Pattern Anal Applic (2003) 6: 97–121
DOI 10.1007/s10044-002-0169-3

ORIGINAL ARTICLE

A. L. Koerich, R. Sabourin, C. Y. Suen

Large vocabulary off-line handwriting recognition: A survey

Received: 24/09/01 / Accepted: 15/03/02

Abstract Considerable progress has been made in handwriting recognition technology over the last few years. Thus far, handwriting recognition systems have been limited to small and medium vocabulary applications, since most of them often rely on a lexicon during the recognition process. The capability of dealing with large lexicons, however, opens up many more applications. This article will discuss the methods and principles that have been proposed to handle large vocabularies and identify the key issues affecting their future deployment. To illustrate some of the points raised, a large vocabulary off-line handwritten word recognition system will be described.

Keywords Handwriting recognition · Large vocabulary · Lexicon reduction · Open vocabulary · Search techniques

Introduction

Handwriting recognition technology is steadily growing toward its maturity. Significant results have been achieved in the past few years both in on-line [1–12] and off-line [13–22] handwriting recognition. While generic content text recognition seems to be a long-term goal [19,21,23–25], some less ambitious tasks are currently being investigated that address relevant problems such as the recognition of postal addresses [15,16,26–29], and the legal amount on bank cheques [30–38]. Current systems are capable of transcribing handwriting with average recognition rates of 90–99%, depending on the constraints imposed (e.g. size of the vocabulary, writer-dependence, writing style, etc.), and also on the experimental conditions [15,16,21,31,35]. The recognition rates reported are much higher for on-line systems when considering the same constraints and experimental conditions [4,7,9,39].

One of the most common constraints of current recognition systems is that they are only capable of recognising words that are present in a restricted vocabulary, typically comprised of 10–1000 words [14–16,18,20,21]. The restricted vocabulary, usually called a *lexicon*, is a list of all valid words that are expected to be recognised by the system. There are no established definitions, however, the following terms are usually used:

- small vocabulary – tens of words;
- medium vocabulary – hundreds of words;
- large vocabulary – thousands of words;
- very large vocabulary – tens of thousands of words.

The lexicon is a key point to the success of such recognition systems, because it is a source of linguistic knowledge that helps to disambiguate single characters by looking at the entire context [17,40,41]. As the number of words in the lexicon grows, the more difficult the recognition task becomes, because more similar words are more likely to be present in the lexicon. The computational complexity is also related to the lexicon, and it increases relatively to its size. Some open vocabulary systems, i.e. systems that are capable of dealing with any word presented at the input without relying on a lexicon, have also been proposed [42,43], but their accuracy is still far below those relying on limited lexicons [43,44]. However, most of the research efforts in the field have been devoted to improving the accuracy of constrained systems, notably small vocabulary systems, without giving much attention to the computational complexity or recognition speed.

We can classify the field of handwriting recognition in several ways. However, the most straightforward one is to distinguish between on-line (also called dynamic) and off-line (also called static) handwriting recognition. The former profits from information on the time order and dynamics of the writing process that is captured by the writing device [9,39]. The temporal information is an additional source of knowledge that helps to increase the

A. L. Koerich (✉) · R. Sabourin
Departement de Génie de la Production Automatisée, École de Technologie Supérieure (ETS), Université du Québec, Montréal, QC, Canada.
E-mail:alexoe@computer.org

C. Y. Suen
Centre for Pattern Recognition and Machine Intelligence (CENPARMI), Concordia University, Montréal, QC, Canada

recognition accuracy. On the other hand, off-line systems need to rely on more sophisticated architectures to accomplish the same recognition task, but the results are still below those obtained by on-line recognition systems under similar testing conditions [9,39]. Due to the availability of this additional source of knowledge, on-line systems can use simpler models, similar to those used in speech recognition. So, more often we find on-line recognition systems that deal with large vocabularies [4,6,11,20,45,46]. However, even in on-line handwriting recognition, the size of the vocabulary poses a serious challenge, and researchers avoid dealing directly with a large number of words [3,10]. This article focuses mainly on off-line handwriting recognition, but some relevant works in large vocabulary on-line handwriting recognition that make use of strategies which might be extended to off-line problems will also appear throughout the sections.

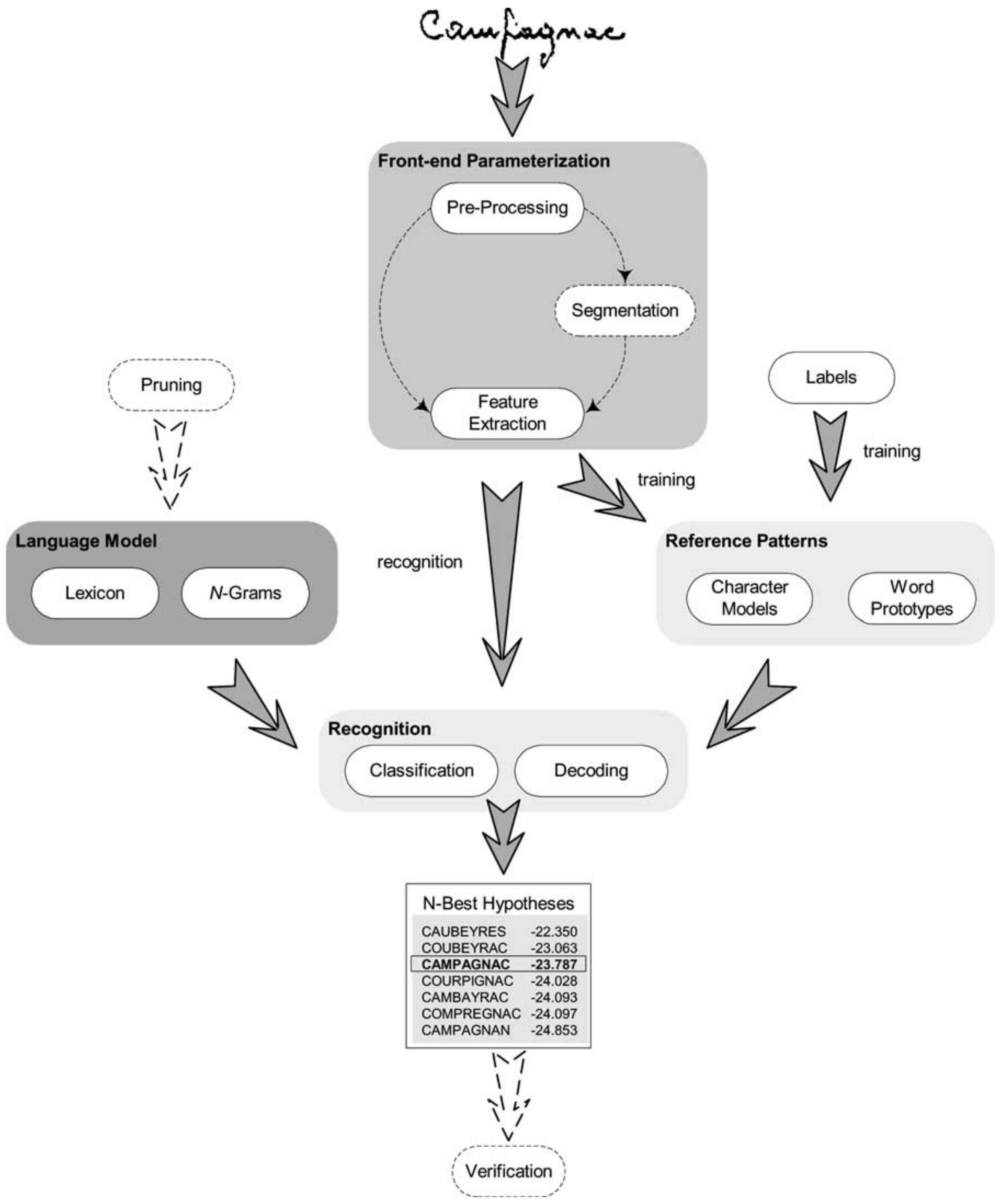

Fig. 1. Overview of the basic modules of an off-line handwriting recognition system

A paradigm for handwriting recognition

A wide variety of techniques are used to perform handwriting recognition. A general model for handwriting recognition, shown in Fig. 1, is used throughout this article to highlight the many aspects of the use of large vocabularies in handwriting recognition. The model begins with an unknown handwritten word that is presented at the input of the recognition system as an image. To convert this image into information understandable by computers requires the solution to a number of challenging problems. Firstly, a front-end parameterisation is needed which extracts from the image all of the necessary meaningful information in a compact form compatible with the computer language. This involves the pre-processing of the image to reduce some undesirable variability that only contributes to complicate the recognition process. Operations like slant and slope correction, smoothing, normalisation, etc. are carried out at this stage. The second step in the front-end parameterisation is the segmentation of the word into a sequence of basic recognition units such as characters, pseudo-characters or graphemes. However, segmentation may not be present in all systems. Some approaches treat words as single entities, and attempt to recognise them as a whole [47–49]. The final step is to extract discriminant features from the input pattern to either build up a feature vector or to generate graphs, string of codes or sequence of symbols whose class is unknown. However, the characteristics of the features depend upon the preceding step, say whether or not segmentation of words into characters was carried out.

The pattern recognition paradigm to handwriting recognition consists of pattern training, i.e. one or more patterns corresponding to handwritten words of the same known class are used to create a pattern representative of the features of that class. The resulting pattern, generally called a *reference pattern* or *class prototype*, can be an exemplar or template, derived from some type of averaging technique, or it can be a model that characterises the statistics of the features of the reference pattern. In spite of the goal of most recognition systems being to recognise words, sometimes it is difficult to associate one class to each word, so then sub-word models (e.g. characters, pseudo-characters and grapheme models) are trained instead, and standard concatenation techniques are used to build up word models during the recognition.

The recognition includes a comparison of the unknown test pattern with each class reference pattern, and measuring a similarity score (e.g. distance, probability) between the test pattern and each reference pattern. The pattern similarity scores are used to decide which reference pattern best matches the unknown pattern. Recognition can be achieved by many methods, such as Dynamic Programming (DP) [47,50], Hidden Markov Modelling (HMM) [14,16,51], Neural Networks (NN) [52], *k* Nearest Neighbour (kNN) [53], expert systems [48,54] and combinations of techniques [31,35]. The recognition process usually provides a list of best word hypotheses. Such a list can be post-processed or verified to obtain a more reliable list of word hypotheses [55,56]. The post-processing or verification may also include some rejection mechanism to discard unlikely hypotheses.

However, for meaningful improvements in recognition, it is necessary to incorporate into the recognition process other sources of knowledge, such as language models. A lexicon representing the recognition vocabulary (i.e. the words that are expected (allowed) at the input of the recognition system) is the most commonly used source of knowledge. Notably, a limited vocabulary is one of the most important aspects of systems that rely on large vocabularies, because it contributes to improving the accuracy as well as reducing computation. In the case of systems that deal with large vocabularies, other additional modules may be included, such as pruning or lexicon reduction mechanisms.

Although the above description is not a standard, it is typical of most modern recognition systems. Many of the issues related to the basic modules are common to small and medium vocabularies, and they are overlooked in this survey. We recommend that the interested reader looks at other references covering these subjects in more detail [12,57–62]. This survey will focus on the most relevant aspects of large vocabulary handwriting recognition systems.

Segmentation of words into characters

The most natural unit of handwriting is the word, and it has been used for many recognition systems [19,32,49]. One of the greatest advantages of using whole word models is that these are able to capture within-word co-articulation effects [49]. When whole word models are adequately trained, they will usually yield the best recognition performance. Global or holistic approaches treat words as single, indivisible entities, and attempt to recognise them as whole, bypassing the segmentation stage [48,49]. Therefore, for small vocabulary recognition, such as the case of bank cheque applications where the lexicons do not have more than 30–40 entries [35], whole word models are the preferred choice.

Nevertheless, many practical applications require larger vocabularies with hundreds or thousands of words [4,19,63]. While words are suitable units for recognition, they are not a practical choice for large vocabulary handwriting recognition. Since each word has to be treated individually and data cannot be shared between word models, this implies a prohibitively large amount of training data. In addition, the recognition vocabulary may consist of words that have not appeared in the training procedure. Instead of using whole word models, analytical approaches use sub-word units, such as characters or pseudo-characters as the basic recognition units, requiring the segmentation of words into these units. Even with the difficulty and the errors introduced by the segmentation stage, most successful approaches are segmentation-recognition methods in which words are first loosely segmented into characters or pieces of characters, and dynamic programming techniques with a lexicon are used in recognition to choose the definitive segmentation as well as to find the best word hypotheses. As a result, the analytical approach is the preferred choice for applications where large vocabularies are required.

Recognition strategies

A key question in handwriting recognition is how test and reference patterns are compared to determine their similarity. Depending on the specifics of the recognition system, pattern comparison can be done in a wide variety of ways. The goal of a classifier is to match a sequence of observations derived from an unknown handwritten word against the reference patterns that were previously trained, and to obtain confidence scores (distance, cost or probabilities) to further decide which model best represents the test pattern. Here, we have to distinguish between word models and sub-word models. As we have pointed out, approaches that use sub-word models are more suitable for large vocabulary applications. So, we thus assume that the reference patterns are related to sub-word units or characters.

An observation sequence can be represented by different ways: by low-level features, such as smoothed traces of the word contour, stroke direction distributions, pieces of strokes between anchor points, local shape templates, etc. [64,65]; by medium-level features that aggregate low-level features to serve as primitives include edges, end-points, concavities, diagonal and horizontal strokes, etc. [32,47]; or by high-level features such as ascenders, descenders, loops, dots, holes, t-bars, etc. Moreover, such features can be used in different ways to build up feature vectors, graphs, string of codes or a sequence of symbols. Here, it is convenient to distinguish between two particular representations of the test pattern: as a sequence of observations, or as a sequence of primitive segments. We define a test pattern O as a sequence of observations such that $O = (o_1\ o_2 \dots o_T)$ in which T is the number of observations in the sequence and o_t represents the tth symbol. We define S as a sequence of primitive segments of the image such that $S = \{s_1, s_2, \dots, s_P\}$, in which P is the number of segments in the sequence and s_p represents the pth primitive. In a similar manner, we define a set of reference patterns $\mathfrak{R}\{R_1, R_2, \dots, R_V\}$, where each reference pattern, R_v, represents a word that is formed by the concatenation of sub-word units (characters), such that $R_v = (c_1^v c_2^v \dots c_L^v)$ in which L is the total number of sub-word units that form a word, and c_l^v represents the lth sub-word unit. The goal of the pattern comparison stage is to determine a similarity score (cost, distance, probability, etc.) of O or S to each R_v, $1 \leq v \leq V$, to identify the reference pattern that gives the best score, and to associate the input pattern with this reference pattern. Since words are broken up into sub-word units, the recognition strategies used are essentially based on dynamic programming methods that attempt to match primitives or blocks of primitives with sub-word units to recognise words. Depending upon how the words are represented, statistical classification techniques, heuristic matching techniques, symbolic matching methods or graph matching methods are some of the possible matching methods that can be used [49]. In word recognition, the optimal interpretation of a word image may be constructed by concatenating the optimal interpretation of the disjoint parts of the word image.

