Online Handwriting Recognition

26

Jin Hyung Kim and Bong-Kee Sin

Contents

J.H. Kim (✉)
Department of Computer Science, Korea Advanced Institute of Science and Technology, Yuseong-gu, Daejeon, Korea
e-mail: jkim@cs.kaist.ac.kr

B.-K. Sin
Department of IT Convergence and Applications Engineering, Pukyong National University, Namgu, Busan, Korea
e-mail: bkshin@pknu.ac.kr

D. Doermann, K. Tombre (eds.), *Handbook of Document Image Processing and Recognition*, DOI 10.1007/978-0-85729-859-1_29,
© Springer-Verlag London 2014

Abstract

Online handwriting recognition has long been studied, and the technology has already been commercialized to some extent. But limited success stories from the market imply that further research is needed on electronic pen interface in general and recognition methods in particular. This chapter describes some of the basic techniques required to build a complete recognition software. At the turn of the millennium, there has been a renewed interest with focus shifted and diversified to multiple languages. Along with this trend, a lot of new ideas and efforts have been made to deal with different characteristics of different scripts. The difference notwithstanding, it should be helpful for designers to revert to the basics and review the established techniques and new ideas developed from afar the field before figuring out new – or maybe not very new – solutions to one's own language. Current big hurdles in online handwriting recognition include stroke order variation and multiple delayed strokes in addition to shape and style variations. This chapter ends with a short list of example systems and softwares, research based or commercial, that have been sort of landmarks or cited more often than not in the field.

Keywords

Delayed stroke • Handwriting recognition • Level-building algorithm • Spatial information • Stroke order variation

Introduction

Although not in wide use even today, online handwriting is still considered a viable interface for future generation mobile devices and text input across many languages. This chapter introduces the basic issues of online script recognition with a special emphasis on differences among scripts and possible recognition strategies thereof. Given that perspective, an in-depth discussion is given around four sources of difficulty in modeling variability of online script patterns.

Overview

The field of online handwriting has a long history dating back to the 1980s with the development of accurate and compact digitizer devices along with improved computing power for real-time recognition [1]. As reliable electronic pens became available, the idea of recognizing online handwriting led to a high hope of introducing a new modal of human-computer interface. This, however, has not been very successful due to unforeseen difficulties of character recognition. This chapter analyzes the difficulties and then discusses practical ideas and techniques that have been developed thus far to overcome those difficulties.

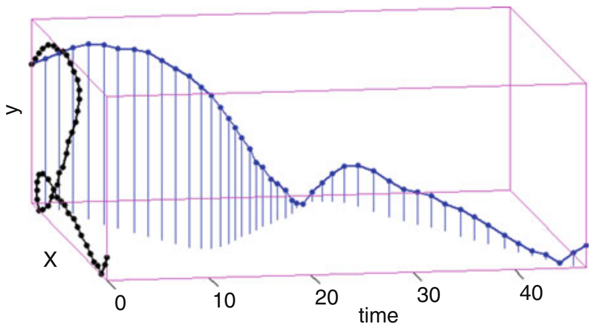

Fig. 26.1 A spatiotemporal trajectory representation of an online handwriting

Up until 1990s the global research interest in online handwriting peaked higher and higher with most research activities targeting the recognition of Latin-based scripts, Kanji or Chinese characters used in Japan, and, to a lesser extent, Korean Hangul characters [2–4]. The research continued on and soon the market began to see a number of commercial products.

More recently we have come to witness a strong renewed interest in this area, this time with applications to a wider range of scripts which include Arabic [4–7], Devanagari, and related scripts in South and Southeast Asia [8, 9].

Since 2007, the popularity of smart-phones and tablet computers has driven the demand and wide acceptance of finger-based touch screens. This, however, have not helped the pen-based interface win users' attention. This can be construed as telling us that online handwriting has its place not in our everyday life but rather in more serious tasks like text editing with pen gestures and in special tasks like Chinese character input or mathematical expression input. For now it remains to be seen how things will develop for future online handwriting recognition.

Online Handwriting

Online handwriting signal is a spatiotemporal signal that is intended to describe a geometrical shape. When viewed in spatial dimension only, it is simply a patterned collection of sample points in the two-dimensional planar surface. With the addition of the time dimension as shown in Fig. 26.1, it turns into a spatiotemporal trajectory of a pen. Those points are captured in a strict sequence; if we link them in order, the handwriting becomes a spatiotemporal curve characterizing the shape, writing order, and speed. One straightforward form of description that completely characterizes the curve would be $(x, y, t), t = 1, \ldots$, or more conveniently, (x_t, y_t) with the same t values. This description is called a time series. Online handwriting is a sequential data that has dynamic information about the order of writing.

The dynamic information is valuable for character recognition since it provides us with the number and order of strokes. These are extremely valuable since they can render the task a lot easier by giving a good starting point: stroke. With the ordered and separate strokes given, we can bypass the difficult step of stroke identification or segmentation and go directly on to concentrate on analyzing the shape of individual strokes and their relations.

Unfortunately, however, the blessing ends there. In English or in many other Western scripts, it continues. But for characters like Chinese, it is half a curse. There are tens of thousand characters in Chinese, and many of them have a lot of strokes, up to two or three dozens. In many of those characters, the order of writing strokes is often not obvious and thus tends to vary. This complicates recognition with additional variations not apparent in offline static images. It is obvious that the temporal information can be used to advantage. But too great a dependence on it may more than nullify the advantage. In this sense the dynamic information of handwriting is a mixed blessing for the designers of character recognizers.

There are times when the violation of the temporal order is unavoidable. In cursive handwritings like Arabic and Western words, a sequence of several strokes usually get connected to form a long stroke followed by zero or more delayed strokes like dots, bars, or even diacritical marks. These are largely script dependent. However, the knowledge of delayed stroke handling techniques or even stroke order-free methods in one script may be worth the effort since one can benefit therefrom for modeling his or her own script.

By convention optical character recognition refers to the offline recognition of character images. It takes an image of words to analyze their two-dimensional shape. An online handwriting data is distinguished from an offline image in that it is a sequential data corresponding to a curve in the spatiotemporal space. Therefore, when the data representation or the recognition technique is concerned, online handwriting is a different problem with entirely different kind of patterns.

But the final outcome of both tasks is the same: text. Hence, it is natural for system developers to think that one can benefit from the other. Since online handwriting has been more successful in both technology development and commercialization than the offline counterpart, some initial efforts were launched to recover dynamic information about the pen trajectory from offline images [10, 11]. But intrinsic ambiguities abound, so the effort was not very successful to date. An integrated approach of tracking pen trajectories based on robust online shape models could be one solution that needs to be tested [12].

Script Classification

Most scripts on Earth today are linear or curved stroke-based system so that we can write characters using a pen. In essence online handwriting recognition is about shape analysis of those strokes given in sequence. The analysis usually proceeds from preprocessing of the input signal and then moves on to extracting effective features in a compact form suitable for shape classification and linguistic decoding. Most of these steps depend greatly on the type of script among others. Cursive scripts need sophisticated ways of modeling curves, while syllable-based characters are free from character segmentation but often suffer from stroke order variations like Chinese. Hence, it will be helpful to know the differences and similarities among scripts and across languages.

Writing systems or scripts can be classified in many ways. But from the perspective of online handwriting, we can try a very simple classification based on the alphabets and the method of composing characters and words. Scripts as well as languages have evolved gradually, sometimes with histories dating back to many millennia ago. Today there are many alphabet-based scripts around the world. Among them, Western alphabets like Latin, Cyrillic, and Greek and Korean Hangul are considered pure alphabets with a large number of users. In Western scripts, the natural unit of handwriting is a word which is written spatially separated from others. It is a linear concatenation of letters from the alphabet and written in the corresponding order left to right, often with delayed strokes. The structure of most handwriting recognizers often reflects the geometrical structure. In fact, historically, most of the initial research efforts have been around those scripts. Therefore, any new online handwriting research should start with the literature thereof.

Korean Hangul is a syllable-based writing system where people write a word as composed of several characters written left to right or, at times, top to bottom. Each character is a composition of two or three letters called graphemes: consonant(C) + vowel(V) or C + V + final consonant(C). This composite character represents a syllable. A big difference from the Western word is that the letters are arranged two-dimensionally inside a virtual square. The spatial relation among component graphemes is very important, if not critical, for readers. Therefore, spatial relation modeling has been an important issue in Hangul recognition. The good news here is that the writing order of graphemes is highly natural that there are few exceptions; C + V or C + V + C.

Another script that is written usually horizontally like the Western scripts is Arabic, an abjad. Although written right to left, an Arabic word is a horizontal sequence of subwords, each consisting of a few Arabic alphabets with varying shapes depending on their position. Just like Hangul, the basic modeling unit of recognition could be such a subword representing a syllable. The overall recognition architecture can be similar to Western words but with many more delayed strokes.

