Handprinted Character and Word Recognition

11

Sergey Tulyakov and Venu Govindaraju

Contents

S. Tulyakov
Center for Unified Biometrics and Sensors, University at Buffalo, Amherst, NY, USA
e-mail: tulyakov@buffalo.edu

V. Govindaraju
Department of Computer Science & Engineering, Center for Unified Biometrics and Sensors, University at Buffalo, Amherst, NY, USA
e-mail: govind@buffalo.edu

D. Doermann, K. Tombre (eds.), *Handbook of Document Image Processing and Recognition*, DOI 10.1007/978-0-85729-859-1_11,

Abstract

The handprinted texts are produced when the writer tries to emulate some standard printed representation of the characters with the goal to make the written texts legible. Postal address blocks, different fillable forms, or other documents are among the examples of handprinting. Current chapter reviews the main techniques in recognizing handprinted characters and words. It outlines the characteristics of the handprinted texts, such as the stroke structure, distribution of strokes inside character bounding box, and frequently separated characters. Then it investigates how the reviewed techniques address such characteristics of the handprinted texts.

Keywords

Binarization • Contour • Dynamic programming • Fillable forms • Handprinted character • Handprinted word • Image histograms • K-nearest neighbor recognizer • Moments • Postal address • Segmentation • Skeleton • Zoning

Introduction

Certain situations might require a person to write some information in a most legible form, easily readable by other people. Such situations include the filling of legal forms, financial and tax documents, postal address information, etc. The person-specific learned cursive writing, while usually easily readable by the writer, might not always be well understood by other persons. Instead, in such situations, a person tries to write the characters resembling some standard printed characters. The handprinted text can be characterized by the characters consisting of a series of separate strokes or arc elements, the frequent separation of characters in a word and the absence of ligatures, and frequent use of uppercase style characters instead of lowercase. Handprinting also differs from machine-printed text by the bigger variability in the shape of written characters, variability in character positions and sizes, the absence of some specific font, and smaller proportion of stroke widths to stroke lengths. The recognition of handprinted characters and words thus represents an intermediate problem between machine-printed text recognition (▸Chap. 10 (Machine-Printed Character Recognition)) and handwritten text recognition (▸Chap. 12 (Continuous Handwritten Script Recognition)) and generally is more difficult than the first problem and easier than the second.

Figure 11.1 contains few examples of handprinted text images extracted from the address blocks of postal mail pieces. The examples show that the distinction between handprinted and handwritten words is subjective, and same address block can contain both separated and clearly handprinted characters, as well as connected, cursive, and rather handwritten characters.

The terms “handwritten” and “handprinted” sometimes have the same meaning in the research literature, and same recognition methods could be employed for both types of written texts. In this chapter it will be assumed that the recognition methods attempting to perform explicit segmentation of text into characters and

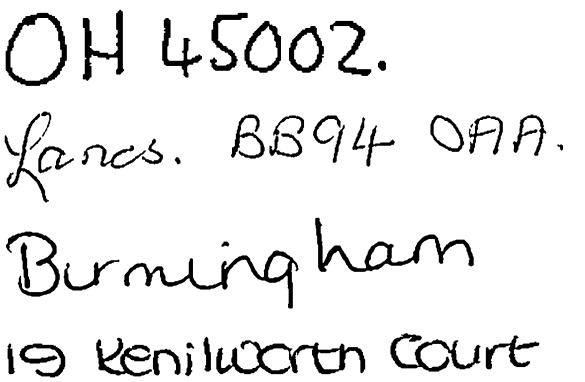

Fig. 11.1 Handprinted samples from postal address images: The United States zip code (**a**), The United Kingdom postcode (**b**), city name (**c**), and street address (**d**)

perform recognition of such segmented characters are specific for handprinted text recognition. In contrast, the recognition methods which rely on implicit segmentation of words into characters, or not performing any segmentation at all, are specific for handwritten text recognition problem.

A significant number of works have been published on the topic of handprinted text processing, including extensive reviews [3, 12, 17, 80, 81, 89, 93]. Some of the methods presented in the published literature have universal nature and might be applicable to other types of texts or imagery type. As a result, they might not be very efficient in processing handprinted text. This chapter outlines the main developed approaches and discusses their strength or weaknesses specifically with respect to the handprinted texts.