In terms of the optimal path problem, the objective of the DP methods is to find the optimal sequence of a fixed number of moves, say L, starting from point i and ending at point j, and the associated minimum cost $\varphi_L(i, j)$. The P points representing the sequence of P primitives are plotted horizontally, and the L points representing the sub-word models (or the moves) are plotted vertically (Fig. 2). The Bellman's principle of optimality [66] is applied in this case, and after having matched the first l moves, the path can end up at any point k, $k = 1, 2, \dots, P$, with the associated minimum cost $\varphi_l(i, k)$. The optimal step, associating the first $l + 1$ characters with the first p primitives, is given as:

$$\varphi_{l+1}(i, p) = \min_k \left[\varphi_l(i, k) + \zeta(k, p)\right] \tag{1}$$

where $\varphi(\cdot)$ is the minimum cost (or best path) and $\zeta(\cdot)$ represents the cost to associate the $l + 1$th character to the aggregation composed by primitives $i + 1$, $i + 2$, $\dots$, p.

Besides the matching strategy, the segmentation-based methods used in large vocabulary handwriting recognition lie within two categories:

- Character recognition followed by word decoding – characters or pseudo-characters are the basic recognition units, and they are modelled and classified independently of the words, i.e. the computation of the cost function is replaced by an ordinary Optical Character Recogniser (OCR) that outputs the most likely character and its confidence level given a primitive or a block of primitives. To this aim, pattern recognition approaches such as template matching, structural

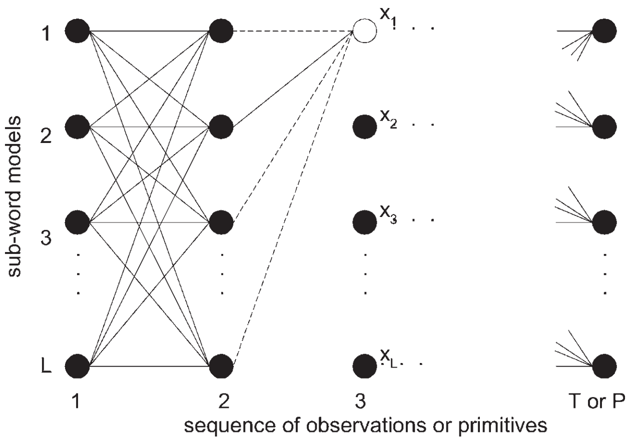

Fig. 2. Trellis structure that illustrates the problem of finding the best matching between a sequence of observations O (or sequence of primitives S) and a reference pattern $R_v = (c_1 c_2 \cdots c_L)$

techniques, neural networks and statistical techniques [67] can be used. Further, character scores are matched with lexicon entries by dynamic programming methods [52,53].

- Character recognition integrated with word decoding – characters or pseudo-characters are the basic recognition units, and they are concatenated to build up word models according to the lexicon. The classification is carried out by dynamic programming methods that evaluate the best match between the whole sequence of observations and word models [14,16,51].

A simplified dynamic programming approach rely on minimum edit-distance classifiers (usually using the Levenshtein's metric) that attempt to find a reference pattern R_v that has the minimum cost with respect to the input pattern O (or S) [61] as:

$$d(O, R_v) = \min \begin{cases} d(o_1 \dots o_{T-1}, c_1^v \dots c_{L-1}^v) + sub(o_T, c_L^v) \\ d(o_1 \dots o_{T-1}, c_1^v \dots c_L^v) + ins(o_T) \\ d(o_1 \dots o_T, c_1^v \dots c_{L-1}^v) + del(c_L^v) \end{cases} \tag{2}$$

where $d(O, R_v)$ is the minimum distance between O and R_v, $del(c_L^v)$, $sub(o_T, c_L^v)$ and $ins(o_T)$ are the cost parameters for deletion, substitution and insertion, respectively.

So far, handwriting recognition using Neural Networks (NN) has mostly been aimed at digit recognition [68,69], isolated character recognition [52] and small vocabulary word recognition [70], because in large vocabulary handwriting recognition, words must be segmented before neural network modelling [71]. With large vocabularies, NNs are not frequently used as front-end classifiers, but as part of hybrid approaches, where they are used to estimate *a priori* class probabilities [21,35,36], *a priori* grapheme probabilities [72] or to verify results of previous classifiers (as a back end classifier) [55].

Statistical techniques use concepts from statistical decision theory to establish decision boundaries between pattern classes [67]. Techniques such as the k nearest neighbour decision rule [53], Bayes decision rule, support vector machines [73], and clustering [74] have been used in handwriting recognition, but mostly aimed at the recognition of isolated characters and digits or words in small vocabularies. However, during the last decade, Hidden Markov Models (HMMs), which can be thought of as a generalisation of dynamic programming techniques, have become the predominant approach to automatic speech recognition [75]. The HMM is a parametric modelling technique, in contrast with the non-parametric DP algorithm. The power of the HMM lies in the fact that the parameters that are used to model the handwriting signal can be well optimised, and this results in lower computational complexity in the decoding procedure, as well as improved recognition accuracy. Furthermore, other knowledge sources can also be represented with the same structure, which is one of the important advantages of Hidden Markov Modelling [76].

The success of HMMs in speech recognition has led many researchers to apply them to handwriting recognition by representing each word image as a sequence of observations. The standard approach is to assume a simple probabilistic model of handwriting production, whereby a specified word w produces an observation sequence O with probability $P(w, O)$. The goal is then to decode the word, based on the observation sequence, so that the decoded word has the maximum *a posteriori* (MAP) probability, i.e.

$$\hat{w} \ni P(\hat{w}|O) = \max_{w \in \mathcal{R}} P(w|O) \tag{3}$$

The way we compute $P(w|O)$ for large vocabularies is to build statistical models for sub-word units (characters) into an HMM framework, build up word models from these subword models using a lexicon to describe the composition of words, and then evaluate the model probabilities via standard concatenation methods and DP-based methods such as the Viterbi algorithm [15,16]. This procedure is used to decode each word in the lexicon.

In fact, the problem of large vocabulary handwriting recognition is turned into an optimisation problem that consists of evaluating all the possible solutions and choosing the best one, that is, the solution that is optimal under certain criteria. The main problem is that the number of possible hypotheses grows as a function of the lexicon size and the number of sub-word units, and that imposes formidable computation requirements on the implementation of search algorithms [15].

The role of language model in handwriting recognition

The fact is that whatever the recognition strategy, contextual knowledge (linguistic, domain, or any other pertinent information) needs to be incorporated into the recognition process to reduce the ambiguity and achieve acceptable performance. The lexicon is such a source of linguistic and domain knowledge. Most of the recognition systems rely on a lexicon during the recognition, the so-called lexicon-driven systems, or also after the recognition as a post-processor of the recognition hypotheses [20,46,77]. However, systems that rely on a lexicon in the early stages have had more success, since they look directly for a valid word [20]. Lexicons are very helpful in overcoming the ambiguity involved in the segmentation of words into characters, and the variability of character shapes [78,79]. Furthermore, lexicons are not only important in improving the accuracy, but also in limiting the number of possible word hypotheses to be searched [20,80]. This is particularly important to limit the computational complexity during the recognition process.

Open vocabulary systems are those based on another form of language model, such as *n-grams* (unigrams, bigrams, trigrams, etc.) [42,44,81,82]. However, when such a kind of language model is used instead of a limited lexicon, the recognition accuracy decreases [43]. The use of n-grams and statistical language modelling are discussed later.

Large vocabulary problems

After having presented the main elements involved in a handwriting recognition system, we now identify the main problems that arise when large vocabularies are used. Generally speaking, there are two basic requirements for any large vocabulary recognition system: accuracy and speed.

The problems related to accuracy are common to small and medium vocabularies. However, the task of recognising words from a small vocabulary is much easier than from a large lexicon (where more words are likely to be similar to each other). With an increasing number of word candidates, the ambiguity increases due to the presence of more similar words in the vocabulary, and that causes more confusion to the classifiers. A common behaviour of the actual systems is that the accuracy decreases as the number of words in the lexicon grows. However, there is not a clear relation between these two factors. It depends upon the particular characteristics of the systems. Figure 3 shows an example of words taken from a real lexicon of French city names, where some of them differ only by one or two characters.

It has been shown that when a large amount of training data is available, the performance of a word recogniser can generally be improved by creating more than one model for each of the recognition units [2,83], because it provides more accurate representation of the variants of handwriting. However, this can be one of the possible ways to improve the accuracy of large vocabulary recognition systems. Furthermore, the use of contextual-dependent models may be another feasible solution. On the other hand, while multiple models may improve the accuracy, they also increase the computational complexity.

Notwithstanding, it is not only the accuracy that is affected; the recognition speed is another aspect that is severely affected by the lexicon growth. Most of the problems with computational complexity and processing time in handwriting recognition arise from the fact that most current recognition systems rely on very time-consuming search algorithms, such as the standard dynamic programming, Viterbi, or forward algorithms. However, the speed aspect has not been considered by many researchers in handwriting recognition, mainly because they have not been dealing with large vocabularies. This aspect is overlooked in small and medium vocabularies because typical recognition speeds are of the order of milliseconds [84]. Besides the problem of accuracy and complexity, the development of large vocabulary recognition systems also requires large datasets both for training and testing. Currently available databases are very limited, both in the number of words as well as in the diversity of words (number of different words).

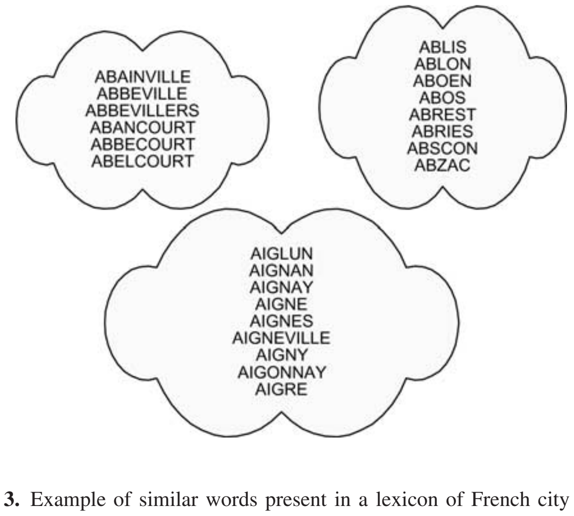

Fig. 3. Example of similar words present in a lexicon of French city names

The complexity of handwriting recognition

It is worth identifying the elements responsible for the computational complexity of a handwriting recognition system. Indeed, the complexity of the recognition is strongly dependent on the representation used for each of the elements. Recall that the basic problem in handwriting recognition is, given an input pattern represented by a sequence of observations (or primitives) O and a recognition vocabulary represented by $\mathcal{R}$, find the word $w \in \mathcal{R}$ that best matches the input pattern. In describing the computational complexity of the recognition process, we are interested in the number of basic mathematical operations, such as additions, multiplications and divisions, it requires. The computational complexity of the recognition, denoted as $\mathcal{C}$ is given by

$$\mathcal{C} = \mathbb{O}(TVLM) \tag{4}$$

where T is the length of the observation sequence, V is the vocabulary size, L is the average length of the words in the vocabulary (in characters), and we assume that subword models are represented by M parameters. This is a rough approximation considering that each character has only one reference pattern. This may be true if we consider only one type of handwriting style, e.g. handprinted words. However, in the unconstrained handwritten case, more than one reference pattern per character is usually necessary, because a single one is not enough to model the high variability and ambiguity of human handwriting. Assuming that each word is either handprinted or cursive (Fig. 4a), and that each character has a cursive and a handprinted reference pattern, the computational complexity increases linearly as

$$\mathcal{C} = \mathbb{O}(HTVLM) \tag{5}$$

where H denotes the number of models per class. However, if we assume that each word contain characters of both styles, that is a mixture of handprinted and cursive characters (Fig. 4b), then the computational complexity blows up exponentially as

$$\mathcal{C} = \mathbb{O}(H^{L}TVM) \tag{6}$$

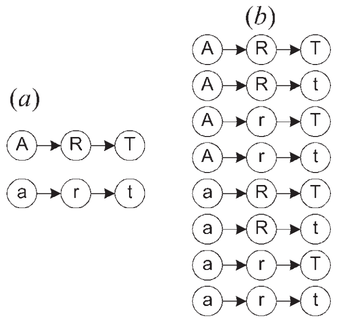

Fig. 4. Possible representations of a word. (a) Assuming only one writing style: handprinted or cursive; (b) assuming all possible combinations of handprinted and cursive characters

To get a feeling for how impractical the computation of Eqs (4)–(6) actually is, consider typical values of $H = 2$ models per character class, $L = 10$ characters per word, $M = 60$ frames, $V = 50{,}000$ words and $T = 60$ frames. With these values we get $\mathscr{C} \approx \mathbb{O}\ (1.8\ 10^9) - \mathbb{O}\ (1.8\ 10^{11})$. This computation to recognise a single word is already excessive for most modern machines[1]. In spite of the size of the vocabulary, it is only one of several factors that contribute to the high computational complexity of the recognition process; it is the most important factor affecting the development of more general applications. Therefore, management of the complexities in large vocabulary recognition systems, especially in real-time applications, poses a serious challenge to researchers.

Large vocabulary applications

Most of the actual research in handwriting recognition focuses on specific applications where the recognition vocabulary is relatively small. Clearly, the size of the vocabulary depends upon the application environment. The larger the vocabulary, the more flexible the application that utilises it can be. More generic applications need to quickly access large vocabularies of several thousand words. For a general text transcription system, a lexicon of 60,000 words would cover 98% of occurrences [21]. Future applications [19,25] have to be flexible enough to deal with dynamic lexicons and also words outside the vocabulary. Typical applications that require large vocabularies are:

- Postal applications: recognition of postal addresses on envelopes (city names, street names, etc.) [16,27,47,85];
- Reading of handwritten notes [21,25];
- Fax transcription [86];
- Generic text transcription: recognition of totally unconstrained handwritten notes [19,25];
- Information retrieval: retrieval of handwritten field from document images;
- Reading of handwritten fields in forms: census forms [87], tax forms [88], visa forms and other business forms;
- Pen-pad devices: recognition of words written on pen-pad devices [3,89].

[1] Current personal computers can perform between 1000 and 3000 million floating-point operations per second (MFLOPS).

In postal applications, the potential vocabulary is large, containing all street, city, county and country names. One of the main reasons for using word recognition in address reading is to disambiguate confusions in reading the ZIP code [21]. If the ZIP code is reliably read, the city will be known, but if one or more digits are uncertain, the vocabulary will reflect this uncertainty and expand to include other city names with ZIP codes that match the digits that were reliably read. However, as pointed out by Gilloux [63], when the recognition of the ZIP code fails, the recognition task is turned into a large vocabulary problem where more than 100,000 words need to be handled. Other applications that require large vocabularies are reading handwritten phrases on census forms [87], reading names and addresses on tax forms [88], reading fields of insurance and healthcare forms and claims, and reading information from subscription forms and response cards.

