to label frames automatically so for later iterations it can ignore the approximate, equal-length segmentation. Alternatively the data could be hand-segmented according to what is actually present in the data. This more accurate labelling could reasonably be expected to give more accurate training and improved results, but with the penalty of a large amount of work.

Given the importance of features demonstrated by studies of reading (see section 1.4), further improvements are to be expected if this parametrisation approach is combined with that of feature spotting. To derive a reliable set of rules for all the features used by Brown [8] would be very difficult, but there are certain important features such as '*i*' dots, turning points, skeleton junctions and endpoints which are easy to identify reliably with simple rules. By spotting these features with rules and indicating them in the data representation passed on to the network, more relevant information can be preserved, and the burden on the network's processing is reduced. This extra data increases the size of a frame from 64 bins to 161. Hence, the number of links in a network with 40 feedback units increases from 7035 to 13534 (a 90% increase) and the training time is correspondingly increased. Results with this enhanced preprocessing are compared to the basic method in section 3.6.

3.4 Recurrent networks

This section describes the 'recurrent networks' which are used to recognise the patterns of line segments present in the preprocessed data and to interpret them as letters. These networks are a type of connectionist (often termed 'neural') network which is to say that they are composed of a large number of very simple processing units with many interconnecting links. Each unit merely outputs a function of the weighted sum of its inputs, but the usefulness of such networks resides in the existence of training algorithms which can adjust the weightings to converge towards a desired function approximation — in this case the network is taught to recognise letters.

The recognition method chosen for the main part of the system is a recurrent error propagation network. Recurrent networks have been successfully applied to speech recognition and other dynamic problems where a series of time-varying signals needs to be classified [55, 63, 64]. A recurrent network is well suited to the recognition of patterns occurring in a time-series because the same processing is performed on each section of the input stream, but internal 'state' units are available to encode multi-frame context information about the preceding signal. Recurrent networks were found to perform better than Time-Delay Neural Networks [35, 76] for this task because of their better performance with time-distorted data. By repeated presentation of training examples, the network parameters can be adjusted automatically so that the network recognises letter shapes in a sequence of preprocessed frames of data.

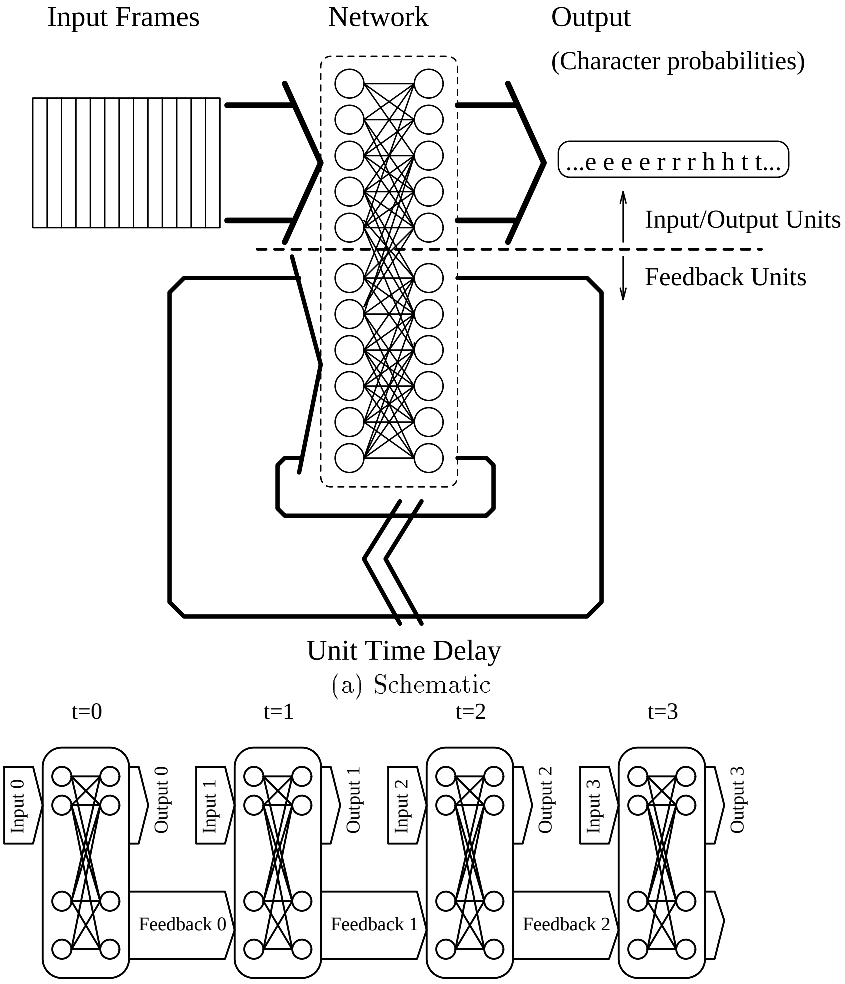
The recurrent network architecture used here is a single layer of standard perceptrons with sigmoid activation functions (as described by Rumelhart, Hinton and Williams [65]), however some of the output units are connected one-to-one to some of the input units with a unit time-delay, and these input units receive no external input (see figure 7a). The remaining input units accept the parametrised input described above and the 27 output units estimate letter probabilities for the 26 lower case letters plus a space character. During the forward pass, successive frames are presented at the input and results are fed back through the time delay, while output letter probabilities are read off from the other outputs. To allow the network to assimilate context information, several frames of data are passed through the network before the probabilities for the first frame are read off. An interval of two frames has been found to be most satisfactory.