If we move a little east from Southwest Asia or the Middle East, we come across a huge population using another syllabic script called Devanagari. Linguistically speaking, it is an abugida, a similarly consonant-based alphabet but with vowels explicitly indicated with diacritics or systematic graphic modification to the letters. Many other scripts in the Southeast Asia are also abugidas. All of them are believed to have descended from the ancient Brahmi script. Therefore, researchers in those scripts could benefit a lot from the knowledge of the others.

Unlike the previous scripts, Japanese kanas are syllabaries where each character represents a complete syllable with a different vowel. There is no composition of new sounds like the above alphabets; hence, they are not classified as an alphabet. Anyway, as the number of distinct characters is small, kana character recognition appears to be relatively easy. Hence, most Japanese scholars devote their effort to the recognition of kanji, a set of Chinese characters used in Japan [3]. The techniques developed for kanji can be readily applied to Chinese character recognition and vice versa.

The Chinese character is quite distinct from others in the pattern recognition perspective. First of all it is a logosyllabic script. Each character has a meaning just like a word. But like many other syllable-based scripts, each character is written separately within a virtual bounding square. At first glance, the biggest feature would be that there are so many strokes. In many cases some of the strokes are grouped to form a stereotyped cluster called a radical. There are hundreds of distinct radicals in Chinese, and most of them are legal character in their own right. They can be combined to form a more complex character, leading to a hierarchical organization within a character. Apart from the complexity, the stroke order variation in handwriting stands out from among the daunting issues in Chinese character recognition. Other issues include modeling increasingly cursive shapes and complex spatial relations among the numerous strokes and radicals. It is a formidable task. Let us keep a close watch on how our Chinese and Japanese colleagues are doing.

Organization of This Chapter

The organization of this chapter is made simple. First, section "Preprocessing" presents several tools of preprocessing. Then section "Feature Extraction" gives a brief description of feature extraction. The next section comprises the largest proportion of this chapter. It is subdivided into four subsections detailing aspects of difficulties in online recognition in general. The remaining sections describe example systems known in the literature and market and then conclude this chapter.

Preprocessing

Online handwriting contains a range of sensor noise originated from the limitation of resolution grid and digitization process, mechanical switch of the pen, as well as the erratic hand motion [1]. These are often harmful to the recognition process. Hence, it is desirable and usually necessary to remove those factors before starting the actual computation. Additionally for practical reasons, we try to isolate units of recognition, which can be a character, an alphabetic letter, a stroke, or even a word (with multiple strokes) based on any shape features. The shape features are usually domain specific and require the knowledge about the target pattern class. But this kind of segmentation is sometimes applied before the recognition. Thus, a brief discussion is apt here.

Segmentation

Segmentation in online handwriting refers to the isolation of scripts into recognition units such as words or characters, or even strokes. If this is done without knowledge of the target class, it is called external segmentation. Segmentation requiring recognition is called internal segmentation [1].

The simplest means of external segmentation is one done by the user. One natural method is giving a spatial clue where the user writes the next character well separated spatially from the last character of the previous word. Alternatively it is also possible that the user stops writing and then, after a time-out, the system starts to analyze the shape. This can be termed temporal clue for word segmentation, unavailable in offline character recognition.

In pure syllable-based writing systems like Chinese, Hangul, and Devanagari, each character corresponds to a syllable and written separately, thus removing the need for character segmentation. In this case the temporal clue is convenient, while the spatial clue is not obvious. This is especially the case in sequences of regular block characters where there are many vertical strokes and short strokes. Therefore, external segmentation in a character string is commonly performed by dynamic programming-based lattice search [3, 4].

Noise Filtering

Noise filtering involves various signal processing techniques that range from removing obvious random noise to reducing various redundancies. Typical random noise in online handwriting includes wild dots causing random jumps in otherwise smooth strokes. They are caused by hardware problems. Using a simple heuristic about physical limitation in abrupt hand motion, we can remove points of high acceleration and sudden change of direction.

Perhaps the opposite form of wild point includes dot loss causing brief loss of pen trajectory or extraneous pen lifts resulting in broken strokes. In the former case, we can introduce a number of intermediate points by interpolation depending on the size of the gap. A typical gap filter uses the Bezier curve [13]. In the latter case when the gap between the end points of the two strokes is small relative to the size of the character, we can simply connect the two strokes into one.

A related filtering technique is data reduction which refers to reducing redundancies in the input signal. Depending on the speed of writing, a sequence of points are often redundant implying no or little hand motion. Data reduction or dot reduction refers to discarding all of them except one.

A more common filtering technique is smoothing where a point in a stroke is replaced by a weighted average of its neighbors. This helps reduce small random noise and jitters embedded in the input signal. This type of filtering is useful when the input handwriting is small relative to the resolution of the digitizer.

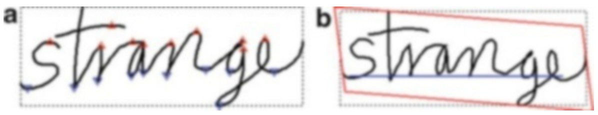

Fig. 26.2 (**a**) Local maxima and minima are marked by *upright* and *inverted triangles*, respectively, (**b**) the baseline aligned and characters deskewed by rotation and shearing transformation

There is an additional form of noise called a hook. A hook is an unintended part of a stroke at the start or the end of an intended stroke body. It is caused by an inaccurate pen-down or lifts and usually occurs at a sharp angle to the body stroke, thus called a "hook." The hook can be interpreted as part of the pen-up motion between two strokes which happened to be registered by earlier pen-down or late pen-up motion. The hook can be considered as a kind of noise and removed. But it can also be modeled as a legal pattern of a ligature in natural cursive handwriting [14].

Normalization

Handwriting normalization refers to the conversion of input handwriting to a canonical form that is free from the influence of sensor devices, writing habits, and environments. The primary goal of normalization is reducing shape and other geometrical variances in order to derive invariant features which are relevant to recognition.

The first and most obvious form of normalization is a linear transformation which includes scaling while preserving the aspect ratio, and rotation and shearing. Rotation is often called slant correction where the drifting baseline of the input handwriting is made straight and aligned to the horizontal line. The key is finding the baseline and midline. A typical technique of detecting a baseline is finding the local minima and then applying linear regression or Hough transform [13]. In the case of Latin-based words or Arabic, we can obtain both the baseline and the midline which are more reliable by constraining the slope equal to the midline aligned to the local maxima. Unfortunately, however, this does not work well for short words and often go awry in the presence of spurious extrema.

Individual characters are also written skewed. A typical deskewing method uses a heuristic of shearing the shape back to an upright form by noting the statistics of skewed region of strokes. See Fig. 26.2. This step is critical for assigning delayed strokes to the letter in the correction position.

Another normalization technique is resampling feature points each of which lie at a uniform distance from the immediately preceding point. This tries to remove the variation in writing speed throughout the entire handwriting and possibly across different handwritings of different scale in different environment. In handwriting the shape of the script is largely determined by curve bends, particularly sharp corners called cusps. So those points are usually retained in the transformed samples.

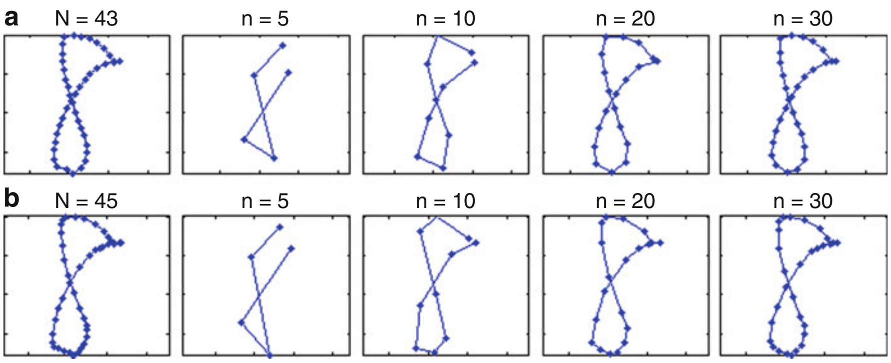

Fig. 26.3 (**a**) Resampling in space for equidistant points, (**b**) resampling in time for equal travel time between points, obtained by Fourier transformation

Resampling is still a very common technique employed regardless of language [1, 3, 4]. It is useful as a means of data reduction, too. But the problem is that the loss of points is great around cusps and corners. Therefore, the spatial resampling could mean a loss of information in the temporal dimension. Hence, it is not always recommended to apply resampling to natural handwriting. Figure 26.3 shows examples of resampling in space and time by varying the number of resampling points.