Organisation of survey

The objective of this survey is to present the current state of large vocabulary off-line handwriting recognition, and identify the key issues affecting the future applications. It reports on many recent advances that occurred in this field, particularly over the last decade. As we have seen, the process of matching the test pattern with all possible reference patterns is clearly impractical to be carried out on today's computing platforms. So, the envisaged solution is to limit either one or more of the variables involved. In the remainder of this survey, we present and discuss the techniques that have been employed to deal with the complexity of the recognition process. These methods basically attempt to reduce the variable V of Eqs (4)–(6). Most of the methods discussed are shown in Fig. 5. Section 2 presents different methods devoted to lexicon reduction. Section 3 presents some ways in which to reorganise the search space, i.e. the organisation of the words in the vocabulary. Several search strategies are presented in Section 4. Some of the methods presented in the preceding sections are illustrated in the case study presented in Section 5. In Section 6 we attempt to predict future issues and difficulties in the field to highlight the importance of improving the basic components of handwriting recognition systems to allow acceptable performance in real applications. The findings of the survey are summarised in the concluding section.

Before presenting the strategies, we observe that in this survey we do not cover the approaches related to feature vector reduction, since feature selection primarily aims to

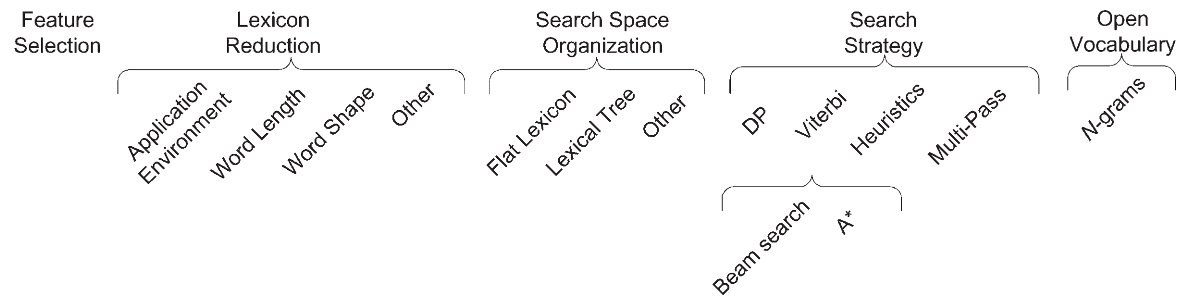

Fig. 5. Summary of strategies for large vocabulary handwriting recognition

select more discriminant features to improve the recognition accuracy. However, these methods can also be used to reduce the dimensionality of the feature vectors, say T, leading to less complex recognition tasks. Readers interested in investigating this particular aspect may refer to a more general description of feature selection methods [90,91], as well as to applications on handwriting recognition [92,93]. In the same manner, the survey does not cover methods related to the reduction of the number of class models (or class prototypes), say H, since this approach is not very common [94,95].

Lexicon reduction

One of the elements that contributes more to the complexity of the recognition task is the size of the lexicon. The problem of large lexicons is the number of times that the observation sequence extracted from the input image has to be matched against the words (or reference vectors) in the lexicon. So, a more intuitive approach attempts to limit the number of words to be compared during the recognition. Basically, pruning methods attempt to reduce the lexicon prior to recognition, that is, to reduce a global lexicon $\mathcal{R}$ to a subset $\mathcal{R}'$.

There is a chance that the pruning methods may throw away the true word hypothesis. Here we introduce the definition of *coverage* that refers to the capacity of the reduced (pruned) lexicon to include the right answer. So, the coverage indicates the error brought about by pruning (reducing) the lexicon. The effectiveness of a lexicon reduction technique can be measured by its coverage, which ideally has to be kept at 100% to avoid the introduction of errors. However, many authors do not report the performance of the lexicon reduction in terms of coverage, but look directly the effect on the recognition accuracy. This is the case for schemes that embed pruning mechanisms into the recognition process.

There are some basic ways to accomplish such a lexicon reduction task: knowledge of the application environment, characteristics of the input pattern, and clustering of similar lexicon entries. The application environment is the main source of information in limiting the lexicon size. In some cases, such as in bank cheque processing, the size of the lexicon is naturally limited to tens of words. Sometimes, even for applications where the number of words in the lexicon is large, additional sources of knowledge are available to limit the number of candidates to tens or hundred words. Other methods attempt to perform a pre-classification of the lexicon entries to evaluate how likely is the matching with the input image. These methods basically look at two aspects: word length and word shape. Other approaches attempt to find similarities between lexicon entries and organise them into clusters. So, during recognition, the search is carried out only on words that belong to more likely clusters. The details of some of methods are presented below.

Other sources of knowledge

Basically, in handwriting recognition the sources of knowledge that are commonly used depend upon the application environment. The application environment is usually a rich source of contextual information that helps us reduce the complexity of the problems to more manageable ones. Typical examples are banking and postal applications and language syntax.

Banking applications

One of the areas where researchers have devoted considerable attention is in the recognition of legal amounts on bank cheques. The reason is very simple: the lexicon size is limited to tens of words. This facilitates the gathering of data required for training and testing. Furthermore, there is also the courtesy amount that can be used to constrain (parse) the lexicon of the legal amount [32,38,96,97], or to improve the reliability of the recognition.

Postal applications

The postal application is perhaps the area where handwriting recognition techniques have been used more often [16,28,63,98]. Most of the proposed approaches first attempt to recognise the ZIP codes to further read other parts of the address, depending on the reliability in recognising the ZIP code. Conventionally, the ZIP code allows the system to reduce the lexicons of thousands of entries to a few hundred words [16,17,63,98–100]. So, the reduced lexicon can be processed using conventional search techniques such as the Viterbi and DP methods.

Language syntax

However, when no additional source of knowledge is available, other alternatives are necessary. In the case of generic content recognition, where the words are associated to form phrases and sentences, the application environment contributes little to reduce the lexicon. But here, linguistic knowledge plays an important role in limiting the lexicon. The use of language models based on grammars is very important to not only reduce the number of candidate words at each part of the text, but also to improve the accuracy [23–25,43,44]. However, this source is more suitable for the recognition of sentences than isolated words.

Word length

Short words can be easily distinguished from long words by comparing only their lengths. So, the length is a very simple criterion for lexicon reduction. The length of the observation sequence (or feature vector) extracted from the input image has intrinsically a hint about the length of the word from which the sequence was extracted. Many lexicon reduction methods make use of such information to reduce the number of lexicon entries to be matched during the recognition process [40,54,98,101–103]. Kaufmann et al [102] use a length classifier to eliminate from the lexicon those models which differ significantly from the unknown pattern in the number of symbols. For each model, a minimal and a maximal length are determined. Based on this range, a distance between a word and the model class is defined and used during the recognition process to select only the pertinent models. Kaltenmeier et al [98] use the word length information given by a statistical classifier adapted to features derived from Fourier descriptors for the outer contours to reduce the number of entries in vocabulary of city names. Koerich et al [103] used the length of the feature vector and the topology of the HMMs that model the characters to limit the length of the words dynamically in a level building framework. Knowing the minimum and maximum number of observations that each character HMM can absorb for a given feature vector, it is possible to estimate the maximum length of the words that can be represented by such a feature vector. However, the lexicon was organised as a tree-structure, and the lengths of the lexicon entries are available only during the search. So, the length constraint is incorporated to the recogniser, and the search is abandoned for certain branches that are above that limit.

Other methods do not rely on the feature vector to estimate the length of words, but on particular methods. Kimura et al [40] estimate the length of the possible word candidates using the segments resulting from the segmentation of the word image. Such estimation provides a confidence interval for the candidate words, and the entries outside of such an interval are eliminated from the lexicon. An over-estimation or an under-estimation of the interval leads to errors. Furthermore, the estimation of the length requires a reliable segmentation of the word, which is still an ill-posed problem. Powalka et al [54] estimate the length of cursive words based on the number of times an imaginary horizontal line drawn through the middle of the word intersects the trace of the pen in its densest area. A similar approach is used by Guillevic et al [101] to estimate word length and reduce the lexicon size. The number of characters is estimated using the counts of stroke crossing within the main body of a word.

Word shape

The shape of the words is another good hint about the length and the style of the words. Zimmerman and Mao [104] use key characters in conjunction with word length estimation to limit the size of the lexicon. They attempt to identify some key characters in cursive handwritten words, and use them to generate a search string. This search string is matched against all lexicon entries to select those best matched. A similar approach is proposed by Guillevic et al [101], but instead of cursive script, they consider only uppercase words. First, they attempt to locate isolated characters that are further pre-processed and input into a character recogniser. The character recognition results are used along with the relative position of the spotted characters to form a grammar. An HMM module is used to implement the grammar and generate some entries that are used to dynamically reduce the lexicon. Kaufmann et al [102] proposed a method of reducing the size of vocabulary based on the combination of four classifiers: a length classifier, the profile range, an average profile and a transition classifier. All the classifiers use as input the same feature vectors used by the recognition system. Seni et al [11] extract a structural description of a word and use it to derive a set of matchable words. This set consists of entries from the system lexicon that are similar in shape or structure to the input word. The set of matchable words forms the reduced lexicon that is employed during the recognition process. Madhvanath and Govindaraju [79] present a holistic lexicon filter that takes as input a chain-code of a word image and a lexicon and returns a ranked lexicon. First, the chain-code is corrected for slant and skew, and features such as natural length, ascenders and descenders are extracted, as well as assertions about the existence of certain features in certain specific parts of the word. The same features are extracted from lexicon entries (ASCII words) by using heuristic rules to combine the expected features of the constituent characters. A graph-based framework is used to represent the word image, the lexicon entries and their holistic features, and for computing three different distance measures (confidence of match, closeness, and degree of mismatch) between them. These three measures are computed for each lexicon entry and used to rank the hypotheses. A 50% reduction in the size of the lexicon with a 1.8% error is reported for a set of 768 lowercase images of city names. The same idea is used by Madhvanath et al [105,106] for pruning large lexicons for the recognition of off-line cursive script words. The holistic method is based on coarse representation of the word shape by downward pen-strokes. Elastic matching is used to compute the distance between the descriptor extracted from the image and the ideal descriptor corresponding to a given ASCII string. Distance scores are computed for all lexicon entries, and all words greater than a threshold are discarded. Henning and Sherkat [48] also use several holistic methods to reduce the lexicon in a cursive script recognition system. Features such as word length, diacritical marks, ascenders, descenders, combined as/descenders and segments crossing the word's axis, as well as several tolerance factors, are used. The first method was based on the letter candidates produced by a hierarchical fuzzy inference method [107], and the known distribution of the width of those candidates achieved a reduction of 44% of the hypotheses with an error rate of 0.5% for a 4126-word vocabulary. Using the number of possible axis crossings instead of letter candidates leads to a reduction of 30% with the same error rate. An extension of the method evaluates the occurrence of other physical features, such as ascenders, descenders, diacritical marks and as/descenders. The resulting reduction in the lexicon ranged from 53% to 99%.

Leroy [108] presents an approach for lexicon reduction in on-line handwriting recognition based on global features. First, alphabetic characters are encoded using global features (silhouette) with an associated probability. Using the silhouette of the characters, the words in the lexicon are encoded, and the probability of each word is given by the product of the compounding character silhouettes. Next, an epigenetic network is used to select the words in the lexicon that are best described by the silhouettes. An extension of this approach is presented by Leroy [109], where the silhouettes are combined to form words and a neural network is used to relate words to silhouettes. Those words that give the best scores are selected and encoded by more sophisticated features such as elliptic arcs and loop-shapes. Another neural network is used to select a new subset of words that give the best scores. Finally, diacritic marks [108] are used to select the final word candidates.

Table 1 summarises the experimental results obtained from some of the lexicon reduction methods presented above. The elements presented in Table 1 are the number of words in the lexicon, the number of samples used to test the proposed approach, the lexicon reduction achieved, the coverage of the resulting lexicon, the reduction in the recognition accuracy due to the reduced lexicon, and the speedup in recognition obtained by using the lexicon pruning. The speedup is not available for all methods, because most of them are presented separately from the recognition system. Table 2 shows some results of pruning methods for on-line handwriting recognition. Notice that the tables have different columns because the respective information is not always available for the methods presented.

Other approaches

Other approaches work towards different principles. Some of them avoid matching the input data against all lexicon entries during the search based on some measure of similarity of the lexicon entries, while others introduce constraints derived from the characteristics of the sequence of observations to restrict the search mechanism.

Gilloux [110] presented a method to recognise handwritten words belonging to large lexicons. The proposed approach uses degraded models of words which do not account for the alignment between letters of the words and features, but zones of the words. This allows a fast computation of word conditioned sequence probabilities. Furthermore, models are clustered independently for the length and features using the Euclidian distance between probability distributions and some distance threshold. So, a reduced number of base models whose words may share are matched only once to the data, resulting in a speedup of the process. The original word HMMs and the degraded word models are compared using a 59 k-entry lexicon and 3000 word images. In spite of being 20 times faster, the use of the degraded model causes a significant drop of 30% in the recognition rate. Gilloux [63] also proposes the use of Tabou search to reduce the number of words of a 59 k-entry lexicon. The approach consists of organising the search space according to the proximity of the lexicon entries. The Tabou method is a strategy for iterative improvement based on the local optimisation of the objective function. The criteria to be optimised is the likelihood of the observation sequence that represents a handwritten word, and the HMMs associated with the lexicon entries, and also a criterion of closeness between the HMMs. Lexicon reduction rates of 83–99.2% that correspond to 1.75–28 speedup factors are reported. But this improvement in speed is at the expense of reducing the coverage of the lexicon from 90% to 46%, which implies a reduction in the recognition rate of 4–23%. Farouz [99] presents a method for lexicon filtering based on bound estimation of Viterbi HMM probability from some properties of the observation sequence extracted from the word image. In the first step of the recognition process, the method estimates this bound for each lexicon entry, and as the entry comes close to the word image, unlikely candidates are eliminated from the lexicon. A lexicon reduction rate of 69% is reported, with a drop of 1% in the recognition rate. A similar approach is proposed by Wimmer et al [46], where an edit distance which works as a similarity measure between character strings is used to pre-select the lexicon to perform a post-processing of a word recognition system. Experimental results show a 100 speedup factor for a 10.6k-entry lexicon, with a

Table 1. Lexicon reduction approaches based on word shape for off-line handwriting recognition. Results are for cursive (lowercase) and handprinted (uppercase) words, respectively

Reference	Lexicon size	Test set	Reduction (%)	Coverage (%)	Speedup factor
Madhvanath et al [79]	1 k	768	50–90	98.2–75.0	–
Madhvanath et al [105]	21 k	825	99	74	–
Madhvanath et al [122]	21 k	825	95	95	–
Madhvanath et al [106]	23.6 k	760	95	75.5	–
Zimmermann et al [104]	1 k	811	72.9	98.6	2.2
Guillevic et al [101]	3 k	500	3.5	95.0	–

Table 2. Lexicon reduction approaches based on word shape for on-line recognition of cursive words

Reference	Lexicon size	Test set	Reduction (%)	Coverage (%)
Seni et al [10]	21 k	750	85.2–99.4	97.7
Leroy [108]	5 k	250	99.9	22
Leroy [109]	6.7 k	600	99.8	76
Hennig et al [48]	4 k	3750	97.5	84

reduction of 3–4% in accuracy. Koerich et al [103] incorporated in a level building algorithm two constraints to limit the search effort in an HMM-based recognition system. A time constraint that limits the number of observations at each level of the LBA according to the position of the character within the word, contributes to speedup the search by 39% for a 30 k-entry lexicon. A length constraint that limits the number of levels of the LBA according to the length of the observation sequence speeds up the search for the same lexicon by 5%. The combination of both constraints in the search algorithm gives 1.53 and 11 speedup factors over the conventional LBA and the Viterbi algorithms, respectively, for a 30 k-word vocabulary. A summary of the results achieved by some of the methods described in this section is presented in Table 3.