History and Importance

The handprinted character and word recognition represent one of the earliest research problems in the image processing and pattern classification fields. The problem development was linked to the appearance of computers, image scanners, and digital tablets, as well as the desire to utilize these new devices for automating certain record-keeping tasks. Due to relatively modest computer hardware requirements and the sufficient diversity of the handprinted character shapes, the area of handprinted character recognition became possibly the most studied area of artificial intelligence field in the years 1980–2000.

The handprinted character recognition problem served as a test case for many pattern recognition and pattern classification algorithms which have been developed in the 1960s–1980s. For example, the early works investigated both syntactical [2, 92] and statistical [5] methods of handprinted character recognition. The syntactical approaches were popular at the beginning due to visual and well-defined nature of features and the sequences of features representing characters. The recognition was performed according to well-formulated syntactical rules and was fairly efficient on a limited hardware of the days. But, with the increased capability of computers,

it was becoming clear that syntactical representation of handprinted characters does not allow the description of all possible character shapes and robust estimation of observation likelihoods. Consequently, the majority of the methods followed the statistical methods for character recognition [5,91].

The research in this area also underscored the importance of publicly available datasets for training and testing pattern recognition algorithms. The handprinted character datasets collected at NIST [32, 36] and CEDAR [43], as well as handwritten word IAM [69] database, provide the researchers the ability to develop algorithms and to compare their performance to the performance of other researcher's algorithms. ▸Chapter 29 (Datasets and Annotations for Document Analysis and Recognition) provides additional references to existing databases.

Finally, the handprinted text recognition algorithms have great value to today's industry. Many applications processing handprinted character and word inputs have been automated, recognizing the handprinted texts in different forms, such as tax forms, postal addresses, and financial documents. Undoubtedly, they will continue to be widely used as long as people transmit the information by means of handprinted text.

Evolution of the Problem

From the beginning, the development of the handprinted text recognition algorithms depended on the availability of hardware. The early research (1960–1970) was mostly performed only in big companies who could afford some computer time and had access to scanners and touchpad devices. As a consequence, many early algorithms [5, 27, 92] were concerned with hardware limitations and designed algorithms accordingly to address low storage and computing power requirements. Some of the important techniques, such as contour [26] and skeleton representations [10] of character images, feature vector extraction and zoning [5] have originated at that time. Online handwritten character recognition [8] took roots at this time as well (▸Chap. 26 (Online Handwriting Recognition)).

The proliferation of less expensive personal computers in 1980–1990 spurred the burst of research activity by universities and other organizations. This time coincided with the apparent peak in handprinted text recognition research. As it was already mentioned, the research has gradually shifted from syntactical matching [77,92] to statistical approaches [37,86,91,93]. In contrast to syntactical methods, the more advanced type of approaches allowing the descriptive representation of characters, structural methods have appeared. The structural methods could be based on graph matching [62, 98] or variable length structural feature vectors with HMM-based matching [103].

Currently, the interest in the area of handprinted recognition is somewhat diminished, probably due to the increased computing power and the ability to perform other image processing tasks: general object recognition and scene analysis, face and gesture recognition, biometric applications, etc. Still, there is an active research that continues in the processing of non-Latin-based scripts

(see ▸Chaps. 13 (Middle Eastern Character Recognition) and ▸14 (Asian Character Recognition)), which were previously ignored, as well as the application of machine learning methods developed in other areas for handprinted texts and the development of new applications (medical form processing, digital libraries, etc.).

Applications

One of the frequent applications of handprinted text recognition is the automated reading of handprinted input fields on different fillable forms [33]. One of the properties of most of these forms helping the recognizers is some limitation on the possible word and character sets. Also, the forms themselves are typically designed so that their reading by the automated algorithms is easy. Some research addresses the topic of designing the forms most easily read by the computers [31].

The examples of form reading applications include the processing of handprinted names, addresses, and other possible information on tax forms [85] or census forms [66]. Recently, some research has been conducted into processing of medical forms [13, 70]. Note that this recent application is significantly more challenging due to less accurate handprinting, low quality of the images (carbon copies), and increased lexicon size. ▸Chapter 19 (Recognition of Tables and Forms) contains the examples of different forms and describes specific algorithms for form processing in detail.