Training the network requires 'unfolding' it in time. During training on a word, the inputs, outputs and feedback activations are stored for each frame. At the end of a word, the errors are propagated back using the generalised delta rule [65], treating the network at different times as different layers of a multi-layer network (figure 7b). One problem with network training is to know when to finish training — too much training can cause the network to over-specialise and to adapt too closely to the particular features of the training set. This would impair generalisation to unseen test data, but if training is stopped too early, the optimum performance will not be achieved. This problem is solved by creating a third data set which is used for validation. After training the network for a short time, the network's performance is tested on the validation set. This train and validate cycle is repeated until the recognition rate on the validation set starts to deteriorate, indicating that the network is starting to become overtrained. At this point training is stopped and the network is tested on the test data which has still not been put through the network.

It is widely recognised that the back-propagation algorithm (introduced as the modified delta rule in Rumelhart, Hinton & Williams [65]) used to train error-propagation networks can be improved in a variety of ways, to speed up convergence and to make convergence to a good local minimum more likely. In addition to the incorporation of a momentum term in the weight update formulae, two such improvements have been used in this work, namely Jacobs' delta-bar-delta update rule [28] and Bridle's Softmax [5]. The former provides for individual learning rates for each weight and the latter a different transfer function on the output units of the network, ensuring that the outputs are all probabilities (between 0 and 1 and summing to 1). Because of difficulties in training stability, modifications suggested by Robinson [64] were incorporated and gave much improved convergence. These changes use multiplicative learning rate changes and prevent the learning rates from deviating too far from the mean. For this work the geometric mean was used, and the additional measure of zeroing momentum terms when the network energy increased was taken.

Prior probabilities

In any corpus of text, the occurrence frequencies $f(\mathcal{L})$ of different letters $\mathcal{L}$ will vary. In a typical section of text, the letter *'e'* may outnumber the letter *'q'* by a factor of several hundred. This discrepancy will severely distort the network's behaviour, since it will rarely be presented with *'q'*s



(a) Schematic

(b) Net unfolded for training

Figure 7: The recurrent error propagation network

and these will be swamped. The network trained on such data will recognise nearly anything as an 'e' and is unlikely to recognise anything as a 'q'. To overcome this problem, two methods are available, both of which have been tested.

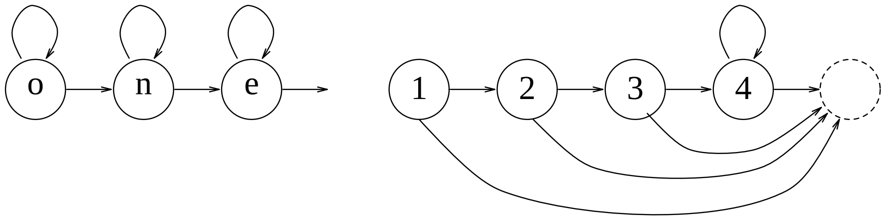
The first method attempts to remove these biases in the training. In the backpropagation algorithm, the objective function or energy E_p is defined by $E_p = \frac{1}{2}\sum_j (o_{pj} - t_{pj})^2$ for pattern p in class $\mathcal{L}(p)$, with outputs o_{pj} and target values t_{pj}. This determines the error signal propagated back through the network. If this error function is biased so that more weight is given to rare letters, all letters will contribute roughly the same amount to the error signal. The new objective function used is:

$$E_p = \frac{1}{2}\frac{\sum_j (o_{pj} - t_{pj})^2}{f(\mathcal{L}(p))} \tag{1}$$