Discussion

The methods presented in this section attempt to prune the lexicon prior to recognition to reduce the number of words to be decoded during the recognition process. Knowledge of the application environment is a very efficient approach to reduce the lexicon, since it does not incur any error, because the reduced lexicon contains only those words that the system has to recognise effectively. However, the other methods rely on heuristics, and the lexicon is reduced at the expense of accuracy.

The approaches based on the estimation of the word length are very simple, and they can also be efficient. However, as they depend upon the nature of the lexicon, their use may be preceded by an analysis of the distribution of the length of the words. The same remark is valid for methods based on analysis of the word shape. The main drawback of the word shape methods is that they depend upon the writing style. This method seems to be more adequate for cursive handwriting. Another point is that some approaches involve the extraction of different features from the word image. Moreover, the robustness of some methods has not been demonstrated in large databases and large lexicons.

In spite of the fact that larger lexicons may cause more confusion in the recognition due to the presence of more similar words, reducing the lexicon by these proposed approaches implies a reduction in coverage, so the recognition accuracy also falls. It is easy to reduce the search space and improve the recognition speed by trading away some accuracy. It is much harder to improve recognition speed without losing some accuracy. The same problem is observed in speech recognition [111–115].

The results presented in Tables 1, 2 and 3 are not directly comparable, but they show the effectiveness of some of the proposed methods under particular experimental conditions. There is a lack of information concerning the effects of the pruning methods on the recognition system. Aspects such as the propagation of errors, selectiveness of writing styles, time spent to reduce the lexicon, etc. are usually overlooked. What is the computational cost of including lexicon reduction in the recognition pro-

Table 3. Pruning and lexicon reduction strategies for unconstrained off-line handwritten word recognition

Reference	Lexicon size	Test set	Reduction (%)	Coverage (%)	Reduction in recognition rate (%)	Speedup factor
Gilloux [110]	29.4 k	3 k	–	45–64	30	24
Gilloux [63]	60 k	4.2 k	99.2–83.0	46.0–90.0	23–4	28–1.7
Farouz [99]	49 k	3.1 k	69	–	1.0	1.75
Koerich et al [103]	30 k	4.6 k	–	–	3.0	11
Wimmer et al [46]	10.6 k	1.5 k	–	–	3–4	100

cess? How reliable are the lexicon reduction mechanisms, and what happens when they fail to reduce the number of lexicon entries? Most of the proposed approaches that we have presented in this section overlooked these aspects.

Search space organisation

In this section we present some approaches that attempt to reorganise the search space, that is, to reorganise all word hypotheses that have to be decoded during recognition, in order to exploit the presence of common prefixes in words that have similar spellings and avoid repeated computation of the same sequence of characters against the sequence of observations generated from the input image [4,6,80,98]. This approach avoids a reduction in the coverage, since the number of words in the lexicon is not changed. There are two basic ways of organising a lexicon: as a flat structure and as a tree structure.

Flat Lexicon

The term *flat lexicon* or *linear lexicon* denotes the fact that the words are kept strictly separate in the recognition process, that is, the matching between a given sequence of observations of unknown class and each word model is calculated independently. Word models are built *a priori* by concatenating the sub-word units or characters, and further, the matching between the sequence of observations and each model is calculated. So, the complexity increases linearly with the number of words in the vocabulary V and the average length of the words L.

Lexical tree

Organising the lexicon to be searched as a character tree instead of a linear structure of independent words has some advantages. This structure is referred to as a *lexical tree*, a *tree-structured lexicon* or as a *lexicon trie*. If the spellings of two or more words contain the same initial characters, in the lexical tree they will share this sequence of characters. If the search strategy adequately exploits the shared parts, the repeated computation of the shared parts can be avoided, reducing the complexity of the recognition. It can be viewed as a reduction of the average word length, given as

$$L' = \frac{L}{rf} \tag{7}$$

where L' is the new average word length and rf is the reduction factor, given as

$$rf = \frac{Ncl}{Nct} \tag{8}$$

where Ncl is the total number of characters in the flat lexicon, and Nct is the total number of characters in the tree-structured lexicon.

It is clear that as more words in the lexicon have common prefixes, there will be more advantages in using a lexical tree. This is more likely to happen when larger lexicons are used. Figure 6 shows the average number of characters for both a linear and a tree-structured lexicon, for different vocabulary sizes. However, the search technique must be adapted to exploit such shared parts and avoid unnecessary computation. The matching scores must be computed at the character level, and they must be retained during the search to be used further by other words with similar prefixes. Many authors have used tree-structured lexicons in the recognition of handwritten words [47,78,80,98,103,116]. Chen et al [78] present an algorithm for lexicon-driven handwritten word recognition where word images are represented by segmentation graphs and the lexicon is represented as a trie. The proposed approach saves about 48% and 15% of computation time over the standard DP algorithm when static and dynamic lexicon is used respectively. Many other authors have organised lexicons as lexical trees [84,103,106,117,118]. In on-line handwriting recognition, Fujisaki et al [116], Manke et al [6], Ratzlaff et al [119] and Jaeger et al [4] have also used a tree representation of the lexicon.

Other approaches

Other sorts of lexicon organisation have also been proposed, such as that based on a self-organising feature map [45] and on n-gram analysis [120]. The former maps the dictionary (build word clusters) in a two-dimensional space in an unsupervised way, preserving neighbourhood relationships. Results on a lexicon of 1125 words show that the number of lexicon entries tested decreases by

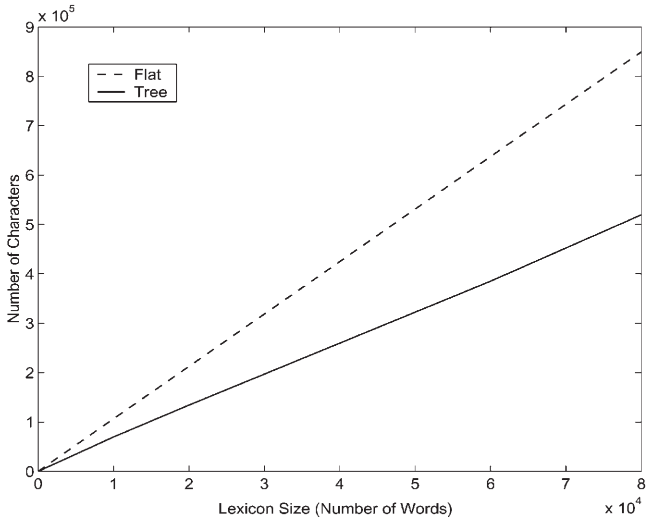

Fig. 6. Number of characters for different lexicon sizes generated from an 85.1 k-entry lexicon and organised as a flat or tree structure

using the proposed mapping, although the effects of the lexicon reduction on the coverage and recognition accuracy are overlooked. Procter et al [20] incorporate a lexicon directly with the search procedure, but instead of searching through all the lexicon entries, they use a list derived from the lexicon with the models that may legally follow the previously recognised sub-sequence. Another way to represent the search space is by a Direct Acyclic Word Graph (DAWG). It consists of sharing not only the prefixes, but also other common parts of the words [111,121], such as terminations. However, it is not clear how such a technique can be used for the problem of large vocabulary handwriting recognition due to the lack of experimental results.

Discussion

For small to medium size vocabularies, it is quite reasonable to use linear lexicons [15,16,21]. However, for larger vocabularies, the search space blows up linearly as a function of the number of entries, and that requires a formidable effort to search the entire vocabulary during the recognition. The organisation of the lexicon as a tree structure may minimise such a problem, however, since its effectiveness depends upon the nature of the lexicon, an analysis of the number of common prefixes within words in the lexicon should be considered. Other methods still have to prove their effectiveness in maintaining the same coverage of the whole vocabulary.

Search techniques

Another avenue that has not been fully exploited in large vocabulary handwriting recognition is to tackle directly the search problem, since it is the main bottleneck to recognition performance. Generally, the matching between the test and the reference patterns through a decoding algorithm is the most time-consuming procedure in a large vocabulary handwriting recognition system. The motivation of investigating search strategies comes from the fact that the majority of the current search techniques used in handwriting recognition are based on expensive DP methods, originally designed for small and medium scale problems. When the recognition task is extended from a 100-word vocabulary to a 10,000-word vocabulary, the number of hypotheses to be searched blows up, and these techniques can no longer efficiently handle such a large space. In handwriting recognition, the aspect of speed has been neglected because most researchers focus on small and medium vocabularies, where this aspect is not so important[2]. For this reason, it is difficult to find a paper that reports the processing time together with the accuracy; recognition rate has been the sole parameter to evaluate the proposed systems. Concerning all the proposed approaches presented so far, a common point is that none has focused on the search mechanism. They attempt to prune the lexicon prior to or during recognition, but they continue to use conventional search techniques to recognise the remaining entries.

Researchers in speech recognition have devoted much more attention to large vocabularies because they reflect real-life problems. So, they have developed alternative solutions to search in large vocabularies. While, in speech recognition, throwing away some accuracy while improving the speed and reducing the memory usage is widely accepted, researchers in handwriting recognition have been stricter, maybe because they are still focusing on improving the accuracy of small-scale problems. Most of the search techniques in handwriting recognition are inherited from speech recognition. However, in spite of the similarities of the recognition tasks, the inputs are quite different. While high level features that yield a sequence of observations that is quite short (e.g. 40–60 observations for a 13 character word) can be extracted from handwritten words, the speech waveform is converted into a sequence of acoustic vectors representing a smoothed log spectrum computed every 10 ms. A number of additional transformations is applied in order to generate the final acoustic vector [115], which usually has hundreds of observations. Furthermore, phone models are usually modelled by 3–5 state HMMs, while the models used in handwriting recognition can be based on structural assumptions, and include a high number of states (more than 10 states) [16,97,98].

To solve the handwriting recognition problem, we have to resolve the following sub-problems:

- The number of characters L that are encoded in the sequence of observations is usually not known (although it is possible to estimate it).
- The character boundaries within the word are not known (except the beginning of the first character and the end of the last character in the word).
- For a set of R reference patterns and for a given value of L, there are R^L possible combinations of composite matching patterns; however, this problem can be solved easily with a lexicon.

The search problem in handwriting recognition

The search problem in handwriting recognition can be formulated as: select a word reference with the highest score, given a test pattern corresponding to an unknown handwritten word represented as a sequence of observations[3] O, and a set of reference patterns denoted as R_v, $1 \leq v \leq V$ for a V-word vocabulary in which each pattern

[2] Depending on the constraints and experimental conditions, many small and medium vocabulary handwriting recognition systems are able to recognise words on personal computers in milliseconds.

[3] Alternatively, the test pattern can be defined as a sequence of primitive segments of the image such that $S = \{s_1, s_2, \ldots, s_P\}$ in which P is the length of the sequence and s_p is the pth primitive.

is of the form $R_v = (c_1c_2 \ldots c_L)$. However, characters are usually modelled by many parameters as $c_l = (f_1^l f_2^l \ldots f_M^l)$, in which M is the number of parameters in the character models, and f_m represents the mth parameter. In this section we provide a description of search techniques used in large vocabulary handwriting recognition.

Dynamic programming matching

Dynamic Programming (DP) methods are based on the principle of optimality and are the most used search strategy both in speech and handwriting recognition. Depending upon how the reference and test patterns are represented, distances or probability scores can be evaluated. DP methods compute the distance between a sequence of observations generated from the test pattern and all possible words in a lexicon. Each reference pattern is computed recursively by Eq. (1), allowing the optimal path search to be conducted incrementally, in a progressive manner. Although there are P possible moves that end at point l, the optimality principle indicates that only the best move is necessary to be considered. At the end, the best word hypotheses are those that have the minimum score with respect to the test pattern [122]. However, DP methods are mostly used with small and medium lexicons [17,34,40,41], since they perform a non-exhaustive, but still expensive, search procedure.

Viterbi search

The Viterbi algorithm is actually the same as the DP algorithm, except that the probability between the test and reference patterns is computed in the HMM rather than the distance measure between primitives. Viterbi search is mostly used when the reference patterns are represented by statistical models, and it has been used widely in handwriting recognition [14–16,50,80,97]. Viterbi search belongs to a class of breadth-first search techniques, where all hypotheses are pursued in parallel. It exploits the time invariance of the probabilities to reduce the complexity of the problem by avoiding the necessity for examining every route through the trellis. This procedure is known as *time synchronous Viterbi search*, because it completely processes at frame t before going into the frame $t + 1$. At the end, a backtracking pass gives the required state sequences. The performance of the conventional Viterbi algorithm in terms of recognition accuracy and speed is reported in some references [4,80,103,123].

An HMM can be completely characterised by the state-transition probability distribution matrix $A = \{a_{ij}\}$, the observation symbol probability distribution $B = \{b_j\}$, and the initial state distribution $\Pi = \{\pi_i\}$ as

$$\lambda = \{A, B, \Pi\} = \{a_{ij}, b_j, \pi_i, i, j = 1, \ldots, N\} \tag{9}$$

where N is the total number of states.

A conventional procedure employed in handwriting recognition is to have several sub-word HMMs that model characters, and concatenate such character HMMs (λ) at state level to build up word HMMs ($\hat{\lambda}$) according to the recognition vocabulary R_v, that are further matched against the sequence of observations using a strict left-right Viterbi algorithm. To find the single best state sequence, $q = (q_1q_2 \ldots q_T)$, given O, we need to define $\delta_t(i)$ that is the best score (highest probability) along a single path at frame t, which accounts for the first t observations and end in state i, and is computed as

$$\delta_t(i) = \max_{q_1, q_2, \cdots q_{t-1}} = P[q_1 q_2 \cdots q_{t-1}, q_t = i, o_1 o_2 \cdots o_t | \hat{\lambda}] \tag{10}$$

where $\hat{\lambda}$ is the word model formed by concatenation of sub-word HMMs as $\hat{\lambda} = \lambda_1 \oplus \lambda_2 \oplus \ldots \oplus \lambda_L$. By induction, we have

$$\delta_{t+1}(j) = \max_{1 \le i \le N} [\delta_t(i) a_{ij}] b_j(o_k) \tag{11}$$

To retrieve the state sequence, we need to keep track of the argument that maximised Eq.(11), for each t and j. A deeper discussion of the Viterbi procedure is out of the scope of this survey, but it can be found in Rabiner and Juang [124].