Feature Extraction

Feature extraction refers to the process of converting a raw data into a set of measurements in a form as required by the recognition algorithm. Despite intensive research during the past couple of decades, there has been no consensus yet about common or universal features that can be applied to any scripts. But it is not that bad because we are left with a rich menu of general, intuitive features that can be useful for diverse scripts.

Perhaps the most obvious kinds of features are local geometrical features like point coordinates, pen-down/up states, and pressure depending on the device. Some of the slightly more advanced and informative features are called "delta features" that include tangential direction or angle at sampled points. They can be described by direction code or by a continuous density function like von Mises or wrapped Gaussian [15, 16]. In addition, the local curvature and the writing speed can be expressed in the feature vector. These features are often robust and reliable in pattern description.

If a more meaningful and intuitive features are desired, we can derive high-level shape features like cusps, t-crossings, loops, and, in the case of Latin and Arabic words, ascenders and descenders. These are powerful by themselves telling us a lot about the overall shape, but are difficult to detect and include into the sequential computation structure. They are by nature more of offline features than online.

Since local features generally do not capture the bigger picture of the handwriting, some form of segmental or regional features have been tried in most recognition systems. They include delayed strokes and related methods, Fourier descriptors describing an oscillating cursive stroke, regional configuration around a sample point often expressed as a small bitmap called a contextmap [13], and spatial relations to neighbor strokes [17, 18]. These features are usually language dependent, and thus, their use can be quite limited. Furthermore it is not obvious how to represent and evaluate them effectively.

A stroke in cursive online handwriting is often a connection of several strokes followed by shape distortion into a smooth winding curve. Depending on the modeling unit in the recognition stage, it is often necessary to segment the stroke into a sequence of basic sub-strokes. Common techniques of stroke decomposition involve the detection of feature points in a stroke, such as vertical or horizontal maxima and minima or simply cusps and points of great curvature.

Still another method dealing with a stroke is to take the whole stroke as it is and label it as an independent entity [4]. This idea has occasionally been tried in many studies. In one extreme, the label could be a simple code in some codebook, and a handwriting script can be described as a sequence of stroke codes. This idea makes the system very simple and works well for syllable-based regular scripts. But the problem is that it is always possible to encounter a new stroke belonging to none of the codewords. One popular technique is to use the stroke data as a template and then use the elastic matching algorithm for stroke or character classification. It is effective for small alphabet or vocabulary recognition tasks. The best example is Graffiti symbol recognition to be discussed in the next section. Note that the discussion has now crossed the boundary from feature extraction to recognition.

Recognition Methods

Given a set of features, the recognition method in online handwriting is more or less determined, or vice versa. Here the method refers to the type and structure of pattern models and the associated classification algorithm. In this section, we assume an online handwriting is given as a time series, a sequence of local vectors. Each feature vector can be as simple as an (x, y)-coordinates.

If the online handwriting signal were a pure time series with only local information, the recognition model would be very simple. It is because some sequence of local decisions would exactly correspond to the global decision. Note, however, that a time series signal with strictly local features is often just a boring random noise of no one's interest.

In many real online handwriting signals, the feature vectors are largely correlated. Hence, most recognizers today try to incorporate regional or global features which are nonsequential and thus considered as static or offline features. The amount and extent of correlation depend on the type of patterns and scripts. In order to avoid getting bogged down in details, let us again assume that the information is captured and represented in the sequence of feature vectors. The information so captured could be incomplete but should not be missed entirely.

A recognition model is a principled representation of the temporal information in real signals. And the associated recognition method should provide the computation of the model given the input data. However, there is no panacea for all scripts. Languages have different character sets or alphabets with different spelling or writing rules. Still, however, most scripts share many linguistic features; hence, it is not surprising that we can learn a lot from other languages. In fact all scripts around the world are neither completely random nor truly independent, but rather they are the products of human minds that have coevolved over a few millennia.

Let us first consider a very simple case of digit recognition before digging deeper into the issues of online handwriting recognition. The following discussion is highly educational providing an insight into the nature of handwriting recognition in general. In addition we will assume the HMM as the basic modeling tool.

Simple String Recognition

One archetypal example is the recognition of a numeral written as a digit string as shown in Fig. 26.4. In general digit strings, one digit does not affect the occurrence of other digits. Hence, the digits are essentially independent. For instance, let us consider the problem of evaluating a model of digit string $W = W_1 W_2 \cdots W_K$ given an input sequence $X = X_1 X_2 \cdots X_T$. In such a case, we can replace the overall problem of evaluating the pair $score(W, X)$ by a sequence of local computations $score(W_k, X(k)), \quad k = 1, \ldots, K$, where $X(k)$ denotes a subsequence mapped to k-th digit model. When the probabilistic function is used for the evaluation, we can rewrite it as

$$P(W, X) = P(W) \prod_{k=1}^{K} P(X(k)|W_k)$$

Now we can devote all our efforts to a sequence of smaller problems of computing simpler functions $P(X(k)|W_k)$ representing the conditional density of the sequence $X(k)$ given the model W_k. This deceptively simple function requires three kinds of missing information: the number of digits K, the identity of models W_k, and the boundaries of $X(k)$, $k = 1, \ldots, K$. The solution to this task comprises the core of digit string recognition.

A numeral or the digit string thereof consists of a sequence of random digits. See Fig. 26.4. Each digit can be any of the ten possibilities. Then the string is nothing more than a sequence of digit choices. The model architecture for this type of tasks is shown in Fig. 26.5, which is well-known and often employed as a baseline model in the literature [14, 19].

It is a direct translation of digit choices into a concatenated sequence of model choices. We can duplicate the choice an arbitrary number of times for arbitrary numeral recognition. Here we choose HMM as the model for individual digits.

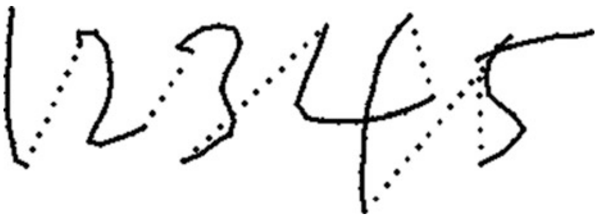
Fig. 26.4 A digit string

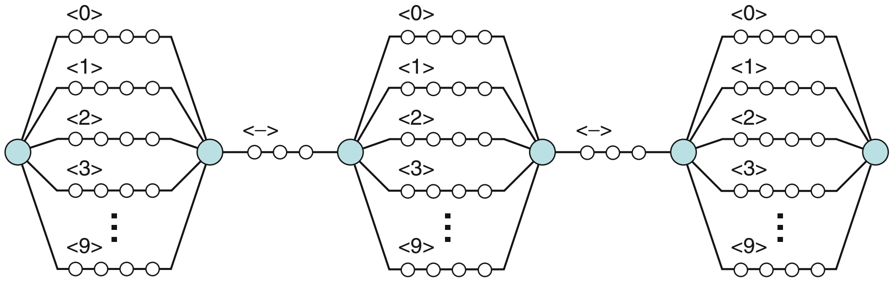

Fig. 26.5 Digit string net for string length

Then the resulting model becomes a network of HMMs. This way of modeling complex patterns by a combination of simple units is so general and useful that it can be applied to a wide range of dynamic signals when the locality conditions are met [14].

Given the network and an input observation sequence X, we are given a problem of optimization over the triples: the digit sequence $W = W_1 W_2 \cdots W_K$ of unknown length K with the segmentation of input sequence $S(X) = X(1)X(2) \cdots X(K)$. The problem of finding the optimal value of the triple (K^*, W^*, S^*) is often called model decoding. Fortunately there are already several algorithms available [20–22]. Among them the level-building algorithm has been chosen here to obtain the optimal digit string that can be recovered by backtracking the forward likelihood computation chart as shown in Fig. 26.6.

Each horizontal band of ten or one scan lines corresponds to a level, starting from the top. So the curve of downward staircase tells us that the first pattern is digit one. It is followed by a between-digit pattern called a ligature in the thin band, which is an imaginary inkless stroke. Here the intensity of each small block of pixels corresponds to the likelihood score of the final state of a digit or ligature HMM. The brighter, the greater score. So each single horizontal line represents the history of input sequence annotated by the corresponding model at the corresponding level. It would rise and fall depending on the current features and the preceding digit. The score must be the maximum in the sequential context to be selected as an output candidate.