The second method, advocated by Makhoul [41] and Renals, Morgan & Bourlard [61,62], is to train the network with the data available, but divide the output probabilities by the prior probability of the class ignoring the data. This gives a measure of the relative probability of the class given the data which is a better measure to use in the post-processing. In initial trials this method did not work as well as the first, which has been used for all experiments presented here.

3.5 Post-processing

The output of the network is in the form of probability estimates for each of the 27 letter classes in each of the frames. From these data, the word that gave rise to them must be determined. This is achieved using Dynamic Programming [68] in the form of the Viterbi algorithm. A Markov model [58] is created for each word, with one state per letter, each state containing the probability that the data so far were generated by the letters in that model. For every new frame, each state's probability is multiplied by the network's estimate of the probability of the letter to which that state corresponds. Transitions (transferring one state's probability to another state) are allowed only from one state to itself or to the next state. Figure 8a shows the model for the word 'one' with two arrows from each state marking the permitted transitions. After the last frame of data has been processed, the probability of the final state of any word model is the probability that the observed data were produced by that model. By choosing the maximum of these likelihood estimators, if the models are good, a good estimate of the identity of the original word is obtained. All of these probabilities are stored and multiplied in the log domain for numerical accuracy. In fact this will increase speed too because these log probabilities can be stored as integers which are added to carry out probability multiplication, and the Softmax formulation means that the network outputs are in any case exponentials.



(a) A simple Markov model (b) A complex durational model for one letter

Figure 8: Markov models

Durational modelling

If the duration statistics (i.e. the probability of a state lasting a given number of frames) of the Markov models are examined, an exponential decay can be seen (the solid line of figure 9). This is far from the bell-shaped curve that one would expect to see for the actual durations of letters observed in the database. This discrepancy points to inaccuracy in the modelling and indicates how the performance of the recogniser might be improved. Ramesh & Wilpon [59] present one attempt to overcome this problem in the modelling of speech using hidden Markov models.

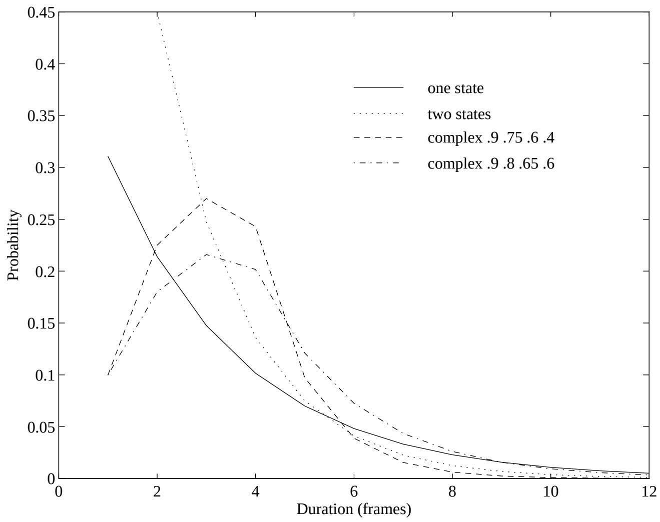


Figure 9: Durational modelling of different Markov models.

A simple solution is to force a minimum duration greater than one. This can be achieved by duplicating each state in the word model which makes each letter last at least two frames. This is seen to improve the performance, but cannot pretend to model the data accurately (the dotted line of figure 9.) Note that both curves are selected to have a mean of 3.22 frames — the observed mean letter length in the database.

A more complex durational model can be used, such as that shown in figure 8b, where each of the original states is replaced with four, with transitions as depicted by the arrows. By choosing the probability of each transition, the probability distribution of the number of frames spent in the model can be made to conform quite closely to that observed in practice, as shown in the dashed curves of figure 9. These two bell-shaped curves generated by four-state complex Markov models are much closer to the histogram of observed letter lengths.