The Viterbi algorithm provides a computationally efficient way of analysing observations of HMMs that exploit recursion to reduce the computational load, but its complexity is comparable to other DP methods. Assuming that a character is modelled by an N-state HMM, the complexity can be approximated by

$$\mathscr{C} = \mathbb{O}(N^2TVL) \tag{12}$$

Beam search

In many applications, a complete Viterbi search is impractical due to the large size of the state space. Many variations of the Viterbi search have been proposed to improve its performance.

Instead of retaining all L candidates at every frame, a threshold can be used to consider only a group of likely candidates. The state with the highest probability can be found first, and each state with a probability smaller than the highest one can then be discarded from further consideration. This is the *beam search* algorithm, which can lead to substantial savings in computation with little loss of accuracy. Referring to the Eq. (1), it consists of setting pruning beams based on the best hypothesis score at l, denoted as φ_l^* (i, n). So, all points with scores lower than φ_l^* (i, n) + *beam* are activate at $l + 1$ and expanded, while all other points are pruned. In Fig. 2, that corresponds to instead of keeping all candidates at every frame (e.g. x_1, x_2, . . ., x_L), only the k candidates with the highest scores are allowed to remain (e.g. x_1, x_3, x_8, . . ., x_k, where $k < L$). The problem is that the beam size is determined empirically, and sometimes it can throw away the optimal solution.

The idea of the beam search can be expanded to the character [3] and word levels. Since word recognition must evaluate the matching between the test pattern and all possible reference patterns, the threshold can be derived from the final scores of words previously decoded, to discard from further consideration words in which the partial match falls below such a threshold.

Bippus et al [117] presented a scheme to reduce the computational load based on a lexical tree Viterbi beam search. However, for a task of recognising German city names with lexicon sizes of between 150 and 400 words, the proposed approach resulted in an inefficient search. Manke et al [6] presented a fast search technique for large vocabulary on-line handwriting recognition that combines a tree representation of the vocabulary with efficient pruning techniques to reduce the search space without losing much recognition performance compared to a flat exhaustive search. Dolfing [81] uses the same technique for on-line recognition of handwritten words. Jaeger et al [4] report a speedup factor of 10 over conventional Viterbi search in on-line handwriting recognition.

Another sort of beam search is proposed by Favata [53], which is a compromise between the exhaustive search and DP methods. The idea is not to speed up the recognition process, but to improve accuracy by carrying the k best partial matches forward by using a queue structure to hold each partial match. The proposed algorithm takes each of the current k matches and expands them to find the next incremental match between a character and segment. In Fig. 2 it corresponds to expanding not only the best path that reaches node x_1, but the k best. This procedure is more complex than conventional DP, because instead of expanding only the L best matches at every frame, it expands the k best matches where $k > L$.

A*

The A* algorithm belongs to a class of depth-first or best-first search techniques, where the most promising hypothesis is pursued until the end of the observation sequence is reached [13]. The A* algorithm requires an evaluation function to compare hypotheses of different lengths, and this is one of the key problems in heuristic search. For example, stack decoding is a variant of heuristic A* search based on the forward algorithm, where the evaluation function is based on the forward probability.

The search begins by adding all possible word prefixes to the stack. Then, the best hypothesis is removed from the stack and all paths from it are extended, evaluated and placed back in the stack. This search continues until a complete path that is guaranteed to be better than all paths in the stack has been found. A pruning mechanism can be used to save only a fixed number of hypotheses in the stack.

Bozinovic and Srihari [13] applied an A* algorithm search in off-line handwritten word recognition to match a sequence of features with words in a lexicon. Hypotheses are generated from the matching between prefixes of lexicon words and a sequence of features. For each hypothesis a score is assigned, and at each step the current best hypothesis is expanded and the list of hypotheses is reported. The resulting list undergoes a lexicon lookup, where inadmissible hypotheses are discarded, and the hypothesis with the best score continues to be expanded. Fujisaki et al [116] also used a decoder based on the A* search algorithm for on-line unconstrained handwriting recognition.

Multi-pass

Multi-pass search algorithms (also called fast-match) employ a coarse-to-fine strategy, where computationally inexpensive models are initially used to produce a list of likely word hypotheses that are later refined using more detailed and computationally demanding models [125]. Ratzlaff et al [119] used this search strategy in the recognition of unconstrained on-line handwritten text, and Bazzi et al [42] used it in an open-vocabulary OCR system.

The computational complexity of the first pass of such an approach can be approximated by

$$\mathscr{C}_1 = \mathbb{O}(M'TVL) \qquad (13)$$

where $M' < M$, and M' is the dimension of such a computationally inexpensive model. At the end to this first pass, only V' word hypotheses are selected, where $V' \ll V$. These word hypotheses represent the reduced vocabulary that is used in the second stage with more complex models M. So, the computational complexity of the second pass is given by

$$\mathscr{C}_2 = \mathbb{O}(MTV'L) \qquad (14)$$

To be efficient the combined computational complexity $\mathscr{C}_1 + \mathscr{C}_2$ must be lower than the conventional DP method. However, there is a risk of reducing the accuracy due to the use of heuristics and coarse models at the first pass.

Another example of multi-pass search is forward-backward search. Forward-backward search algorithms use an approximate time-synchronous search in the forward direction to facilitate a more complex and expensive search in the backward direction [125].

Discussion

Although the Viterbi algorithm and DP methods are the search strategies used more often in small and medium vocabulary applications, calculating the probability in this manner is computationally expensive, particularly with large models or long observation sequences. For large vocabulary handwriting recognition, a complete DP search is impractical. The other search techniques (A*, beam search and multi-pass) have been widely used in speech recognition, but not in handwriting recognition. Because they are faster, generally they are less accurate, providing sub-optimal solutions. Most of the research in handwriting recognition is focused on small and medium vocabulary problems that do not justify the use of less accurate search techniques. The Viterbi beam search has been the preferred choice of researchers dealing with large vocabularies. The stack decoder suffers from problems of speed, size, accuracy and robustness. For example, a bad choice of heuristic can lead to an explosion of the effective search space and the possibility of repeated computation [113].

In fact, there is a lack of studies that compare the advantages and disadvantages of different search methods applied to handwriting recognition both in terms of accuracy and speed.

Applications: a case study

In this section we present a large vocabulary off-line handwritten word recognition system and some attempts to speed up the recognition process without losing accuracy. Particularly, we report the results obtained by applying some of the proposed techniques presented in the preceding sections, say reorganisation of the search space, lexicon pruning by word length, and a fast search strategy. Furthermore, a distributed recognition scheme based on task partitioning among several processors is also presented.

Baseline system

The baseline handwritten word recognition system is composed of several modules: pre-processing, segmentation, feature extraction, training and recognition. The pre-processing normalises the word images in terms of slant and size. The images are then segmented into graphemes, and the sequence of segments is transformed into a sequence of symbols (or features). There is a set of 70 models among characters (26 uppercase and 26 lowercase), digits (10) and special symbols (8) that are modelled by 10-state transition-based HMMs with forward and null transitions [16]. The HMMs were trained and validated on a set of 12,049 and 3470 words, respectively, using the Maximum Likelihood criterion and through the Baum–Welch algorithm.

Experiments have been carried out using a test set of 4674 binary images of handwritten French city names. Two vocabularies were used: one containing 36,116 French city names (36.1k), with an average length of 11.09 characters, where the shortest word has three characters and the longest has 45 characters, and another with 85,092 city names[4] where the average word length is 11.20 characters. Compound words are present in both lexicons, and they correspond to a single city name made up of more than one word, such as ‘*Chire en Montreuil*’.

[4] We added to the 36.1 k entry lexicon more words corresponding to US city names (29.1 k), Italian city names (13.8 k), Brazilian city names (5.3 k), and Quebec city names (1.7 k). After eliminating duplicated words, we ended up with a vocabulary of 85,092 city names (85.1 k).

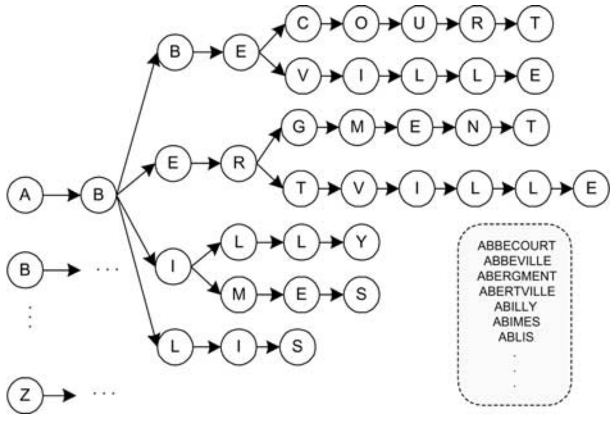

Fig. 7. Example of a lexical tree generated from some lexicon entries

Reorganisation of the search space

Recently, we proposed a large vocabulary off-line handwritten word recognition system based on a Syntax-Directed Level Building Algorithm (SDLBA) that outperforms a system using a conventional Viterbi search together with a flat lexicon in terms of recognition speed. In such a system, the sequences of features extracted from the input images are matched against the entries of a tree-structure lexicon (Fig. 7), where each node is represented by a character HMM. The asynchronous search proceeds breadth-first, and each tree node is decoded by the SDLBA. Contextual information about writing styles and case transitions is injected between the levels of SDLBA. Table 4 shows the average number of characters and the average reduction rate for different lexicon sizes taken from a 36.1 k entry lexicon of French city names.

The performance of both schemes (SDLBA and Viterbi flat lexicon) was evaluated using the same trained models and a testing database. Table 5 summarises the results for recognition accuracy, processing time and speedup factor. We have obtained a 5.5–7.7 speedup factor when employing a lexical tree and a decode algorithm based on the SDLBA. On the other hand, the accuracy was slightly

Table 4. Comparison between the average number of characters in the lexicon considering a flat structure and a tree structure

Lexicon size	Number of characters		Reduction factor (rf)
	Flat (Ncf)	Tree (Nct)	
10	119.36	113.05	1.05
100	1,198.4	987.53	1.21
1 k	11,998	8,361	1.43
10 k	120,035	66,558	1.80
30 k	360,012	173,631	2.07

Table 5. Comparison of recognition rate, processing time and speedup factor of the SDLBA with lexical tree, and Viterbi with a flat lexicon (VFL)

Lexicon size	Recognition rate (%)						Processing time (sec/word)		Speedup factor (× VFL)
	TOP1		TOP5		TOP10		SDLBA	VFL	
	SDLBA	VFL	SDLBA	VFL	SDLBA	VFL			
10	98.76	98.93	99.93	99.93	100.0	100.0	0.040	0.222	5.5
100	95.46	95.89	98.82	98.99	99.40	99.40	0.354	1.989	5.6
1 k	89.00	89.79	95.49	95.97	96.81	97.30	3.091	19.50	6.3
10 k	78.22	79.50	88.49	89.53	90.99	91.89	24.75	182.5	7.4
30 k	71.03	73.30	84.02	85.17	87.06	88.15	64.16	493.1	7.7

reduced (<2.5%). The advantage of using a lexical tree instead of a flat lexicon was highlighted by the fewer computations required and a consequent reduction in the processing time, notably for large lexicons.

Lexicon pruning

An extension of the latter approach is the Constrained Level Building Algorithm (CLBA) that limits the number of frames and the number of levels of the SDLBA [103]. A regression model that fits the response variables (namely the accuracy and the speed) to a nonlinear function of the constraints was proposed, and a statistical experimental design technique was employed to analyse the effects of the two constraints on the responses. Table 6 shows the speedup factor and loss of accuracy of the CLBA compared with the SDLBA. Limiting the number of observations according to the level of the LBA, as well as limiting the number of levels of the LBA by taking into account the length of the observation sequences, led to speedup factors of 1.43–1.53 and a slight reduction of 0.26–0.56% (TOP1) in the recognition rate for lexicons with 10–30,000 entries, respectively. If we compare these results with those of a previous version of the system based on a Viterbi-flat-lexicon scheme [16,80], the speedup factor is more impressive (7.2–11), with a reasonable reduction in the recognition rate (0.45–1.8%).

Table 6. Performance of the constrained SDLBA (CLBA) compared with the performance of the SDLBA scheme

Lexicon size	Speedup × SDLBA	Loss in accuracy (%)		
		TOP1	TOP5	TOP10
10	1.43	0.26	0.09	0.00
100	1.46	0.51	0.24	0.21
1 k	1.50	0.58	0.45	0.41
10 k	1.52	0.62	0.60	0.62
30 k	1.53	0.56	0.60	0.47

Fast search technique

Recently, we proposed a new search strategy designed to decode search spaces-where single character models appear repeatedly [123]. The search space is represented by words made up of single characters which may appear several times within the words. The search strategy handles the paradox of decoding the single characters separately from the context (words), but relying on the entire words to compute the overall probabilities, ensuring an optimal solution comparable with the one provided by the Viterbi search. Given a sequence of observations, it computes the *a posteriori* probabilities of single characters individually and independently of the context, and afterwards uses such probabilities to account for the different contexts imposed by a lexicon. We have demonstrated that this approach is equivalent to the Viterbi search in terms of finding the optimal solution, however, for simple HMM structures, the computational cost of the search can be reduced considerably.

Experiments have been carried out with the fast search strategy using the same test set mentioned before. Figures 8a and 8b show the average accuracy and processing time over the test set of 4674 words, respectively. The likelihood scores found by the fast search agree exactly with those found by the standard Viterbi algorithm in all cases [16], and the recognition rates are exactly the same obtained by the baseline system using Viterbi algorithm and a flat lexicon [16,80]. Figure 8c compares the processing time for the baseline system (VT flat) and the fast search strategy (FS flat and FS tree). We can observe that the average speedup factor is 25. Moreover, even for small and medium vocabularies, the fast search strategy is still advantageous.