Reading of the bank checks is another application which can be easily automated [47, 50, 54, 88]. Two fields in the bank checks, the numerical amount of the check (courtesy) and the handwritten amount (legal), could both be read by handprinted text recognition algorithms, and two read amounts could be verified with each other.

The more challenging application for handprinted text recognition is the processing of postal mail. In contrast to fillable forms, the postal envelopes might not have well-defined address blocks with particular fields, but instead allow free-form writing of the address. Still, due to the specific order of address information and the limited lexicon involved, automated reading of postal addresses has been successfully implemented. As an example, the system for reading the United States postal addresses first tries to recognize numerical ZIP codes and house street numbers, retrieves from the database the street names and city names which have such ZIP codes and house numbers, and verifies the presence of such names in the address block [19, 20, 83]. ▸Chapter 21 (Document Analysis in Postal Applications and Check Processing) discusses the postal address and bank check recognition applications in more detail.

The above applications for reading handprinted text have been successfully developed and used in different countries. But, due to the increased use of Internet communications and the replacement of paper documents by their digital versions, it is possible that such applications will have limited use in the future. On the other hand, it is possible that the widespread use of tablet computers will result in some need to process the handprinted text entered on it. In contrast to text written on paper and later scanned as digital image (*off-line handwriting*),

the digital handwriting devices capture the trajectory of pen movements (analogous to skeleton representation, see section "Representations of Handprinted Text") along with timestamps of each drawn point (*online handwriting*). Such information is inherently more precise, and specifically designed online handwriting recognition methods (refer to ▸Chap. 26 (Online Handwriting Recognition)) typically deliver better performance than off-line handwriting recognition methods considered in this chapter.

Main Difficulties

Given high-quality images of a well-separated characters or character strings, the current recognizers are able to produce nearly perfect recognition results. Most of the errors appear when the input consists of low-quality, poorly segmented, or just ambiguous images. Therefore, the main problem of handprinted text recognition research might be outside the scope of character and word recognizers, but with the preprocessing and noise removing algorithms. Still, the efficient recognizer should be able to overcome some of the difficulties and possibly verify the results of preprocessing algorithms.

1. *Touching or Intersecting Characters or Extraneous Strokes*

   Humans can relatively easily read the texts containing, for example, intersecting characters or crossed-out characters. Intuitively, the extraneous information from the character image is getting automatically discarded during the reading process. Most of the current character recognizers do not attempt to discard any such information, and the features are extracted from the whole image. Although the presence of true character's features mitigates the effect, and the character could be recognized correctly in such situations, the inability to account for non-character information introduces an uncertainty in the recognition results and leads to further errors in document analysis.
2. *Low-Quality Images*

   The low image quality expresses itself in broken components of binarized characters, missing structural elements due to too light or too dark image parts, extraneous noise elements, etc. Some of the recognition algorithms might try to reconstruct the strokes, merge together broken components, and so on. The reconstruction of the strokes might make sense if, for example, it was known beforehand that the large percentage of images contains broken strokes. Usually, researchers will look at the samples of the datasets and decide whether such reconstruction is needed. But given other datasets with no broken strokes, such modifications might introduce errors in the images and result in poorer recognition rate.

   The difficulty in processing low-quality images and the images containing extraneous information seems to be inherent in current recognition algorithms. Instead of trying to connect together the low-level features, build a structural representation of characters, and match these structural representations with stored prototypes, a simpler approach based on feature vectors and discriminating