3.6 Results

The system performance is judged by the number of words in the test set which are correctly identified, and these are quoted as a percentage of the 124 test words used. All results below are for networks with 40 feedback units trained using the cross-validation technique described in section 3.4. The results shown are averaged across four test runs, and are presented with standard deviations to indicate the sensitivity to initial conditions. The four test runs were identical except for the seed used by the random number generator which randomises the initial weights of the network to break symmetry. Experiments under other conditions used the same initial weights (the same seeds) so paired data analysis can be conducted to assess performance improvement. The first two columns of results are the means and sample standard deviations for models trained with equal-length segmentation, and the next two show the results after the network has been retrained using its own segmentation estimates.

Networks trained using one type (one, two or four state) of Markov model ('MM') have also been tested with other models. From the significantly higher recognition rates when testing on

Method	Equal		Auto	
	Mean (%)	σ_{n-1}	Mean (%)	σ_{n-1}
4 state MM	80.2	2.2	79.2	2.2
4 state MM (New preprocessing)	84.3	3.0	85.5	3.8
Tested on training data	*96.5*	*1.8*	*98.3*	*1.4*
Tested with 2 state MM	83.4	1.5	84.0	4.2
Tested with 1 state MM	83.7	3.3	82.6	4.4
2 state MM	83.4	4.2	81.6	2.7
Tested with 4 state MM	84.8	3.5	82.4	2.3

Table 3: Percentage recognition rates for networks with 40 feedback units.

the training set compared to the test set, it can be seen that a larger training set capturing more variation would improve the performance.

Using paired statistics, the significance of any improvements can be calculated using Student's t-test. The newer preprocessing method which incorporates the feature spotting mentioned in section 3.3 is better than the original. ($t_7 = 3.84$ gives a significance greater than 99%.) The change in performance by retraining using automatic segmentation is not always positive and in the case of the best results is not significant. Training using the 2 state Markov models is worse than that using 4 state MMs, to 95% significance ($t_7 = 2.0$). In fact, the network trained with 2 state models performs better when tested with the 4 state models ($t_7 = 2.4$ i.e. 97.5% significance.)

Though work so far has concentrated on the numbers corpus, the system described above has been tested on a small LOB-based database. Initial results (on a 40 unit network) are encouraging (62.1%) but a larger network and more training data could be expected to give better results.

3.7 Conclusions

The above results show that the method of recurrent error propagation networks can be applied with some success to the task of off-line cursive script recognition. Comparison of results with other researchers is difficult because of differences in experimental details, the actual handwriting used and the method of data collection. The results which have been published for similar problems are noted in section 2.3.

These experiments show that an 85.5% recognition rate can be achieved on the task attempted, and that the enhancements in preprocessing and durational modelling have significantly improved this performance. Further improvement can be expected with a future increase in the training set size. Improvements might also be achieved by finding a more robust, more detailed preprocessing technique.

This report has only addressed the problem of small vocabulary, single-writer handwriting recognition, so has limited applicability. Future work will continue to aim for higher recognition rates on this task, but must also begin to look at large vocabulary or writer independent testing.

Acknowledgements

The author would like to extend his thanks to the following for help in producing this report: F. Fallside, T.T. Jervis, A.J. Robinson, G. Wong.

References

[1] Aldus & Microsoft. *Tiff Standard Definition*, 5.0 edition, August 1988.

[2] Elie Bienenstock. Visual pattern processing using a neural-network based approach. In *Cambridge University Summer School on Neural Networks*, April 1991.

[3] H. Bouma. Visual recognition of isolated lower case letters. *Vision Research*, 11:459–474, 1971.

[4] R.M. Božinović and S.N. Srihari. Off-line cursive word recognition. *IEEE PAMI*, 11(1):68–83, January 1989.

[5] John S. Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. *Neurocomputing*, F 68:227–236, 1990.

[6] E.R. Brocklehurst, D.M. Ford, and Hilary J. Symm. Feature extraction for cursive script recognition. Technical report, National Physical Laboratory, November 1988.