Distributed scheme

One of the possible solutions to the problem of large vocabulary handwriting recognition could be the use of multiprocessor architectures with two to tens of multipurpose processors, since modern commercial processor architectures are increasingly capable of multiprocessor

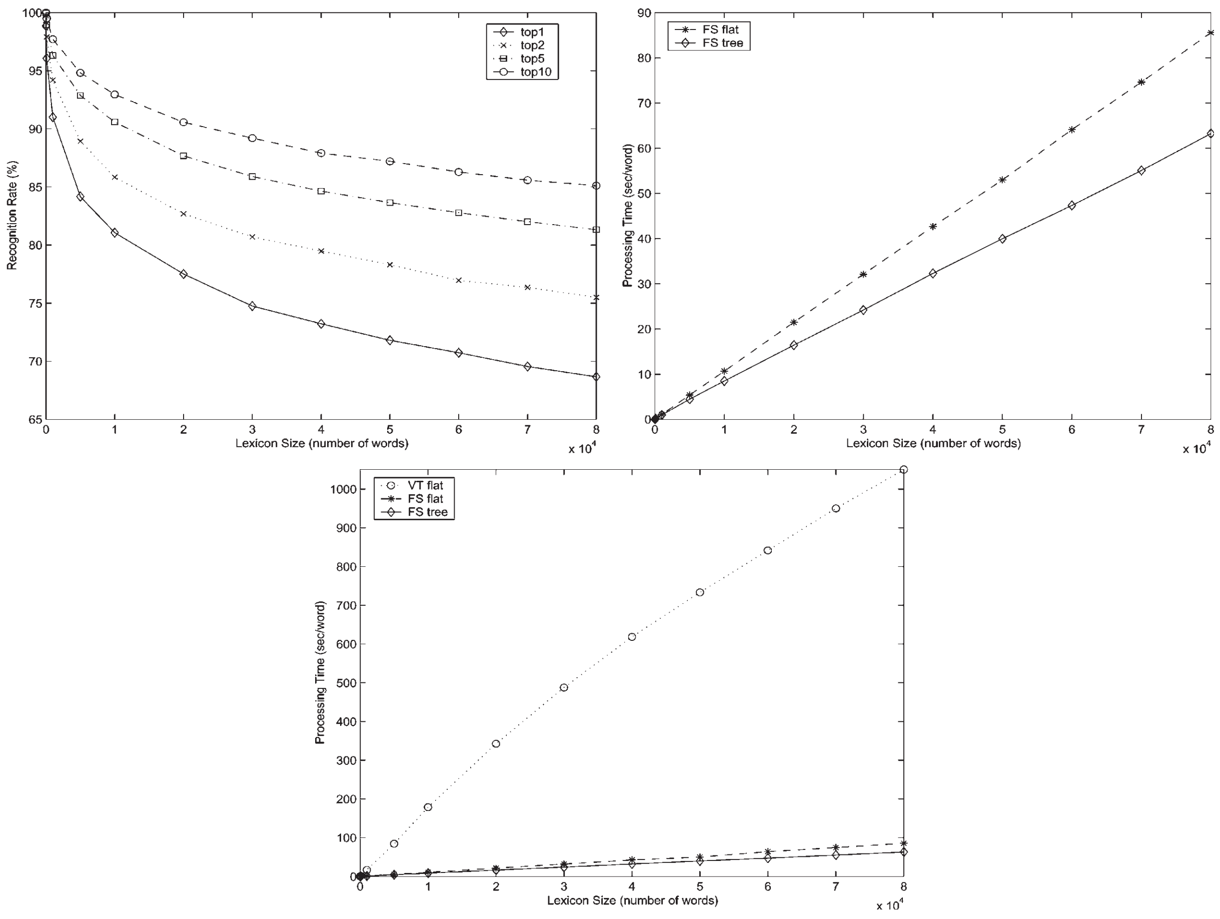

Fig. 8. Performance of the fast search strategy. (a) Recognition rates for different vocabulary sizes, (b) average processing time considering flat and tree structured lexicons, (c) comparison with the baseline system based on the Viterbi algorithm and a flat structured lexicon in terms of average processing time for different lexicon sizes

operation, and commercial operating systems support concurrency and multithreading within single applications.

This idea is exploited by Koerich et al [84] for speeding up an off-line handwritten word recognition system. The goal of the system is to achieve both full accuracy and high speed with large vocabularies. In a lexicon-driven approach, the matching of a sequence of observations against different words present in the lexicon can be executed in parallel on different processors by partitioning the lexicon. Figure 9 illustrates the parallelisation of the recognition engine through the partitioning of the lexicon.

The lexicon is split into *Pr* partial lexicons, where each will have *V*/*Pr* entries, (*Pr* denotes the number of processors and *V* the number of entries of the global lexicon). A thread is created for each partial lexicon, and a classifier is used just to match the sequence of features against the entries of such a lexicon. The same is done for the rest of the partial lexicons.

The output of each classifier is a list with the Top N best word hypotheses that give the highest likelihood score for that part of the global lexicon. The outputs of all classifiers are combined to decide which are the best

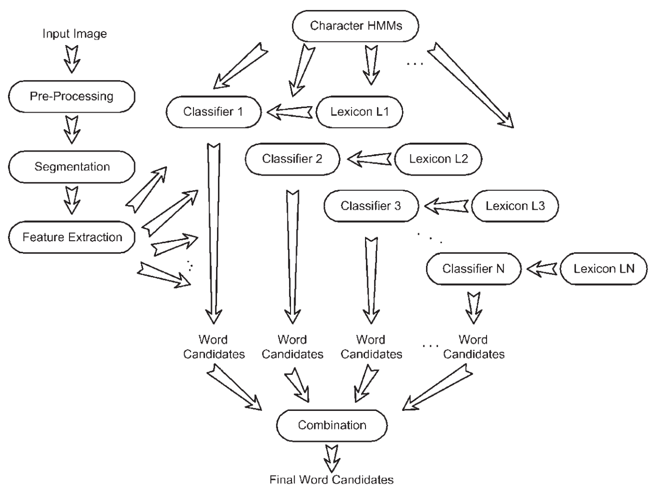

Fig. 9. Distributed recognition scheme where the recognition task is split into several similar classifiers that deal with subsets of the lexicon

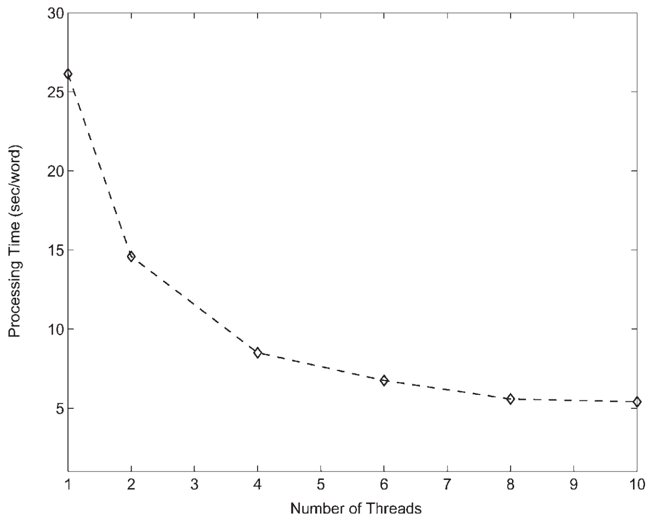

Fig. 10. Average processing time for the distributed recognition scheme according to the number of threads for a 30 k-entry lexicon

word candidates among all partial lexicons. Figure 10 illustrates the performance of the distributed recognition scheme according to the number of processors/threads used. Table 7 presents a summary of the results obtained by using the recognition/search strategies discussed in this section.

Discussion

A full performance evaluation of each method presented earlier in this survey was not the aim of this case study. Instead, we have tried to show that it is possible to implement a large vocabulary recognition system with a minor reduction in the recognition accuracy. Evaluating the performance in terms of recognition speed is certainly not the best way, because it is a machine-dependent measurement, and it can vary according to the machine load. On the other hand, it gives an idea on how long it takes in real applications.

Table 7. Summary of the performance of the baseline system incorporating different pruning, recognition and search techniques*

Reference	Search technique	Lexicon size	Coverage (%)	Speedup factor
80	Viterbi	30 k	100	–
80	SDLBA	30 k	98.0	7.7
103	CLBA	30 k	97.0	11
123	Fast Search (FS)	30 k	100	25
84	Distributed FS	30 k	100	100

*Results for a testing set containing 4674 words

Future issues

The ultimate goal of handwriting recognition will be to recognise generic handwritten texts. According to Plamondon and Srihari [7], the ultimate handwriting computer will have to process electronic handwriting in an unconstrained environment, deal with many writing styles and languages, work with arbitrary, user-defined alphabets, and understand any handwritten message by any writer [19,23,25,126]. So, to reach some of these goals, the use of large or open vocabularies is essential. Some attempts have been made to recognise sentences and phrases, but restricted to a medium lexicon (1600 words) [19,23,25]. Although large vocabulary off-line handwriting recognition appears to be feasible, some challenges need to be met before wide-spread applications.

Language modelling

The recognition of isolated words in a large vocabulary is already a demanding problem. If we attempt to recognise phrases and sentences, linguistic constraints have to be used to limit the search space. The use of a language model brings great gains in recognition accuracy. But a lexicon which limits the search to a set of permitted words is not the only solution. Grammars can also be used to limit which words are permissible in a given context to account for the frequencies of different words. However, grammars are typically used at the word level in the recognition of phrases and sentences, but not on isolated words [19,82].

For isolated word recognition, the idea is to use character-based language properties to model a word as a sequence of characters rather than word models. The use of n-grams enables the recognition of words with open vocabulary [120]. In spite of the improvement in the recognition rate that can be obtained by using such a language model, the achieved performance is still far below that of approaches that rely on limited lexicons, and it is not good enough for practical use [81].

Contextual-dependent models

To model co-articulation effects in large vocabulary handwriting recognition, the basic requirement is to model characters according to their context. Context refers to the immediate left and/or right neighbouring characters. If two characters have the same identity but different left of right context, they are considered different models. While contextual-dependent models are good for modelling co-articulation effects, there are a great number of them, which requires a large amount of data for training, as well as increasing the complexity of the recognition process.

Open vocabulary systems

One of the major problems of lexicon-driven systems is the encounter of an out-of-vocabulary word [20,82]. In such cases, words cannot be correctly recognised, so the system must provide some sort of mechanism to recognise them as out-of-vocabulary words instead of misclassifying them. However, for many applications, the size and content of vocabulary is controllable, and this problem can be managed (handled). Senior [82] uses a non-word Markov model together with a lexicon to allow open vocabulary recognition.

Lexicon-based systems allow only a closed set of words that constitute the recognition lexicon, however, open-vocabulary systems become possible with the use of a lexicon of characters and statistical language model at the character level (n-gram on sequences of characters) [42–44]. However, if open-vocabulary systems are faster and more flexible, they are also less accurate. Brakensiek et al [43,44] report a decrease of about 26% in word accuracy using no lexicon for an off-line handwritten word recognition system.

Real-time operation

In addition to obtaining an acceptable level of recognition accuracy, computationally efficient implementation is needed to exploit large vocabulary handwriting recognition technology. The throughput requirements for real-time applications are of the order of tens of milliseconds.

One of the possible solutions to achieve the throughout requirements for real-time applications could be the use of distributed processing in low cost workstation clusters. Modern commercial processor architectures are increasingly capable of multiprocessor operation, and commercial operating systems support concurrency and multithreading within single applications. The potential of distributed processing has not been fully exploited by researchers in handwriting recognition. Some few results were published recently [84,85,97]. In real applications the throughput is as important as the accuracy [29].

Workstation clusters have become an increasingly popular alternative to parallel supercomputers. Workstation clusters can be built from similar networked workstations. Since they can be built from regular components, they enjoy a tremendous price/performance advantage over traditional supercomputers [127]. With PCs available at less than $1000, clusters have become an extremely compelling way to run computationally demanding tasks in handwriting recognition, such as the training of classifiers and large vocabulary applications. Table 8 shows some performance figures of a handwriting recognition system run on different machines. The standardised processor benchmark for floating point performance (SPECpf95 – www.specbench.org) is given for some of these machines. As we can see, the large vocabulary problem is still a challenge, even for 1 GHz machines.

Database

For the development of unconstrained large vocabulary handwriting recognition systems, another limitation is the amount of data required for training and testing. Statistical methods provide an automatic procedure to ‘learn’ the regularities in the handwriting data directly. The need of a large set of good training data is thus more critical than ever. Furthermore, with the tendency of incorporating contextual information in the recognition, more data are needed for the training and validation of systems. The available databases (e.g. BERN, CAMBRIDGE, CEDAR and IRONOFF) are not adequate for such a task. The Bern database [24] has 12,198 words comprising a 100 word lexicon extracted from a handwritten page database produced by 200 writers. The Cambridge database [21] has 4053 words produced by a single writer comprising a 1334-word lexicon. The CEDAR database [128] contains 5632 city words plus 4938 state words. The IRONOFF database has 31,346 isolated words from a 197 word lexicon collected from about 600 different writers.

The above-mentioned databases are limited not only in the number of samples, but also on the diversity of words. Usually, not all words presented in the lexicon have a sample and there are many repetitions of more common words. However, building databases imply several steps such as the gathering, digitising, verification and labelling of data. These operations may be straightforward but tedious when the amount of data is large.

Discussion and concluding remarks

In this paper we have reviewed the field of large vocabulary handwriting recognition. First, the basic structure of a handwriting recognition system was introduced and attention was focused on the parts concerning large vocabulary applications. The main problems related to large vocabularies, with special attention on the complexity, and some of the foreseen applications that may use large vocabularies were presented. While we have focused on the recognition aspect, it is clear that pre-processing, feature extraction, character modelling and segmentation have to be treated in an integrated manner to achieve high performance.

Next, we presented some of the proposed methods to handle large vocabularies. These methods attempt to limit the number of words in the lexicon by gathering some dimensional information from the input such as length, shape, perceptual features, etc. However, since all of these methods rely on heuristics, they may reduce the coverage. Furthermore, some of them cannot be applied to unconstrained handwriting, since they are limited to cope with specific writing styles. So, to avoid introducing errors, instead of pruning the lexicon, other methods attempt to reorganise the lexicon to eliminate some redundancy and share common word prefixes. For both of these strategies, the recognition can be accomplished by classical algorithms that are exhaustive and not very efficient from the

Table 8. Figures of performance of the fast search strategy (Section 5.4) on different machines for an 85.1 k recognition task

Machine	Speed (MHz)	Specpf95	Processing time (sec/word)	Speedup factor
Ultra1	167	9.06	77.8	–
Ultra60	296	18.4	44.8	1.73
Pentium II	450	13.7	29.8	2.61
AMD Athlon	1100	30.2	10.7	7.27
Enterprise 6000	10 × 176	20.9	10.1	7.70
Cluster AMD Athlon	10 × 1100	–	1.11	70.09

point of view of search. So, another sort of strategy that looks directly at the classification mechanism comes up. The presentation of the majority of the methods and approaches is very limited, and authors rarely provide a careful analysis of proposed methods both in terms of accuracy and speed. So, it is very difficult to evaluate the effectiveness of the methods.

Different from other small-scale problems, such as digit recognition and small and restricted lexicon applications, large-scale problems as is the case of large vocabulary handwriting recognition, is still immature. It is clear that much more needs to be done before robust, general-purpose recognition systems become available. Given a well-defined task, the technology is useable now. However, to pass from small vocabularies to large ones, from isolated words to phrases and sentences, will require a tremendous effort. It is our view that the problem of handwriting recognition in the case of large vocabularies remains open. Improvements are still needed to achieve satisfactory performance, both in terms of accuracy and speed. Maybe a combination of different pruning methods, lexical tree, and fast search strategies will find the best trade-off. The availability of new hardware seems to be very helpful, but we cannot put our hopes on it, because as we have seen, the increase of computational power is not fully reflected in the performance of the systems. Furthermore, acceptable performance has to be achieved on regular personal computers to widen the use of large vocabulary handwriting recognition.