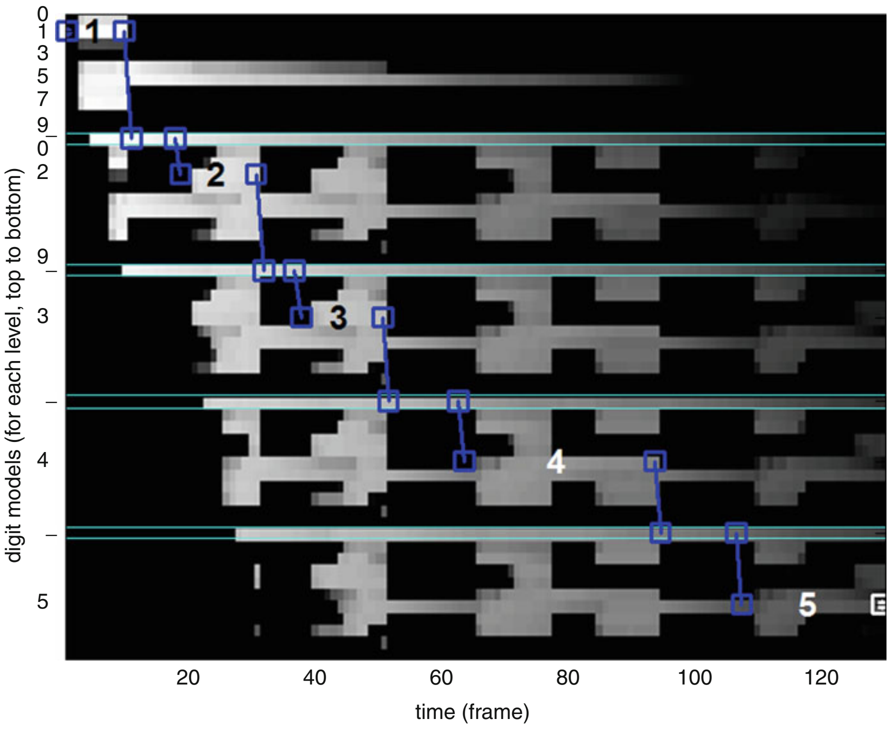

Fig. 26.6 Level-building chart for the input of Fig. 26.4. Each of the nine wide or narrow bands represents a level in sequence from *top* to *bottom*. There are ten digit models (from 0 to 9) in the wide bands, while only one ligature model in the narrow bands. The best model and the best (and brightest) segmentations in each band are marked inside the lattice by *numbers* and *square markers*. The sequence of digits "12345" as shown in the chart is the best candidate

With M HMMs and given a T-long sequence, the amount of computation involved is $O(MN^2 \times TK)$ where up to K digits are expected. Accordingly, the complexity is already high. Fortunately there are heuristic measures to cut it down to a practical level, such as beam search, maximum string length, and slope constraints in the chart. In Fig. 26.6, we assumed that $K \leq 5$ for the sake of illustration. The best number of levels should be determined simultaneously in the same computational framework.

In digit string recognition, the amount of computation can be reduced by the factor of K if the number of digits and digit boundaries are known. Graffiti (see section "Commercial Recognizers") is a typical example where every character is written in a single stroke and there is no connecting successive strokes. Figure 26.7a shows a sample word written in Graffiti. In this case the number of strokes is the number of characters, and the end points of each stroke are its boundaries. Furthermore, we need not care about the between-character pen-up movements, and we can even "overwrite" the characters over the previous ones.

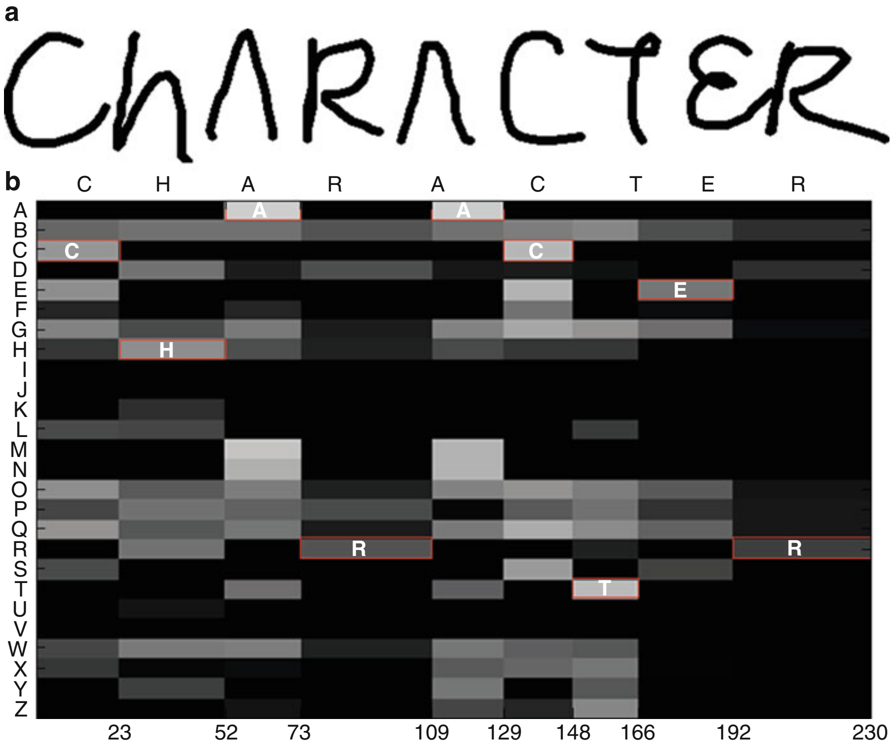

Fig. 26.7 (**a**) a Graffiti sample and (**b**) a computation lattice

Note that Graffiti word recognition is in fact an isolated character recognition problem. The classifier is merely applied to each stroke separately. Each column of Fig. 26.7b shows the matching score of the models given a stroke, where brighter cells represent a greater matching score for the corresponding model. There is only one classifier, which can be realized by any of the known methods, such as template matching, and neural network, in addition to the HMM.

Handwriting Recognition Techniques

It is well known that the HMM is a very useful tool for modeling temporal variability of dynamic signals. So the HMM with the dynamic programming search algorithms developed for the HMM could be the tools of choice. But the problem is not so simple when it comes to online recognition of most scripts beyond the "boring" digits. Hence, the rationale for all our efforts even today.

Here is a list of some major sources of difficulty in online handwriting that includes:

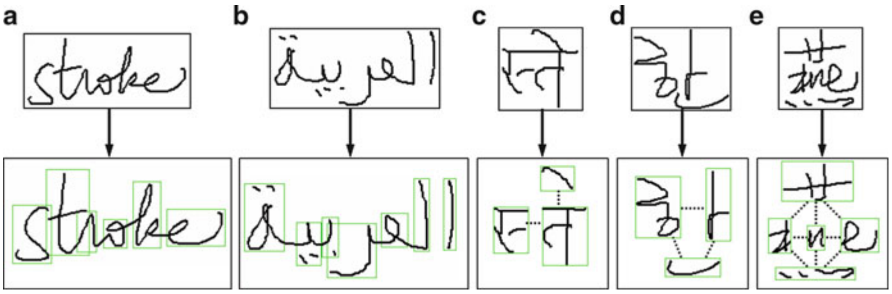

Fig. 26.8 Handwriting structure analyzed (**a**) an English word; (**b**) an Arabic word; (**c**) a Hindi character or part of some word; (**d**) a Korean Hangul character, also part of a word; and (**e**) a Chinese character, also part of a word

- Characters has a spatial structure, and there is a global shape which is not easy to capture with simple local feature measurements.
- Stroke order variations and delayed strokes which complicates the sequential processing of signals.
- Stroke connection in fluent and fast script, and subsequent shape distortion and ambiguity.
- Large vocabulary or character set that can strain even powerful multi-core processors.

Let us explore each of these issues in turn in a wide perspective.

Modeling Spatial Structure

A character, be it machine printed or handprinted, is by nature a spatial pattern that has a certain shape on a two-dimensional plane. In online handwriting, characters are "drawn" sequentially stroke by stroke, represented as a time sequence of point coordinates, and then expressed as a sequence of feature vectors. The problem is that these feature vectors represent only local features of the strokes and fail to capture the overall shape of the characters. Even though the global information is contained in the sequence implicitly, it is not captured well in most character or shape pattern models due to their basic modeling assumptions and structural reasons. For example, the well-known first-order Markov assumption makes it impossible to tell what happened before a single clock unit. But character strokes or their features are correlated far and wide. This is known as "the long-range correlation" problem that has challenged numerous researchers around the globe. How do we solve this problem? Let us look at Fig. 26.8.

Western and Arabic words. Although there are differences, most Western words and Arabic scripts are similar in that they are written linearly in one direction, left to right or right to left. Frequently there are delayed strokes which complicate sequential processing. Arabic scripts just have more delayed dots above or below the body strokes. Characters differ depending on the presence and position of delayed dots. There are optional diacritics for marking vowels, especially in print documents. Since, however, Arabic describes a sound by composing a few letters, one may benefit from the knowledge of syllable-based alphabetic writing systems like Hangul.