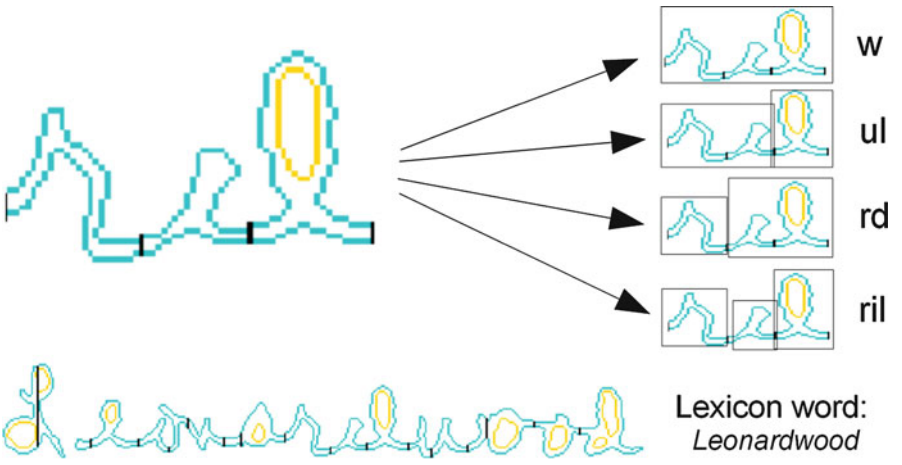

Fig. 11.2 Ambiguous appearance of the segmented subimage might lead to four possible recognition results. Using the lexicon word and the original image to match all the characters in the word, the recognizer deduces that the subimage should be matched to "rd"

these vectors in the feature space dominates the research field. The current methods based on structural feature extraction [103] might not have the desired precision to find and represent structural features sufficiently well, and as a result, they are generally outperformed by the statistical methods.

3. *Ambiguous Characters*

   Although the goal of handprinting is to make the text as clear as possible, different factors, such haste, inattention, and physical difficulties, could influence the legibility of written images. Some characters could also be more similar to each other than to other characters due to the presence of common structural elements and the ways of writing them. The segmentation choices can also add to the ambiguities during reading. Figure 11.2 has an example of segmented image which could be recognized in different ways.

   This problem is difficult to solve since the characters seem to not have a uniform distribution in the image space, and the confusion among recognition results is common. One of the practical solutions is to use some external knowledge about the image. Figure 11.2 presents an example of how the segment corresponding to two letters "rd" is recognized using the knowledge about lexicon word "leonardwood" and the matches between its characters and other segments of the image. The ambiguity problem is so common that most word recognizers rely on either the word lexicon or some grammatical model during recognition. Since the general texts can have very large lexicons, the recognition of such texts still presents a large problem.

4. *Performance Evaluation*

   The research into handprinting recognition is somewhat hampered by the difficulty of properly evaluating the algorithms and comparing their performance

to the algorithms of other researchers. The reasons for this difficulty include the absence of common benchmarks (the training and testing sets are not defined even for publicly available datasets), the easiness of available datasets (it might be difficult to compare algorithms if they have the recognition rate of 99 % or more), and the lack of detailed description of the published algorithms [81]. It might be even difficult to compare the recognition algorithms operating on the same benchmarks [65], since the results of recognition could be greatly influenced by the preprocessing techniques, feature extraction, or postprocessing additions. As an example, the k-nearest method has been shown to have better performance than multilayer perceptron on the features obtained by Karhunen-Loève transform [37] but worse performance on other features [63]. ▸Chapter 30 (Tools and Metrics for Document Analysis Systems Evaluation) has additional discussion on the possible benchmarks and algorithm performance evaluation methods.

Summary of the State of the Art

It is reasonable to say that the area of handprinted text recognition research has mostly matured to the application stage, and a significant portion of the development is done by the commercial companies, such as BBN and Siemens. The current systems could implement the recognition of the scripts from different languages and with large lexicons or with the grammar models defined by the recognized language. The reported error rates could be sufficiently small for the successful reading of postal addresses or government forms. The extensive review of some current commercial and university recognizers can be found in [81].

Most current algorithms for handprinted and handwritten word recognition seems to be based on *Hidden Markov Models* (HMM). The HMMs offer more transparent derivation of the matching scores than dynamic programming-based approaches (see section "Character Based Word Recognition") and could be more easily trained with the word images. HMM-based word recognition could be *segmentation-free* or it could incorporate some type of presegmentation of the image into candidate characters [22]. It is not quite clear whether segmentation free or segmentation-based algorithms have better performance; the reviews argue for both first [81] and second [99] kinds.

The segmentation-free HMMs can incorporate the HMM-based character recognizers which are trained as parts of word HMM. Most other methods incorporate explicitly trained character recognizers. The methods for extracting features and recognizing characters vary significantly. Although discriminative methods, such as support vector machines (SVM), might show better performance for separate character recognition [65], the use of character recognizers inside HMM might require a proper probabilistic measures, e.g., likelihood, and generative classification methods, such as *Gaussian Mixture Models* (GMM).