Another important aspect that is two folded is the use of a language model. While it plays an important role in the actual systems, it is also responsible for the high complexity of the recognition task. Modelling handwriting without such knowledge is not viable and is unreliable. On the other hand, the use of language models incurs more complex and computationally expensive approaches, that in spite of being more accurate, are time consuming. An intermediate solution is to use n-grams to build open vocabulary systems. However, a lot of effort has to be put in this direction to improve the accuracy, which is still is much inferior to lexicon-driven approaches. Future issues should focus on broadband applications with few constraints, and that will be a challenge for the coming years.

More studies have to be conducted, mainly in the search aspect. Our case study has already compared a number of search strategies. Most of the works are limited to the use of a specific technique. A comprehensive comparative study will be very useful to guide future researchers in large vocabulary applications.

The paper has concentrated on an appreciation of the principles and methods. In spite of this, we have presented several experimental results; we have not attempted to compare the effectiveness of the methods directly. In practice, it is very difficult to assess techniques implemented in different systems and tested under different experimental conditions. An exception is the case study, where the results are comparable since they were obtained under the same conditions, i.e. hardware, database, etc. However, the results are very limited, since it was not possible to cover all the techniques presented in this survey.

We have included a list of references sufficient to provide a more detailed understanding of the approaches described. However, some of them are not directly related to large vocabulary problems or off-line handwriting recognition, even though, we judged that it is pertinent to have them. We apologize to researchers whose important contributions may have been overlooked and we welcome their feedback.

Acknowledgements

The authors would like to acknowledge the CNPq-Brazil (grant refs 200276–98/0) and the MEQ-Canada for supporting this research, and the Service Technique de la Poste (SRTP–France) for providing us the database and the baseline system.

References

1 Bellegarda EJ, Bellegarda JR, Nahamoo D, Nathan K. A probabilistic framework for on-line handwriting recognition. Proc 3rd International Workshop on Frontiers in Handwriting Recognition, Buffalo, NY, 1993; 225–234
2 Connell S. Online Handwriting Recognition Using Multiple Pattern Class Models. PhD thesis, Michigan State University, East Lansing, MI, May 2000
3 Jaeger S, Manke S, Reichert J, Waibel A. Online handwriting recognition: The NPEN++ recognizer. Int J Document Analysis and Recognition 2001; 3:169–180
4 Jaeger S, Manke S, Waibel A NPEN++: An on-line handwriting recognition system. Proc 7th International Workshop on Frontiers in Handwriting Recognition, Amsterdam, Netherlands, 2000; 249–260

5 Kosmala A, Rottland J, Rigoll G. Improved on-line handwriting recognition using context dependent hidden Markov models. Proc 4th International Conference on Document Analysis and Recognition, Ulm, Germany, 1997; 641–644

6 Manke S, Finke M, Waibel A. A fast search technique for large vocabulary on-line handwriting recognition. In: Downton AC, Impedovo S, eds, Progress in Handwriting Recognition, World Scientific, Singapore, 1996; 437–444

7 Plamondon R, Srihari SN. On-line and off-line handwriting recognition: A comprehensive survey. IEEE Trans Pattern Analysis and Machine Intelligence 2000; 11(1):68–89

8 Schenkel M, Guyon I, Henderson D. On-line cursive script recognition using time delay neural networks and hidden Markov models. Machine Vision and Applications 1995; 8(4):215–223

9 Seiler R, Schenkel M, Eggimann F. Off-line cursive handwriting recognition compared with on-line recognition. Proc International Conference on Pattern Recognition, Vienna, Austria, 1996; 505–509

10 Seni G, Srihari RK. A hierarchical approach to on-line script recognition using a large vocabulary. Proc 4th International Workshop on Frontiers in Handwriting Recognition, Taipei, ROC, 1994; 472–479

11 Seni G, Srihari RK, Nasrabadi N. Large vocabulary recognition of on-line handwritten cursive words. IEEE Trans Pattern Analysis and Machine Intelligence 1996; 18(7):757–762

12 Tappert CC, Suen CY, Wakahara T. The state of art in on-line handwriting recognition. IEEE Trans Pattern Analysis and Machine Intelligence 1990; 12(8):787–808

13 Bozinovic RM, Srihari SN. Off-line cursive script word recognition. IEEE Trans Pattern Analysis and Machine Intelligence, 1989; 22(1):63–84

14 Bunke H, Roth M, Schukat-Talamazzini EG. Off-line cursive handwriting recognition using hidden markov models. Pattern Recognition 1995; 28(9):1399–1413

15 Chen MY, Kundu A, Zhou J. Off-line handwritten word recognition using a hidden Markov model type stochastic network. IEEE Trans Pattern Analysis and Machine Intelligence 1994; 16(5):481–496

16 El-Yacoubi A, Gilloux M, Sabourin R, Suen CY. Unconstrained handwritten word recognition using hidden markov models. IEEE Trans Pattern Analysis and Machine Intelligence 1999; 21(8):752–760

17 Gader PD, Mohamed MA, Chiang JH. Handwritten word recognition with character and intercharacter neural networks. IEEE Trans Systems, Man and Cybernetics – Part B 1994; 27:158–164

18 Kim G, Govindaraju V. A lexicon driven approach to handwritten word recognition for real-time applications. IEEE Trans Pattern Analysis and Machine Intelligence 1997; 19(4):366–379

19 Kim G, Govindaraju V, Srihari SN. An architecture for handwriting text recognition systems. Int J Document Analysis and Recognition 1999; 2:37–44

20 Procter S, Illingworth J, Mokhtarian F. Cursive handwriting recognition using hidden Markov models and a lexicon-driven level building algorithm. IEE Proc Vision, Image and Signal Processing 2000; 147(4):332–339

21 Senior AW, Robinson AJ. An off-line cursive handwriting recognition system. IEEE Trans Pattern Analysis and Machine Intelligence 1998; 20(3):309–321

22 Srihari S, Govindaraju V, Srihari R. Handwritten text recognition. Proc 4th International Workshop on Frontiers in Handwriting Recognition, Taipei, Taiwan, 1994; 265–274

23 Marti U, Bunke H. Towards general cursive script recognition. Proc 6th Int Workshop on Frontiers in Handwriting Recognition, Taejon, Korea, 1998; 379–388

24 Marti U, Bunke H. A full English sentence database for off-line handwriting recognition. Proc 5th International Conference on Document Analysis and Recognition, Bangalore, India, 1999; 705–708

25 Marti U, Bunke H. Handwritten sentence recognition. Proc 15th International Conference on Pattern Recognition, Barcelona, Spain, 2000; 467–470

26 Heutte L. Reconnaissance de caractère manuscrits: Application à la lecture automatique des chèques et des enveloppes postales. PhD thesis, Université de Rouen, Rouen, France, 1994

27 Kornai A. An experimental HMM-based postal ocr system. Proc International Conference on Acoustics, Speech and Signal Processing, Munich, Germany, 1997; 3177–3180

28 Srihari SN. Recognition of handwritten and machine-printed text for postal address interpretations. Pattern Recognition Letters 1993; 14:291–302

29 Srihari SN, Kuebert EJ. Integration of handwritten address interpretation technology into the united states postal service remote computer reader system. Proc 4th International Conference on Document Analysis and Recognition, Ulm, Germany, 1997; 892–896

30 Dimauro G, Impedovo S, Pirlo G, Salzo A. Automatic bankcheck processing: A new engineered system. In: Impedovo S, Wang PSP, Bunke H, eds, Int J Pattern Recognition and Artificial Intelligence, World Scientific, 1997; 467–503

31 Guillevic D, Suen CY. HMM-KNN word recognition engine for bank cheque processing. Proc International Conference on Pattern Recognition, Brisbane, Australia, 1998; 1526–1529

32 Guillevic D, Suen CY. Cursive script recognition applied to the processing of bank cheques. Proc 3rd International Conference on Document Analysis and Recognition, Montreal, Canada, 1995; 11–14

33 Heutte L, Pereira P, Bougeois O, Moreau J, Plessis B, Courtellemont P. Multi-bank check recognition system: Consideration on the numeral amount recognition module. In: Impedovo S, Wang PSP, Bunke H, eds, Int J Pattern Recognition and Artificial Intelligence, World Scientific, 1997; 595–617

34 Kim G, Govindaraju V. Bankcheck recognition using cross validation between legal and courtesy amounts. In: Impedovo S, Wang PSP, Bunke H, eds, Int J Pattern Recognition and Artificial Intelligence 1997; 657–673

35 Kim JH, Kim KK, Suen CY. Hybrid schemes of homogeneous and heterogeneous classifiers for cursive word recognition. Proc 7th International Workshop on Frontiers in Handwriting Recognition, Amsterdam. Netherlands, 2000; 433–442

36 Knerr S, Augustin E. A neural network-hidden Markov model hybrid for cursive word recognition. Proc 14th International Conference on Pattern Recognition, Brisbaine, Australia, 1998; 1518–1520

37 Kornai A, Mohiuddin KM, Connell SD. Recognition of cursive writing on personal checks. Proc 5th International Workshop on Frontiers in Handwriting Recognition, Essex, UK, 1996; 373–378

38 Lee L, Lizarraga M, Gomes N, Koerich A. A prototype for Brazilian bankcheck recognition. In: Impedovo S, Wang PSP, Bunke H, eds, Automatic Bankcheck Processing, World Scientific, 1997; 549–569

39 Lallican PM, Gaudin CV, Knerr S. From off-line to on-line handwriting recognition. Proc 7th International Workshop on Frontiers in Handwriting Recognition, Amsterdam, Netherlands, 2000; 303–312

40 Kimura F, Shridhar M, Chen Z. Improvements of a lexicon directed algorithm for recognition of unconstrained handwritten words. Proc. International Conference on Document Analysis and Recognition, Tsukuba, Japan, 1993; 18–22

41 Shridhar M, Houle G, Kimura F. Handwritten word recognition using lexicon free and lexicon directed word recognition algorithms. Proc 4th International Conference on Document Analysis and Recognition, Ulm, Germany, 1997; 861–865

42 Bazzi I, Schwartz R, Makhoul J. An omnifont open-vocabulary ocr system for English and Arabic. IEEE Trans Pattern Analysis and Machine Intelligence 1999; 21(6):495–504

43 Brakensiek A, Rottland J, Kosmala A, Rigoll G. Off-line handwriting recognition using various hybrid modeling techniques and character N-grams. Proc 7th International Workshop on Frontiers in Handwriting Recognition, Amsterdam, Netherlands, 2000; 343–352

44 Brakensiek A, Willett D, Rigoll G. Unlimited vocabulary script recognition using character n-grams. Proc 22nd DAGM Symposium, Tagungsband Springer-Verlag, Kiel, Germany, 2000

45 Menier G, Lorette G. Lexical analyzer based on a self-organizing feature map. Proc 4th International Conference on Document Analysis and Recognition, Ulm, Germany, 1997; 1067–1071

46 Wimmer Z, Dorizzi B, Gallinari P. Dictionary preselection in a neuro-Markovian word recognition system. Proc 5th International Conference on Document Analysis and Recognition, Bangalore, India, 1999; 539–542

47 Dzuba G, Filatov A, Gershuny D, Kill I. Handwritten word recognition – the approach proved by practice. Proc 6th International Workshop on Frontiers in Handwriting Recognition, Taejon, Korea, 1998; 99–111

48 Hennig A, Sherkat N. Cursive script recognition using wildcards and multiple experts. Pattern Analysis and Applications 2001; 4(1):51–60

49 Madhvanath S, Govindaraju V. The role of holistic paradigms in handwritten word recognition. IEEE Trans Pattern Analysis and Machine Intelligence, 2001; 23(2):149–164

50 Mohamed MA, Gaden P. Handwritten word recognition using segmentation-free hidden Markov modeling and segmentation-based dynamic programming technique. IEEE Trans Pattern Analysis and Machine Intelligence, 1996; 18(5):548–554

51 Chen MY, Kundu A, Srihari SN. Variable duration hidden markov model and morphological segmentation for handwritten word recognition. IEEE Trans Image Processing 1995; 4(12):1675–1688

52 Blumenstein M, Verma B. Neural-based solutions for the segmentation and recognition of difficult handwritten words from a benchmark database. Proc 5th International Conference on Document Analysis and Recognition, Bangalore, India, 1999; 281–284

53 Favata JT. Offline general handwritten word recognition using an approximate beam matching algorithm. IEEE Trans Pattern Analysis and Machine Intelligence 2001; 23(9):1009–1021

54 Powalka RK, Sherkat N, Whitrow RJ. Word shape analysis for a hybrid recognition system. Pattern Recognition 1997; 30(3):412–445

55 Koerich AL, Leydier Y, Sabourin R, Suen CY. A hybrid large vocabulary handwritten word recognition system using neural networks with hidden Markov models. 8th International Workshop on Frontiers in Handwriting Recognition, Niagara-on-the-Lake, Canada, 2002, (accepted)

56 Kwok TY, Perrone MP. Adaptive n-best list handwritten word recognition. Proc 6th International Conference on Document Analysis and Recognition, Seattle, WA, 2001

57 Arica N, Yarman-Vural F. An overview of character recognition focused on off-line handwriting. IEEE Trans Systems, Man and Cybernetics – Part C: Applic and Rev 2001; 31(2):216–233

58 Casey RG, Lecolinet E. A survey of methods and strategies in character segmentation. IEEE Trans Pattern Analysis and Machine Intelligence, 1996; 18(7):690–706

59 Lecolinet E, Baret O. Cursive word recognition: Methods and strategies. In: Fundamentals in Handwriting Recognition, Springer-Verlag, France, 1994; 126:235–263

60 Lu Y, Shridhar M. Character segmentation in handwritten words – an overview. Pattern Recognition, 1996; 29(1):77–96

61 Steinherz T, Rivlin E, Intrator N. Offline cursive script word recognition – a survey. Int J Document Analysis and Recognition 1999; 2:90–110

62 Trier O, Jain AK, Taxt T. Feature extraction methods for character recognition. Pattern Recognition 1996; 29(4):641–662

63 Gilloux M. Réduction dynamique du lexique par la méthode tabou. Proc Colloque International Francophone sur l'Ecrit et le Document, Quebec, Canada, 1998; 24–31

64 Gorsky ND. Experiments with handwriting recognition using holographic representation of line-images. Pattern Recognition Letters 1994; 15(9):853–859

65 Govindaraju V, Krishnamurthy RK. Holistic handwritten word recognition using temporal features derived from off-line images. Pattern Recognition Letters 1996; 17(5):537–540

66 Bellman RE. Dynamic Programming. Princeton University Press, Princeton, NJ, 1957

67 Jain AK, Duin RPW, Mao J. Statistical pattern recognition: A review. IEEE Trans Pattern Analysis and Machine Intelligence 2000; 20(1):4–38.