Devanagari characters. Basically it is a syllable-based character writing system. There are some 48 alphabets mostly for consonants. In addition there are diacritical marks to denote vowels and rules of composing two or more characters. Figure 26.8c shows a composite character that combines two basic Devanagari alphabets. There are three components; the two lower components represent the combination of sounds s(a) + ta into "sta," while the diacritic in the upper block specifies the vowel change to "e." Thus, the resulting character represents "ste." What is noteworthy is the appearance of two-dimensional structure unlike Western words. Unlike Arabic the diacritical marks are mandatory.

Korean Hangul is a phonetic writing system based on a pure alphabet with 24 letters including separate vowel symbols. Its basic writing unit is a character which represents a syllable. Each character combines two or three letters arranged inside a virtual square as in Fig. 26.8d. These features have been incorporated into the structure of the cursive character model [14].

Also known as a featural alphabet, Hangul has a very simple character composition and writing rule allowing us to spell any sound as it is heard. Figure 26.8d shows a sample character with three letters **C**(consonant, left) + **V**(vowel) + **C**(below), combined into a box (/h/+/a/+/n/ = /han/ where /a/ is said like "a" in "palm"). There are only six different arrangements determined by the shape of the vowels. The **CV** or **CVC** order is so natural that there are few exceptions in Hangul handwriting. Stroke order variations within a letter are not rare, though. So the HMM-based method, aka BongNet, has been very successful for cursive Hangul character recognition [14]. In this method, the ligature modeling has first been tried to represent ligature parts in cursive strokes and the direction of movement or the spatial relation between strokes and letters. According to this method, the character in Fig. 26.8d is described and to be decoded as

$$/\mathrm{h}/ + / - /^{5,7} + /\mathrm{a}/ + / - /^{11,16} + /\mathrm{n}/$$

where the superscripts denote the ligature identifiers as used in Fig. 26.9 with an appropriate node numbering. BongNet models the spatial relation implicitly using a set of separate HMMs and applies a frame-synchronous version of Viterbi algorithm which builds levels while making node transitions to the right [21, 22].

The spatial structure of a character is very important, if not critical, in Hangul. Then we can even choose to focus only on modeling spatial relations between various components in a Hangul character [17]. Given a sequence of strokes, we can divide each stroke into halves like Fig. 26.10a. The two segments have a certain spatial relation called a within-stroke relation (WSR), which can be modeled by a small Bayesian network as shown to the right with three nodes. The stroke can further be divided recursively until some covariance condition is met resulting in a proportionately complex network as shown within each of the square blocks of Fig. 26.10b. On top of these, the relationships between strokes (ISR or inter-stroke

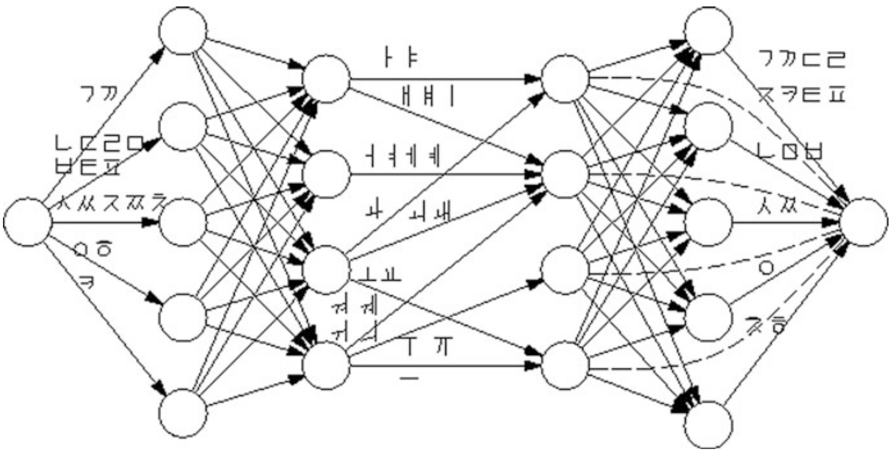

Fig. 26.9 Online Hangul character model BongNet. There are six columns of grammar nodes including the start node on the *left* and the end node on the *right*. They are numbered in order left to right and top to bottom. The leftmost five arcs represent consonant types, and each arc is labeled by a group of consonant letter HMMs. The mesh of arcs between the second and third columns of nodes is labeled by an HMM for a particular ligature shape

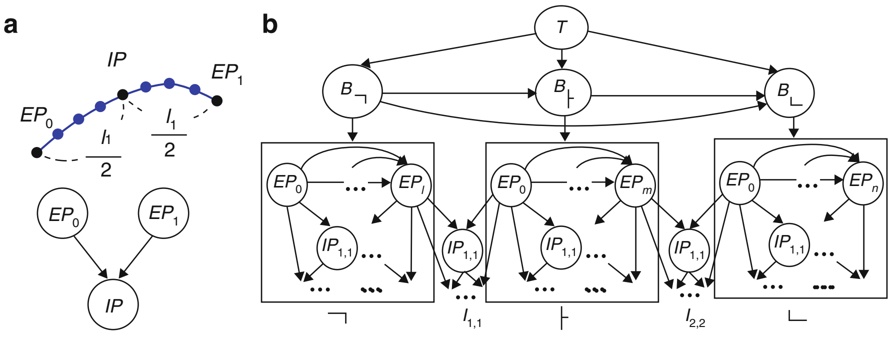

Fig. 26.10 Bayesian network-based modeling of (**a**) feature point relation in a stroke, and (**b**) inter-letter and inter-stroke relation in Hangul character /gan/

relation) and between graphemes (IGR or inter-grapheme relation) can similarly be modeled by a Bayesian network over graphemes. This method could be most effective for regular scripts where strokes are written discrete in scripts like Hangul and Chinese characters.

Chinese character and its variants like Japanese Kanji and Korean Hanja go to the extreme of exploiting the spatial relations among character components called radicals [4]. There are numerous radicals, as many as 300, in the first place. The sample character in Fig. 26.8e has five radicals. Each radical or its variant is an independent character in its own right. We can create a character by combining two or more radicals and arranging them inside a 2D rectangular block with an appropriate scaling. Unlike Hangul character, the spatial composition in Chinese character is somewhat arbitrary if not completely random.

One of the most distinguishing features of Chinese character is the hierarchical composition of radicals, with a complex multi-radical character at the top and a number of individual strokes at the bottom. Those components at any level are arranged spatially and can comprise a different character with different arrangement. In online handwriting, this kind of spatial information is no more local than the shape information is. It is not easy to represent it in sequential feature vectors. For this and other reasons, many researchers have turned away from the HMM for more static and structural models.

The structural information in general implies the global shape where, typically, some feature points are separated apart in time but are close to each other geometrically and thus related. This kind of long-range correlation is not easy to capture with sequential feature vectors. Besides, the next issue of writing order variation has led us to almost forgetting the dynamic information altogether and rather relying on static offline features. For this reason, Chinese character is special as far as its recognition method is concerned. Despite numerous research efforts, there is no consensus as yet about established features and recognition methodology. Investigations are still ongoing. For instance, we can extract primitives at the level stroke or sub-stroke and evaluate them using a distance function or DP matching or even HMMs. Then we can employ an attributed relational graph (ARG) or a decision tree to analyze the hierarchical structure and between-component relationships [23]. In Chinese character recognition, the problem is further complicated by a substantial amount of variations in writing order.

Finally, if we extend the target to a sentence or a sequence of characters, we have to deal with detecting character boundaries. Surprisingly enough, even in character-based writing, this is not so simple because character boundaries often become ambiguous as the number of strokes increases. A popular solution is to apply the dynamic programming search again that evaluates all possible stroke combinations and generates the best while consulting a dictionary and other linguistic information.

Modeling Stroke Order Variation

The dynamic information of online handwriting has been perceived to be useful by default, often leading to the view that the recognition problem would be easy without challenging issues. But an appropriate response out of experience would be "It depends" or "Yes or no."

As remarked earlier, the dynamic information of online handwriting can be a double-edged sword. First of all, it gives the number and the order of strokes, and the direction of stroke direction. Definitely these are valuable and can be used to advantage, as long as they do not depart from the norm. But there are always exceptions and variations, which come in two forms.