[7] E.R. Brocklehurst and P.D. Kenward. Preprocessing for cursive script recognition. Technical report, National Physical Laboratory, November 1988.

[8] M.K. Brown. *Cursive Word Script Recognition.* PhD thesis, University of Michigan, 1981.

[9] John Browning. Artificial intelligence survey. *The Economist*, 322(7750):21, March 1992.

[10] D.J. Burr. Designing a handwriting reader. *IEEE PAMI*, 5(5):554–559, September 1983.

[11] E.R. Davies. *Machine Vision: Theory, algorithms, practicalities.* Microelectrics and signal processing Number 9. London Academic, 1990.

[12] P. de Bruyne and R. Korolnik. Segmentation and recognition of calligraphic text. In *IEE 2nd conference on Neural Networks*, number 349, pages 214 –218, November 1991.

[13] S. Edelman, S. Ullman, and T. Flash. Reading cursive script by alignment of letter prototypes. *International Journal of Computer Vision*, 5(3):303–331, 1990.

[14] D.G. Elliman and R.N. Banks. A comparison of two neural networks for hand-printed character recogition. In *IEE 2nd Neural Networks*, number 349 in IEE, pages 224–228, November 1991.

[15] Raouf F.H. Farag. Word level recognition of cursive script. *IEEE Transactions on Computers*, C-28(2):172–175, February 1979.

[16] Thomas Fontaine and Lokendra Shastri. Character recognition using a modular spatiotemporal connectionist model. Neuroprose, University of Pennsylvania, Philadelphia PA 19104-6389, 1992.

[17] Herbert Freeman. On the digital computer classification of geometric line patterns. *Proceedings of the National Electronics Conference*, 18:312–324, 1962.

[18] Kunihiko Fukushima. Neocognitron: a self-organising neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202, 1980.

[19] Elisabeth Geake. Letters to a computer. *New Scientist*, pages 30–33, 27 June 1992.

[20] Rafael C. Gonzalez and Paul A. Wintz. *Digital Image Processing.* Addison Wesley, Reading, Mass., 1977.

[21] V.K. Govindan and A.P. Shivaprasad. Character recognition – a review. *Pattern Recognition*, 23(7):671–683, 1990.

[22] I. Guyon, P. Albrecht, Y. LeCun, J. Denker, and W. Hubbard. A time delay neural network character recogniser for a touch terminal. In *INNC Paris '90*, 1990.

[23] Ralph Norman Haber and Lyn R. Haber. Visual components of the reading process. *Visible Language*, XV(2):147–182, 1981.

[24] Daniel J. Hepp. An application of backpropagation to the recognition of handwritten digits using morphologically derived features. *Proc SPIE*, 1451:228–233, February 1991.

[25] Geoffrey E. Hinton, Christopher K.I. Williams, and Michael D. Revow. Adaptave elastic models for hand-printed character recognition. In *Advances in NIPS*, volume 4, 1992.

[26] Yizhak Idan and Raymond C. Chevalier. Handwritten digits recognition by a supervised Kohonen-like learning algorithm. *ICJNN 91*, 3:2576–2581, December 1991.

[27] S. Impedevo, G. Dimauro, and G. Pirlo. A new decision tree algorithm for handwritten numerals recognition using topological features. *Proc SPIE*, 1384:280–284, November 1990.

[28] Robert A. Jacobs. Increased rates of convergence through learning rate adaptation. *Neural Networks*, 1:295–307, 1988.

[29] Kazuki Joe, Yoshihiro Mori, and Sei Miyake. Construction of a large scale neural network. *Concurrency: Practice and Experience*, 2(2):79–107, June 1990.

[30] S. Johansson, E. Atwell, R. Garside, and G. Leech. The tagged LOB corpus. Technical report, Norwegian Computing Centre for The Humanities, Bergen, 1986.

[31] Maha Kadirkamanathan. *A Scale-Space Approach to Segmentation and Recognition of Cursive Script.* PhD thesis, Cambridge University Engineering Department, 1989.

[32] F. Kimura and M. Shridhar. Handwritten numerical recognition based on multiple algorithms. *Pattern Recgnition*, 24(10):969–983, 1991.

[33] T. Kohonen. *Self Organisation and Associative Memory.* Springer-Verlag, 1984.