68 LeCun Y, Matan O, Boser B, Denken J, Henderson D, Howard R, Hubbard W, Jackel L. Handwritten zip code recognition with multilayer networks. Proc 10th International Conference on Pattern Recognition, Atlantic City, NJ, 1990; 35–40

69 Oliveira LS, Sabourin R, Bortolozzi F, Suen CY. A modular system to recognize numerical amounts on brazilian bank cheques. Proc 6th International Conference on Document Analysis and Recognition, Seattle, WA, 2001; 389–394

70 Ollivier D, Weinfeld M, Guegan R. Combining different classifiers and level of knowledge: A first step towards an adaptive recognition system. Proc 6th Int Workshop on Frontiers in Handwriting Recognition, Taejon, Korea, 1998; 89–98

71 Burges CJC, Ben JI, Denken JS, Lecun Y, Nohl CR. Off-line recognition of handwritten postal words using neural networks. Int J Pattern Recognition und Artificial Intelligence, 1993; 7(4):689–704

72 Farouz C, Gilloux M, Bertille JM. Handwritten word recognition with contextual hidden markov models. Proc 6th Int Workshop on Frontiers in Handwriting Recognition, Taejon, Korea, 1998; 133–142

73 Wang F, Vuurpij L, Schomaker L. Support vector machines for the classification of western handwritten capitals. Proc 7th International Workshop on Frontiers in Handwriting Recognition, Amsterdam, Netherlands, 2000; 167–176

74 Vuurpijl L, Schomaker L. Two-stage character classification: A combined approach of clustering and support vector classifiers. Proc 7th International Workshop on Frontiers in Handwriting Recognition, Amsterdam, Netherlands, 2000; 423–432

75 Ney H, Ortmanns S. Dynamic programming search for continuous speech recognition. IEEE Signal Processing Magazine, 1999; 16(5):64–83

76 Huang XD, Ariki Y, Jack MA. Hidden Markov Models for Speech Recognition. Edinburgh University Press, 1990

77 Srihari RK. Use of lexical and syntactic techniques in recognizing handwritten text. Proc of ARPA Workshop on Human Language Technology, Princeton, NJ, 1994; 403–407

78 Chen DY, Mao J, Mohiuddin K. An efficient algorithm for matching a lexicon with a segmentation graph. Proc 5th International Conference on Document Analysis and Recognition, Bangalore, India, 1999; 543–547

79 Madhvanath S, Govindaraju V. Holistic lexicon reduction. Proc 3th International Workshop on Frontiers in Handwriting Recognition, Buffalo, NY, 1993, 71–78

80 Koerich AL, Sabourin R, Suen CY, El-Yacoubi A. A syntax-directed level building algorithm for large vocabulary handwritten word recognition. Proc 4th International Workshop on Document Analysis Systems, Rio de Janeiro, Brasil, 2000; 255–266

81 Dolfing JGA. Handwriting Recognition and Verification – A Hidden Markov Approach. PhD thesis, Eindhoven University of Technology, Eindhoven, Netherlands, 1998

82 Senior A. Off-Line Cursive Handwriting Recognition using Recurrent Neural Networks. PhD thesis, University of Cambridge, UK, 1994

83 Rabiner L, Lee GH, Juang BH, Wilpon JG. HMM clustering for connected word recognition. Proc International Conference on Acoustics, Speech and Signal Processing, Glasgow, UK, 1989; 405–408

84 Koerich AL, Sabourin R, Suen CY. A distributed scheme for lexicon-driven handwritten word recognition and its application to large vocabulary problems. Proc 6th International Conference on Document Analysis and Recognition, Seattle, WA, 2001; 660–664

85 Belkacem B. Une application industrielle de reconnaissance d'addresses. Proc. 4eme Colloque National Sur l'Ecrit et le Document, Nantes, France, 1996; 93–100

86 Elms AJ, Procter S, Illingworth J. The advantage of using an hmm-based approach for faxed word recognition. Int J Document. Analysis and Recognition 1998; 1:18–36

87 Madhvanath S, Srihari SN. Effective reduction of large lexicons for recognition of offline cursive script. Proc 5th International Workshop on Frontiers in Handwriting Recognition, Essex, UK, 1996; 189–194

88 Srihari SN, Shin Y-C, Ramanaprasad V, Lee DS. A system to read names and addresses on tax forms. Proc IEEE 1996; 84(7):1038–1049

89 Yaeger LS, Webb BJ, Lyon RF. Combining neural networks and context-driven search for on-line, printed handwriting recognition in the Newton. AI Magazine 1998; 73–89

90 Jain A, Zongker D. Feature selection: Evaluation, application, and small sample performance. IEEE Trans Pattern Analysis and Machine Intelligence 1997; 19(2):153–158

91 Kudo M, Sklansky J. Comparison of algorithms that select features for pattern recognition. Pattern Recognition 2000; 33:25–41

92 Kim G, Kim S. Feature selection using genetic algorithms for handwritten character recognition. Proc 7th International Workshop on Frontiers in Handwriting Recognition, Amsterdam, Netherlands, 2000; 103–112

93 Oliveira LS, Benahmed N, Sabourin R, Bortolozzi F, Suen CY. Feature subset selection using genetic algorithms for handwritten digit recognition. Proc 14th Brazilian Symposium on Computer Graphics and Image Processing, Florianopolis, Brazil, 2001; 362–369

94 Koerich AL, Sabourin R, Suen CY. Large vocabulary off-line handwriting recognition using a constrained level building algorithm. Int J Document Analysis and Recognition 2002, (submitted)

95 Vuori V, Laaksonen J, Oja E, Kangas J. Speeding up on-line recognition of handwritten characters by pruning the prototype set. Proc 6rd International Conference on Document Analysis and Recognition, Seattle, WA, 2001

96 Guillevic D, Suen CY. Cursive script recognition: A sentence level recognition scheme. Proc. 4th International Workshop on the Frontiers of Handwriting Recognition, Taipei, Taiwan, 1994; 216–223

97 Saon G. Modeles Markoviens Uni et Bidimensionnels pour la

Reconnaissance de l'Ecriture Manusctrice Hors-Ligne. PhD thesis, Université Henri Poincaré, Nancy, France, 1997
98 Kaltenmeien A, Caesar T, Gloger JM, Mandler E. Sophisticated topology of hidden markov models for cursive script recognition. Proc International Conference on Document Analysis and Recognition, Tsukuba, Japan, 1993; 139–142
99 Farouz C. Reconnaissance de Mots Manuscrits Hors-Ligne dans un Vocabulaire Ouvert par Modélisation Markovienne. PhD thesis, Université de Nantes, Nantes, France, 1999
100 Lee CK, Leedham G. Rapid analytical verification of handwritten alphanumeric address fields. Proc 7th International Workshop on Frontiers in Handwriting Recognition, Amsterdam, Netherlands, 2000; 571–576
101 Guillevic D, Nishiwaki D, Yamada K. Word lexicon reduction by character spotting. Proc 7th International Workshop on Frontiers in Handwriting Recognition, Amsterdam, Netherlands, 2000; 373–382
102 Kaufmann G, Bunke H, Hadorn M. Lexicon reduction in an hmm-framework based on quantized feature vectors. Proc 4th International Conference on Document Analysis and Recognition, Ulm, Germany, 1997; 1997–1101
103 Koerich AL, Sabourin R, Suen CY. A time-length constrained level building algorithm for large vocabulary handwritten word recognition. Proc 2nd International Conference on Advances in Pattern Recognition, Rio de Janeiro, Brazil, 2001; 127–136
104 Zimmermann M, Mao J. Lexicon reduction using key characters in cursive handwritten words. Pattern Recognition Letters 1999; 20:1297–1304
105 Madhvanath S, Govindaraju V. Holistic lexicon reduction for handwritten word recognition. Proc SPIE – Document Recognition III, San Jose, CA, 1996; 224–234
106 Madhvanath S, Krpasundar V, Govindaraju V. Syntatic methodology of pruning large lexicons in cursive script recognition. Pattern Recognition, 2001; 34:37–46
107 Hennig A, Sherkat N, Whitrow RJ. Recognising letters in on-line handwriting with hierarchical fuzzy inference. Proc 4th International Conference on Document Analysis and Recognition, Ulm, Germany, 1997; 936–940
108 Leroy A. Lexicon reduction based on global features for on-line recognition. Proc 4th International Workshop on Frontiers in Handwriting Recognition, Taipei, ROC, 1994; 431–440
109 Leroy A. Progressive lexicon reduction for on-line handwriting. Proc 5th International Workshop on Frontiers in Handwriting Recognition, Essex, UK, 1996; 399–404
110 Gilloux M. Real-time handwritten word recognition within large lexicons. Proc 5th International Workshop on Frontiers in Handwriting Recognition, Essex, UK, 1996; 301–304
111 Hanazawa K, Minami Y, Furui S. An efficient search method for large-vocabulary continuous-speech recognition. Proc International Conference on Acoustics, Speech and Signal Processing, Munich, Germany, 1997; 1787–1790
112 Kenny P, Hollan R, Gupta VN, Lennig M, Mermelstein P, O'Shaughnessy D. A* admissible heuristics for rapid lexical access. IEEE Trans Speech and Audio Processing 1993; 1(1):49–58
113 Renals S, Hochberg MM. Start-synchronous search for large vocabulary continuous speech recognition. IEEE Trans Speech and Audio Processing 1999; 7(5):542–553
114 Robinson T, Christie J. Time-first search for large vocabulary speech recognition. Proc International Conference on Acoustics, Speech and Signal Processing, Seattle, WA, 1998; 829–832
115 Young S. Large vocabulary continuous speech recognition. IEEE Signal Processing Magazine 1996; 13(5):47–55
116 Fujisaki T, Nathan K, Cho W, Beigi H. On-line unconstrained handwriting recognition by a probabilistic method. Proc 3rd International Workshop on Frontiers in Handwriting Recognition, Buffalo, NY, 1993; 235–241
117 Bippus R, Margner V. Script recognition using inhomogeneous p2dhmm and hierarchical search space reduction. Proc 5th International Conference on Document Analysis and Recognition, Bangalore, India, 1999; 773–776
118 Ford DM, Higgins CA. A tree-based dictionary search technique and comparison with n-gram letter graph reduction. In: Plamondon R, Leedham G, eds, Computer Processing of Handwriting, World Scientific, 1990; 291–312
119 Ratzlaff EH, Nathan KS, Maruyama H. Search issues in the IBM large vocabulary unconstrained handwriting recognizer. Proc 5th International Workshop on Frontiers in Handwriting Recognition, Essex, UK, 1996; 177–182
120 Suen CY. N-gram statistics for natural language understanding and text processing. IEEE Tran Pattern Analysis and Machine Intelligence 1979; 1(2):15–21
121 Lifchitz A, Maire F. A fast lexically constrained Viterbi algorithm for on-line handwriting recognition. Proc 7th International Workshop on Frontiers in Handwriting Recognition, Amsterdam, Netherlands, 2000; 313–322
122 Madhvanath S, Krpasundar V. Pruning large lexicons using generalized word shape descriptors. Proc 4th International Conference on Document Analysis and Recognition, Ulm, Germany, 1997; 552–555
123 Koerich AL, Sabourin R, Suen CY. Fast two-level Viterbi search algorithm for unconstrained handwriting recognition. Proc 27th International Conference on Acoustics, Speech, and Signal Processing, Orlando, FL, 2002; 3537–3540
124 Rabiner L, Juang BH. Fundamentals of Speech Recognition. Prentice Hall, 1993
125 Chen JK, Soong FK. An n-best candidates-based discriminate training for speech-recognition applications. IEEE Trans Speech and Audio Processing 1994; 2(1):206–216
126 Heutte L, Paquet T, Nosary A, Hernoux C. Handwritten text recognition using a multiple-agent architecture to adapt the recognition task. Proc 7th International Workshop on Frontiers in Handwriting Recognition, Amsterdam, Netherlands, 2000; 413–422
127 Stone J, Ercal F. Workstation clusters for parallel computing. IEEE Potentials 2001; 20(2):31–33
128 Hull JJ. Database for handwritten word recognition research. IEEE Trans Pattern Analysis and Machine Intelligence 1994; 16:550–554

Originality and Contribution

The main contribution of this article is in presenting the current state of large vocabulary off-line handwriting recognition and identifying the key issues affecting the future applications. It reports many recent advances that have occurred in this field, particularly over the last decade. Various aspects related to accuracy and computational complexity are first presented to highlight the suitability of different approaches to tackle such problems. We present a case study with large lexicons where some of the methods presented throughout this article are used. Finally, we attempt to predict future issues in the field and the difficulties to highlight the importance of improving the basic components of the handwriting recognition systems to allow acceptable performances in real applications. Its originality comes from the fact that no other survey or article has grouped strategies devoted to large vocabularies, but only some isolated methods and strategies have been reported.

Alessandro L. Koerich received the BSc degree in electrical engineering from the Federal University of Santa Catarina (UFSC), Brazil in 1995, and the MSc degree in electronics and communications from the University of Campinas (UNICAMP), Brazil, in 1997. He is currently a PhD candidate in the Département de Génie de la Production Automatisée at the École de Technologie Supérieure (ETS), Université du Québec, Montréal, QC, Canada. He is also a visiting scientist at the Centre for Pattern Recognition and Machine Intelligence (CENPARMI). His research interests include pattern recognition, handwriting recognition and multimedia.

Robert Sabourin received Bing, MScA and PhD degrees in electrical engineering from the École Polytechnique de Montréal in 1977, 1980 and 1991, respectively. In 1977, he joined the physics department of the Université de Montréal where he was responsible for the design and development of scientific instrumentation for the Observatoire du Monte Mégantic. In 1983, he joined the staff of the École de Technologie Supérieure, Université du Québec, Montréal, QC, Canada, where he is currently a professeur titulaire in the Département de Génie de la Production Automatisée. In 1995 he also joined the Computer Science Department of the Pontificia Universidade Cataólica do Paraná (PUCPR, Curitiba, Brazil), where since 1998 he has been co-responsible for the implementation of a PhD program in applied informatics. Since 1996, he is a senior member of the Centre for Pattern Recognition and Machine Intelligence (CENPARMI). His research interests are in the areas of handwriting recognition and signature verification for banking and postal applications.

Ching Y. Suen received an MSc (Eng) degree from the University of Hong Kong and a PhD degree from the University of British Columbia, Canada. In 1972, he joined the Department of Computer Science of Concordia University where he became Professor in 1979 and served as Chairman from 1980 to 1984, and as Associate Dean for Research of the Faculty of Engineering and Computer Science from 1993 to 1997. Currently, he is the Director of CENPARMI, the Centre for Pattern Recognition and Machine Intelligence. Professor Suen is the author/editor of 11 books and more than 300 papers on subjects ranging from computer vision and handwriting recognition, to expert systems and computational linguistics. He is the founder and Editor-in-Chief of a journal, and an Associate Editor of several journals related to pattern recognition. A Fellow of the IEEE, IAPR and the Academy of Sciences of the Royal Society of Canada, he has served several professional societies as President, Vice-President or Governor. He is also the founder and chair of several conference series, and is currently the General Chair of the International Conference on Pattern Recognition, Quebec City, 2002. Dr Suen is the recipient of several awards, including the ITAC/NSERC Award in 1992 and the Concordia 'Research Fellow' award in 1998.