The first one is delayed writing of strokes. Examples are found in i-dots, t-bars, and x-crosses in Latin alphabet-based scripts. See Fig. 26.11. They can be detected using simple heuristics like their relative size and position with respect to the last

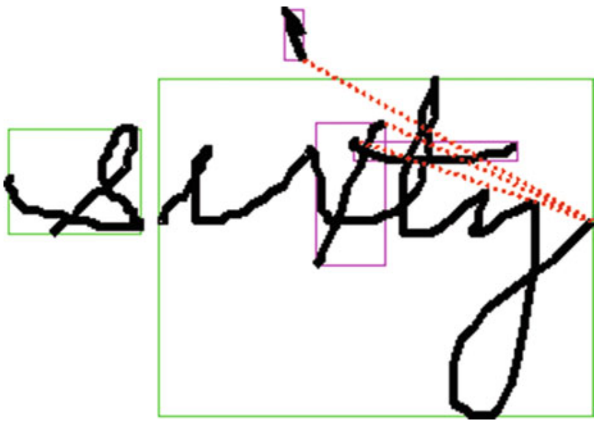
Fig. 26.11 Delayed strokes. They are written to the left, opposite to the writing direction, with respect to the end point of the body stroke

body stroke and its end point. Once they are identified as delayed strokes, they are removed from the normal input sequence of body strokes. Here we introduce three kinds of letter models for "i," "j," "t," and "x" with three possible writing orders. For example, the body stroke of "i" can be written after a dot, or immediately before a dot, or long before a delayed dot. The first two cases pose no problem. Only for the third case do we have to design a dot-free model and include a postprocessing step that follows the forward search like the string recognition of section "Simple String Recognition." When a dot-free "i" pattern is detected from a string search, then we reevaluate the segment including any nearby dot that has been left out previously.

One final remark about delayed stroke processing is that, although cumbersome, you can eventually jump clear the hurdle quite successfully in the case of Western and quite possibly Arabic scripts. There are some important similarities in Arabic scripts. Although written right to left with many delayed dots above or below, we can apply a similar method, mutatis mutandis, to recognize Arabic handwriting.

The second form of stroke order variation concerns a direct challenge to the basic assumption in online handwriting recognition. This is best exemplified by the stroke variations in Chinese characters. There are general rules in the order of writing strokes in a character. Children do learn and practice them. But over time they form their own style while forgetting or breaking the rules. The great number of strokes in many characters and their complex shape may be contributing to the degradation.

A Chinese character in general has a lot of strokes which are arranged in a complex way within a bounding block. Naturally people find different ways and orders of writing and gradually harden it into their own writing style. The order variation is arguably the greatest issue that has long harassed Japanese and Chinese researchers. In fact, because of the frequency and variety, many have given up using the HMM framework, opting for more classical tools like template matching and nearest classifier [3, 4].

Three approaches are possible to resolve order variation in Chinese characters. The first method is to limit the variation only at the level of radicals [3]. Radicals are found at any levels in the hierarchy of character structure, and their size and shape can vary depending on the position in the hierarchy. By applying a suitable affine transformation to each of the canonical radical models, a whole character model can be built by consulting a character dictionary. Note that a character is viewed as a two-dimensional layout of several radicals scaled, distorted, and translated into a square. This is compared to a Hangul character or even a Western word, only with such a big alphabet. Here one observation or, more precisely, an assumption is that the number and the complex organization of radicals in a character frequently lead to order variations among those radicals, but not within individual radicals. Although not without exceptions, this assumption seems reasonable that it is considered to be worth trying [3].

The second method is going to the extreme of simply forgetting the stroke order [4]. Still, however, individual strokes are identified and thus can be used to advantage, much greater than in offline characters. The problem can be approached in two steps: stroke shape analysis and stroke relation analysis. Stroke analysis involves analyzing the structure of a stroke and identifying its type. Any of the traditional methods can be employed including template matching and neural network. Based on the result, we can analyze the spatial relations between strokes and radicals, with one option being the attributed relational graph as referred to in section "Modeling Spatial Structure." This second step could be similar to analyzing spatial relation after stroke extraction. Unfortunately, however, it is rather (doubtful) if there is much to gain from the conversion to offline recognition problem. Although offline task has been studied a lot, we have yet to find a breakthrough there. Therefore, it would be more desirable to stay and play within online recognition framework for now.

Still another method is searching the best stroke correspondence to reference models. There is an interesting effort named cube search that tries to find the best correspondence between strokes in input to references [24]. It will most likely be useful for discrete regular scripts.

The basic assumption in stroke order resolution is that people write regular or less cursive scripts. In normal and natural handwriting of characters or words in most languages, people often write cursively connecting strokes and subsequently distorting the overall shape. This is the third source of difficulty in handwriting recognition.

Modeling Cursive Strokes

The third point of difficulty in online handwriting recognition is the great variability in cursive writing. Everyone has his or her own writing style developed gradually over time since the first grade at school. The style is the result of optimizing the motor skill of the hand and arm [25].

As one masters handwriting, strokes become smoother and rounder, entailing considerable shape distortion insofar as the characters do not get ambiguous. At first, two or more strokes written in succession get connected. Simply without pen-lifts

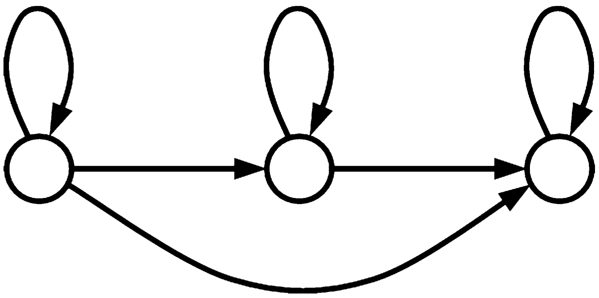
Fig. 26.12 Ligature model topology

or with pen-down drags, the strokes are connected into one. In the latter case, the added part of the stroke is referred to as a ligature [14]. In either case, once strokes are linked, the pen motion is constrained physically and psychologically and causes substantial shape change. Except for the originally cursive scripts like Arabic and cursive Latin, the amount of distortion is highly dependent on the connection and the degree of writing skill. This being the case, it is necessary to model the ligature patterns explicitly.

For instance, it is strongly desired to introduce a ligature model if a linking pattern between two letters or characters occurs frequently. Rejecting them or discounting them as mere nuisance or simply a variable part of a character means avoiding the reality and consequently makes the pattern modeling more difficult.

One method of modeling ligatures is adapting the idea of triphone which is popular for continuous speech recognition [26]. It is in fact a context-dependent model where each phoneme is assigned multiple models each of which is distinguished by its left and right context. Of course, the number of models will grow geometrically and render training a lot more difficult. But we can reduce the number down to a practical level by clustering the two end contexts. The resulting model with context implies a ligature in online handwriting. The difference is that the script ligature model is not part of a character but a separate entity representing a combination of the right and the left contexts of two successive letters [14].

We can define a number of distinct ligatures based on the context and shape. Those ligature patterns appear linear or almost linear with characteristic direction. Each type of ligatures can be modeled using an HMM. One big advantage of using HMM is that both pen-down patterns and pen-up virtual patterns can be modeled in one homogeneous framework. Figures 26.5 and 26.9 show two examples of handwriting pattern model with a set of ligature models embedded in alternate levels. Considering the simple shape but variable length of ligatures, we can design an HMM with a very simple topology as shown in Fig. 26.12. The number of states can be adjusted, if desired, depending on the pattern length statistics.

Let us look at the picture in Fig. 26.6 where the even numbered levels are assigned a single ligature HMM. In the digit string of Fig. 26.4a, there is no pen-down ligature stroke, and all the pen-up ligature motions are not observed in the input except digits 4 and 5. Thus, the activity of digit HMMs in the ligature section is almost nonexistent (of course except digits 4 and 5).

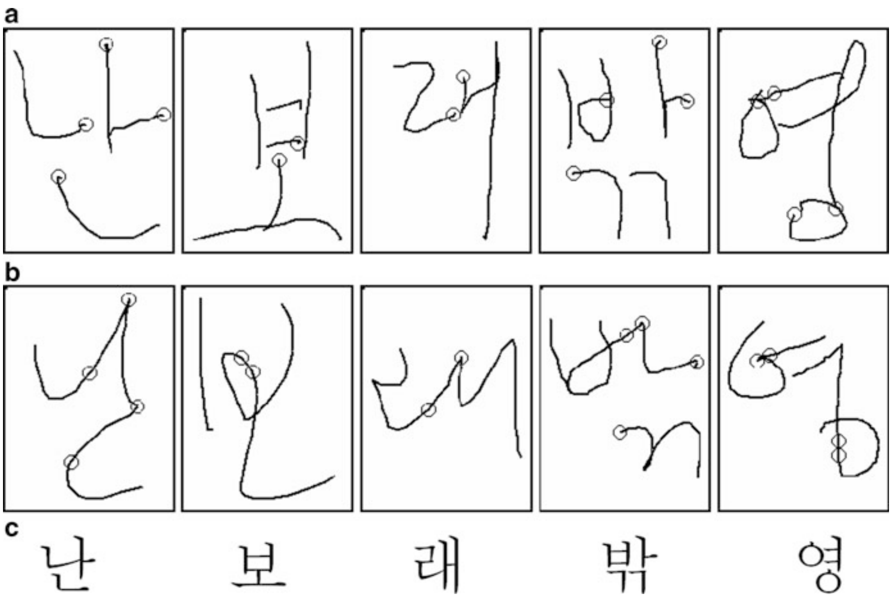

Fig. 26.13 Letter segmentation examples of five characters, discrete (*upper*) and cursive (*lower*), and labels at the bottom. Note that many segmentation points are on the smooth regions of strokes. It is almost impossible to detect them by heuristic methods only due to lack of prominent features around them and shape variability in free, cursive handwriting. But they have been found by the HMM-based recognizer of Fig. 26.9 computing simultaneous recognition and segmentation. There are many ligatures which are usually short, some very short, and linear

When it comes to cursive handwriting, we can distinguish some scripts which are not developed by personal skill but evolved out of history. The pure cursive script in English and the grass script in Chinese are two examples. They are so different from regular block characters that it would be better to regard them as a different script with distinct alphabets or character sets. To recognize them, developing certain forms of functions for handwriting distortion transformation could be an idea but it would not be successful in covering all kinds of nonlinear shape variations. In fact we have no special reason to object to studying hand motor and writing behaviors [25]. But the right and more effective way of doing research in online handwriting recognition will be directly modeling the shape features and their variability from a given dataset.