[34] Roland Kuhn and Renato de Mori. A cache-based natural language model for speech recognition. *IEEE PAMI*, 12(6):570–583, June 1990.

[35] K.J. Lang and G.E. Hinton. A time delay neural network for speech recognition. Technical Report CMU-CS-88-152, CMU, 1988.

[36] Eric Lecolinet and Jean-Pierre Crettez. A grapheme-based segmentation technique. In *International Conference on Document Analysis and Recognition*, volume 2, pages 740–748, 1991.

[37] Y. LeCun, B. Boser, J.S. Denker, D. henderson, R.E. Howard, W. Hubbard, and L.D. Jackel. Backpropagation applied to handwritten zip code recognition. *Journal of Neural Computation*, 1:541–551, 1989.

[38] M. Leroux, J.C. Salome, and J. Badard. Recognition of cursive script words in a small lexicon. In *International Conference on Document Analysis and Recognition*, volume 2, pages 774–782, 1991.

[39] Laurence Likforman-Sulem, Henri Maitre, and Colette Sirat. An expert vision system for analysis of hebrew characters and authentication of hebrew manuscripts. *Pattern Recognition*, 24(2):121–137, 1991.

[40] Si Wei Lu, Ying Ren, and Ching Y. Suen. Hierarchical attributed graph representation and recognition of handwritten chinese characters. *Pattern Recognition*, 24(7):617–632, 1991.

[41] John Makhoul. Pattern recognition properties of neural networks. In *Neural Networks for Signal Processing - Proceedings of the IEEE workshop*, pages 173–187. IEEE, Piscataway, NJ, 1991.

[42] David Marr. *Vision.* Freeman, San Francisco, 1982.

[43] W. Martins and N.M. Allinson. Visual search of postal codes by neural networks using human examples. In *IEE 2nd Conference on Neural Networks*, 1991.

[44] Ofer Matan, Henry S. baird, Jane Bromley, Christopher J.C. Burges, John S. Denker, Lawrence D. Jackel, Yann LeCun, Edwin P.D. Pednault, William D. Satterfield, Charles E. Stenard, and Timothy J. Thompson. Reading handwritten digits: A zip code recognition system. *IEEE Computer*, 25(7):59–62, July 1992.

[45] P. Morasso. Self-organisation of an allographic lexicon. In *IJCNN 1989*, volume 1, pages 141–144, June 1989.

[46] P. Morasso, F. Kennedy, E. Antonj, S. DiMarco, and M. Dordoni. Self-organisation of an allographic lexicon. In *International Neural Network Conference Paris*, volume 1, pages 141–144, July 1990.

[47] Jean-Vincent Moreau, Brigitte Plessis, Olivier Bougeois, and Jean-Luc Plagnaud. A postal cheque reading system. In *International Conference on Document Analysis and Recognition*, volume 2, pages 758–766, 1991.

[48] Shunji Mori, Ching Y. Suen, and Kazuhiko Yamamoto. Historical review of OCR research and development. *Proceedings of the IEEE*, 80(7):1029–1058, July 1992.

[49] Yoshihiro Mori and Kazuki Joe. A large scale neural network which recognizes handwritten kanji characters. *Advances in NIPS*, 2:415–422, 1989.

[50] Yoshihiro Mori and Kazuhiko Yokosawa. Neural networks that learn to discriminate similar kanji characters. *Neural Information Processing Systems*, 1, 1988.

[51] R. Nag, K.H. Wong, and F. Fallside. Handwritten script recognition using hidden Markov models. In *ICASSP '86*. IEEE, 1986.

[52] J. Nellis and T.J. Stonham. A fully integrated hand-printed character recognition system using artificial neural networks. In *IEE 2nd Conference on Neural Networks*, number 349 in IEE, pages 219–223, November 1991.

[53] Paul W. Palumbo, Sargur N. Srihari, Jung Soh, Ramalingam Sridhar, and Victor Demajenko. Postal address block location in real time. *IEEE Computer*, 25(7):34–42, July 1992.

[54] T. Paquet and Y. Lecourtier. Handwriting recognition: Application on bank cheques. In *International Conference on Document Analysis and Recognition*, volume 2, pages 749–757, 1991.

[55] Barak A. Pearlmutter. Dynamic recurrent neural networks. Technical Report CMU-CS-88-191, CMU, School of Computer Science, Pittsburgh, PA15213, December 1990.