Character Set and Large Vocabulary

As a means of communication, most modern languages have tens or even hundreds of thousand words in their vocabulary as well as grammatical rules, for naming or describing physical objects, abstract concept, relations, phenomena, and actions.

These words are the basic targets of handwriting recognition. In some cases like logographic Chinese, they could be characters. For practical real-time recognition of these words or characters, it is imperative to reduce the computation and space.

The large character set of Chinese poses some practical difficulty for character recognition. Although it may not be as hard as that of stroke order variation and shape variability, it has long been an important topic in optical character recognition [4]. One common approach to large set character recognition is to reduce the candidates down to a manageable level, say 100 or less. This can be achieved in two ways: one is coarse preclassification of the character set into clusters and/or building a decision tree using some predefined set of features, such as the number of strokes. Once done, the result can be used repeatedly for all future input characters.

Another method of candidate reduction is to apply some dynamic evaluation function using some of the input features and generate a small set of candidate characters. This is applied to each input character resulting in a different set. In both methods, a detailed classification should follow the initial coarse classification. Chinese characters are complex with as much as three dozen strokes; therefore, the resulting models could be proportionately complex both in structure and computation. Hence, it is essential to introduce any type of coarse classification for reducing candidates.

Large vocabulary recognition requires a dictionary [27]. Although written as a sequential combination of letters or syllabic characters, the basic unit of writing and communication is word. Simple string recognition could return an arbitrary sequence of letters. Thus, we have to filter out those illegal, nonsensical words in order to build a practical system. This can be done by incorporating a priori linguistic knowledge, often in the form postprocessing after shape-based classification.

Another popular method is incorporating a language model, either a word lexicon or a statistical language model, into the recognition engine. Screening illegal words using a word lexicon can be embedded into the sequential computational structure of the recognizer [13]. For this, it is most appropriate to organize the lexicon into a trie that can help search only legal candidates. Dictionary-based method, however, suffers from the problem of rejecting many proper nouns like a friend's name. The statistical language model comes to the rescue here. Also called an N-gram or a Markov source model, it uses the statistical information about substrings of certain lengths; it is called a bigram if $N = 2$ and a trigram if $N = 3$. These are popular partly because these models fit the Markovian property of HMM and the optimality principle of dynamic programming.

Still another method of incorporating linguistic knowledge is to configure the handwriting pattern model like Figs. 26.5 and 26.9 dynamically based on bigram or trigram [28]. As the forward computation of the Viterbi algorithm proceeds, the model network is made to expand level by level. In this case it is natural to use context-dependent models like triphone equivalents in handwriting recognition. In fact this is a well-known approach in continuous speech recognition.

Example Systems and Softwares

Although online handwriting recognition still abounds with problems and issues, we can say that the technology of the field has largely matured with a number of high-performance systems and commercial products. In this section, let us briefly review some of them.

Research Systems

Most, if not all, research efforts and papers in the literature in online handwriting recognition present certain forms of recognizers or, at least, methods and algorithms. Therefore, it is impossible to enumerate all of them here. Hence, just a few of them which are relevant to this chapter are named here.

BongNet/UniRec

BongNet is online cursive Hangul character recognition model for Korean Hangul characters [12, 14, 29]. Developed at Korea Advanced Institute of Science and Technology (KAIST), this system was the first to introduce explicit modeling of ligatures as separate entities. Along with a frame-synchronous version of level-building algorithm, the model, as shown in Fig. 26.9, solves the problem of mixed cursive handwriting recognition while determining the optimal boundaries of letters and ligatures simultaneously. The performance recorded 93.7 % using HMMs only and ultimately reached upwards of 95 % on their Hangul databases using heuristic structural constraints. The idea has been successfully extended to recognizing English words and ultimately to UniRec [11], the system accepting sentences of mixed languages based on a big circular network connecting several script recognizers.

NPenn++

NPenn++ was developed at University of Karlsruhe, Germany, and Carnegie Mellon University, USA, using a multistate time delay neural network [13]. It is actually a hybrid method combining features of neural network and HMM as well. For word recognition, a tree-structured dictionary is used along with a pruning technique for reducing the search space. For sentence recognition, there is another structure, a search tree that expands each time a letter is to be explored. Each node in the tree is assigned a letter HMM. The terminal nodes are linked back to the root node representing between-word space, thus modeling a sentence as a sequence of words separated by a space. It is reported that the system achieved 91∼92 % hits for 20k vocabulary recognition tasks using their cursive datasets.

Frog on Hand

Frog on hand has been developed using a number of modeling tools in pattern recognition [30]. Besides HMMs, this system employs a clustering technique and a dynamic time-warping algorithm. The developers highlighted a holistic combination of cluster generative statistical DTW (CSDTW) and the HMM-based method to treat handwriting variations. The recognition rate reached around 90 % accuracy using allegedly difficult UNIPEN dataset.

There are numerous other systems worth mentioning but omitted here due to limited space. In particular there are many good systems dealing with Chinese character recognition. But to our relief, there is a wonderful list of systems in the survey paper by Liu et al. [4] and Jaeger et al. [3].

With any of the known pattern recognition techniques, one can start from scratch to build a full-blown system with a complete set of functionalities. But, today, it would often be more desirable to use free or open sources that provide some or most of those functionalities mentioned here. Although often not guaranteed to work without bugs, you can, at least, get some help from any of the softwares listed below.

- **CellWriter** (http://risujin.org/cellwriter/)
  An open source program, designed to be writer-dependent, writing in cells
- **LipiTk** (http://lipitk.sourceforge.net)
  A general toolkit for online handwriting recognizer
- **Rosetta** (http://www.handhelds.org/project/rosetta/)
  A multistroke and full-word handwriting recognition software for X Window System

Commercial Recognizers

There are several well-known products.

- **Graffiti** (http://en.softonic.com/palm/text-entry-graffiti)
  Graffiti is a single-stroke shorthand handwriting recognition system. It has been adopted as an input interface for PDAs based on the Palm OS. It has long been an object of a lawsuit from Xerox. Several variants are available.
- **CalliGrapher** (ParaGraph, acquired by PhatWare, http://ww.PhatWare.com)
  This software provides a multilingual support for Western European languages, available for Windows Mobile. It accepts all handwriting styles, print, cursive or mixed.
- **MyScript Notes** (Vision Objects, http://www.visionobjects.com)
  This software recognizes over 80 languages, including Russian, Chinese, Korean Hangul, Japanese, Cyrillic, and Arabic. It has been trained through millions of handwritten samples from native writers.
- **riteScript** (http://www.ritescript.com)
  It was developed by Evernote, a successor of Parascript's division. riteScript accepts both cursive and block letters and supports most of the functionalities such as vocabulary and non-vocabulary words and baseline detection in sentences.

Conclusion

This chapter discussed many of the issues that need to be considered to develop a practical system for natural cursive handwriting recognition. Albeit brief, the discussion has been made broad enough to provide quick and helpful tips for designing diverse script recognizers with different characteristics. It is based on the belief that designers of different language script recognizers can benefit a lot from the techniques and solutions of other languages.

Signal processing often constitutes the core of many practical system development. Indeed the success of this step is a key to successful recognition. Nevertheless, there is no great theory other than traditional "good old" techniques, and heuristics and task-dependent insights abound here. Thus, the discussion has carefully been limited to a number of generic techniques. The extraction of features largely depends on the type of recognition algorithm employed. Thus, only a few generic issues are addressed including dynamic information and high-level shape features.