[56] J.C. Pettier and J. Camillerapp. An optimal detector to localize handwriting strokes. In *International Conference on Document Analysis and Recognition*, volume 2, pages 710–718, 1991.

[57] Réjean Plamondon and Guy Lorette. Automatic signiature verification and writer identification – the state of the art. *Pattern Recognition*, 22(2):107–129, 1989.

[58] L.R. Rabiner and B.H. Juang. An introduction to hidden Markov models. *IEEE ASSP magazine*, 3(1):4–16, January 1986.

[59] Padma Ramesh and Jay G. Wilpon. Modelling state durations in hidden Markov models for automatic speech recognition. In *ICASSP 92*, volume 1, pages 381–384, 1992.

[60] Keith Rayner and Alexander Pollatsek. *The Psychology of Reading*. Prentice-Hall, 1989.

[61] Steve Renals, Nelson Morgan, and Hervé Bourlard. Probability estimation by feed-forward networks in continuous speech recognition. In *IEEE Workshop on Neural Networks for Signal Processing*. IEEE, IEEE, October 1991.

[62] Steve Renals, Nelson Morgan, Hervé Bourlard, Michael Cohen, Horacio Franco, Chuck Wooters, and Phil Kohn. Connectionist speech recognition: Status and prospects. Technical Report TR-91-070, ICSI, December 1991.

[63] A.J. Robinson. *Dynamic Error Propagation Networks.* PhD thesis, Cambridge University Engineering Department, February 1989.

[64] Tony Robinson and Frank Fallside. A recurrent error propagation network speech recognition system. *Computer Speech and Language*, 5:259–274, 1991.

[65] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. In D.E. Rumelhart and J.L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1, chapter 8, pages 318–362. Bradford Books, 1986.

[66] S. Sasanuma. Can surface dyslexia occur in japanese. In *Orthographies and Reading.* Lawrence Erlbaum Associates, 1984.

[67] A.W. Senior and F. Fallside. Off-line handwriting recognition by recurrent error propagation networks. In David Hogg and Roger Boyle, editors, *British Machine Vision Conference 1992.* Springer Verlag, September 1992.

[68] H.F. Silverman and D.P. Morgan. The application of dynamic programming to connected speech recognition. *IEEE ASSP Magazine*, pages 6–25, July 1990.

[69] Jean-Claude Simon. Off-line cursive word recognition. *Proceedings of the IEEE*, 80(7):1150–1161, July 1992.

[70] Sargur N. Srihari and Radmilo M. Božinović. A multi-level perception approach to reading cursive script. *Artificial Intelligence*, 33:217–255, 1987.

[71] Ching Y. Suen, Marc Berthod, and Shunji Mori. Automatic recognition of handprinted characters— the state of the art. *Proc. IEEE*, 68(4):469–487, April 1980. Contains 244 references.

[72] I. Taylor and M.M. Taylor. *The Psychology of Reading.* New York Academic, 1983.

[73] J.M. Tazelaar. Recognizing script. *Byte*, 16(4):210, April 1991.

[74] Arnold J.W.M. Thomassen, Hans-Leo Teulings, Lambert R.B. Schomaker, Pietro Morasso, and Jonathan Kennedy. Towards the implementation of cursive script understanding in an on-line handwriting recognition system. In *Esprit '88: Putting the Technology to Use*, pages 628–639. North Holland, 1988. Project 419.

[75] M. van Braekel and P. Daenens. The infrared determination of writing sequence of ball-point pen strokes. In *10th Meeting fo the International Association of Forensic Sciences*, volume 24/4, page 450, July/August 1984.

[76] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang. Phoneme recognition using time delay neural networks. *IEEE Acoustics, Speech and Signal Processing*, 37(3):328 –339, March 1989.

[77] Ching-Huei Wang and Sargur N. Srihari. A framework for object recognition in a visually complex environment and its applicaion to locating address blocks on mail pieces. *International Journal of Computer Vision*, 2:125–151, 1988.

[78] P.T. Wright. On-line recognition of handwriting. *GEC Journal of Research*, 8(1):42–48, 1990.

[79] Atsushi Yamadori. Ideogram reading in alexia. *Brain*, 98:231–238, 1975.