The lions' share of this chapter has been devoted to the recognition methods. And this is the place where most ideas and theories have been developed. The discussion is around an online handwriting recognition method based on the HMM and explores the issues of modeling data variabilities and incorporating linguistic information. The use of the HMM is highly appropriate in that it is one of the most successful tools to date, and it often is a designer's choice as it is simple to develop and easy to extend for complex patterns.

Finally this chapter cited a few research and commercial softwares, the list of which is far from being complete or comprehensive. In fact there are numerous other successful softwares claiming a large market share today. But only a handful of selected systems with historical importance and technical significance have been listed here. System developers may benefit a lot from case studies on those systems before designing their own recognizers.

A final remark is that the technology is highly mature and experiences are many. And the task that remained is a better and deeper understanding of the problems at hand – language and script characteristics, writing behaviors and styles, the type of applications, and so on – and a more extensive exploitation of a priori knowledge.

Executive Summary

This chapter introduced roughly the techniques in online handwriting recognition. Those techniques are language and script-dependent. Therefore, a mere chapter would never be successful in digging out all the details. Instead this chapter has focused on understanding the differences and similarities across different scripts and then hard issues in developing real-world solutions. One script could be similar to some others linguistically or in pattern recognition perspective. So we can learn a lot by watching what other people are doing.

Since the introduction and following the rapid expansion of smartphones as a more attractive, personal device, the touch interface has become popular. The main interface, however, is still the keyboard, only just soft. But this does not mean that there is no place in those or other devices for a pen. In fact the pen is more convenient in many cases, such as large alphabet or large character set-based scripts, correcting errors or mistakes in texts, and in tasks extending hours or requiring more than touch.

During the past decades, the recognition algorithm has improved a lot. When typing errors and corrections are taken into account, pen-based text input could be faster or more accurate than the keyboard. The problem, if any, could be the interface. Most smartphone users are accustomed to keyboard, but not to stylus even though they have lived with lots of pens since childhood. Therefore, the real issue surrounding online handwriting is making the interface more intuitive and familiar.

Finally, the recognition engine technology has matured, if not finished, a lot after decades of research and development. Here again a real and more promising solution would be designing a "recombinant" system – combining existing techniques in an integrated framework while incorporating script-specific features and knowledge available.

Cross-References

►Asian Character Recognition
►Datasets and Annotations for Document Analysis and Recognition
►Handprinted Character and Word Recognition
►Middle Eastern Character Recognition
►Online Signature Verification

References

1. Tappert CC, Suen CY, Wakahara T (1990) The state of the art in on-line handwriting recognition. IEEE Trans Pattern Anal Mach Intell 12(8):787–808
2. Plamondon R, Srihari SN (2000) On-line and off-line handwriting recognition: a comprehensive survey. IEEE Trans Pattern Anal Mach Intell 22(1):63–82
3. Jaeger S et al (2003) The state of the art in Japanese on-line handwriting recognition compared to techniques in western handwriting recognition. Int J Doc Anal Recognit 6(2):75–88
4. Liu C-L et al (2004) On-line recognition of Chinese characters: the state of the art. IEEE Trans Pattern Anal Mach Intell 26(2):198–213
5. Al Emami S, Usher M (1990) On-line recognition of handwritten Arabic characters. IEEE Trans Pattern Anal Mach Intell 12(7):704–710
6. Lorigo LM, Govindaraju V (2006) Offline Arabic handwriting recognition: a survey. IEEE Trans Pattern Anal Mach Intell 28(5):712–724
7. Mezghani N, Mitiche A, Cheriet M (2008) Bayes classification of online Arabic characters by Gibbs modeling of class conditional densities. IEEE Trans Pattern Anal Mach Intell 30(7):1121–1131
8. Pal U, Chaudhuri BB (2004) Indian script character recognition: a survey. Pattern Recognit 37(9):1887–1899

9. Dongre V, Mankar V (2010) A review of research on Devanagari character recognition. Int J Comput Appl 12(2):8–15
10. Boccignone G, Chianese A, Cordella LP, Marcelli A (1993) Recovering dynamic information from static handwriting. Pattern Recognit 26(3):409–418
11. Jaeger S (1998) Recovering dynamic information from static, handwritten word images. PhD thesis, University of Freiburg, Foelbach
12. Sin B-K, Kim JH (1998) Network-based approach to Korean handwriting analysis. Int J Pattern Recognit Artif Intell 12(2):233–249
13. Jaeger S, Manke S, Reichert J, Waibel A (2001) Online handwriting recognition: the NPen++ recognizer. Int J Doc Anal Recognit 3:169–180
14. Sin B-K, Kim JH (1997) Ligature modeling for online cursive script recognition. IEEE Trans Pattern Anal Mach Intell 19(6):623–633
15. Bishop C (2006) Pattern recognition and machine learning. Springer, New York, p 740
16. Bahlmann C (2006) Directional features in online handwriting recognition. Pattern Recognit 39:115–125
17. Cho S-J, Kim JH (2004) Bayesian network modeling of strokes and their relationships for on-line handwriting recognition. Pattern Recognit 37:253–264
18. Chan K-F, Yeung D-Y (2000) Mathematical expression recognition: a survey. Int J Doc Anal Recognit 3(1):3–15
19. Meyers CS, Rabiner LR (1981) Connected digit recognition using a level-building DTW algorithm. IEEE Trans Acoust Speech Signal Process ASSP-29:351–363
20. Sakoe H (1979) Two-Level DP-matching – a dynamic programming-based pattern matching algorithm for connected word recognition. IEEE Trans Acoust Speech Signal Process ASSP-27(6):588–595
21. Ney H (1984) The use of a one-stage dynamic programming algorithm for connected word recognition. IEEE Trans Acoust Speech Signal Process ASSP-32:263–271
22. Lee C-H, Rabiner LR (1989) A frame-synchronous network search algorithm for connected word recognition. IEEE Trans Acoust Speech Signal Process 7(11):1649–1658
23. Chen JW, Lee SY (1997) On-line Chinese character recognition via a representation of spatial relationships between strokes. Int J Pattern Recognit Artif Intell 11(3):329–357
24. Shin J-P (2002) Optimal stroke-correspondence search method for on-line character recognition. Pattern Recognit Lett 23:601–608
25. Plamondon R, Guerfali W (1998) The generation of handwriting with delta-lognormal synergies. Biol Cybern 78:119–132
26. Lee KF (1989) Automatic speech recognition: the development of the SPHINX system. Kluwer Academic, Boston
27. Seni G, Srihari RK, Nasrabadi N (1996) Large vocabulary recognition of on-line handwritten cursive words. IEEE Trans Pattern Anal Mach Intell 18(6):757–762
28. Hu J, Lim SG, Brown MK (2000) Writer independent on-line handwriting recognition using an HMM approach. Pattern Recognit 33:133–147
29. Lee J, Kim J (1997) A unified network-based approach for online recognition of multilingual cursive handwritings. In: Progress in handwriting recognition. World Scientific Publishing, pp 81–86
30. Bahlmann C, Burkhardt H (2004) The writer independent online handwriting recognition system frog on hand and cluster generative statistical dynamic warping. IEEE Trans Pattern Anal Mach Intell 26(3):299–310
31. Lee S-W (ed) (1999) Advances in handwriting recognition. Series in machine perception and artificial intelligence, vol 34. World Scientific, Singapore/River Edge, p 587
32. Liu Z-Q, Cai J-H, Buse R (2003) Handwriting recognition, soft computing and probabilistic approaches. Studies in fuzziness and soft computing, vol 133. Springer, New York, p 230
33. Impedovo S (2012) Fundamentals in handwriting recognition. Springer-Verlag, p 496
34. Doermann D, Jaeger S (eds) (2006) Arabic and Chinese handwriting recognition. In: SACH 2006, Summit, College Park. LNCS. Springer-Verlag, New York

35. Su T (2013) Chinese handwriting recognition: an algorithmic perspective. Springer, Berlin/New York, p 139
36. Mondal T (2010) On-line handwriting recognition of Indian scripts – the first benchmark. In: Proceedings of the international conference on frontiers in handwriting recognition, Kolkata, pp 200–205, Nov 2010

Further Reading

There is a great body of literature on online handwriting recognition in numerous conference and workshop proceedings. And some of important technologies and results are well documented usually in the form of handbooks, such as those edited by Lee [31] and Liu et al. [32], while the most recent book on handwriting recognition methods will be the work by Impedovo [33].

The last decades saw a surge of research in Chinese and Arabic scripts. Interested readers may start their intellectual exploration with handbooks by Doerman and Jaeger [34] and Su [35]. On the other hand, many Indian scripts in the South Asian subcontinent seem to have thus far been defined as an offline character recognition problem. However, enthusiastic students and serious system designers may refer to any journal and conference papers such as the one on a benchmark test (2010) [36] to get a picture of the technology landscape in the continent.