Preprocessing of Handprinted Character and Word Recognition Texts

▸Chapters 4 (Imaging Techniques in Document Analysis Processes), ▸5 (Page Segmentation Techniques in Document Analysis), and ▸8 (Text Segmentation for Document Recognition) of this book have already addressed general methods for preprocessing the document images, including binarization, noise removal, and segmentation of lines and words. The current section underlines some of the important methods which specifically deal with handprinted texts, in contrast to other text types such as machine-printed and cursive handwriting.

Properties of Handprinted Text

The existing methods dealing with handprinted texts can be evaluated on how well they utilize the inherent properties of such texts. Therefore, let us define the properties of handprinted texts, which could be useful for recognition algorithms.

1. The image pixels come from two separate classes: background (paper) and foreground (ink). The colors of background and foreground pixels are generally different.
2. The areas of background and foreground pixels, as well as the boundary between them, have some smoothness constraints.
3. The width of the foreground areas, which can be defined as an average distance from the foreground pixel to the closest background pixel, is determined by the pen width and should remain relatively constant for a handwritten document. The stroke width is usually small relative to the character size for handprinted documents.
4. The characters are composed of a small number of structural elements: strokes, arcs, and dots. Each character class has its own typical set of such elements arranged in a specific order.
5. Each character occupies a limited region of space, and the structural elements of a character are somewhat uniformly distributed inside its bounding box.
6. The characters can be separate from each other or have some stroke elements, *ligatures*, connecting them.
7. The characters in a handprinted word have similar sizes, with some variations possible depending on the character class.

The above properties of handprinted texts are similar to the properties of handwritten texts. The possible difference lies in the better separation of characters, more prominent nature of structural elements inside each character, and more uniform placement of these elements inside character's bounding box. The handprinted texts are also less writer specific, and while handwritten text recognizers might need to be adjusted for a particular writer's style, the recognizers of handprinted texts

could be trained in writer-independent manner. These properties also differ from the properties of machine-printed texts, which could have greater similarity between characters of the same class, the presence of vertical and horizontal strokes, greater separability between classes, and particular font characteristics, e.g., serif.

The coarticulation effect is another property of the cursive writing, which could be possibly accounted for during the recognition [80]. The effect consists in the appearance changes of characters with respect to the neighboring characters. For example, the upper or lower ligature from a previous character will influence the appearance of current character. It could be assumed that the coarticulation effect is less pronounced in handprinted texts than in the cursive handwritten ones and that handprinted text recognizers could omit it from consideration.

If not all properties are utilized by the particular approach, then it might not be optimal and performance will suffer. For example, the character recognition algorithms which operate on grayscale character images, such as image moment methods, do not assume usually clear boundary between foreground and background. As a result, their performance is typically worse than the performance of algorithms explicitly locating this boundary and extracting features based on it [86].

Binarization

Due to the first property of handprinted text, foreground, and background pixels, the binarization, that is the assignment of pixels to one of these two classes, could rely on the difference in distributions of the grayscale pixel values of these two classes. By counting the numbers of pixels in the image with different intensity, a histogram of pixel intensities could be constructed. The foreground pixels should contribute to the peak of the histogram near black pixel intensity, and the foreground should form a peak near white pixel intensity. Simple thresholding of the histogram, e.g., at the lowest point between two peaks, will separate the image pixels into two classes. For better performance, some prior distributions of pixel intensities can be assumed along with prior class probabilities as it was done by Otsu [75]. One of the possible modification of thresholding technique is to perform multiple candidate binarizations with different thresholds and then estimate, whether the obtained binary images satisfy some properties of handprinted text [104]. Sezgin and Sankur [84] provides a review of different thresholding methods, including adaptive methods which restrict histograms to the local neighborhood of a currently thresholded pixel.

The traditional thresholding methods do not account for the smoothness of foreground and background regions or their boundary; consequently they can be applied in many research areas with the same success as in handprinted character binarization. The methods, which are more suitable for processing handprinted characters, should take into account the assumption of the smoothness of these regions and their boundary. The application of smoothing filters during binarization

will result in smoother boundaries between binarized regions and more closely match the "true" boundaries of characters.

Further developments in binarization techniques for handprinted characters include the use of filters accounting for the edges between foreground and background areas [76], the filters depending on the width of the strokes and thus well suitable to preserve the stroke structure [48], and, finally, the filters which assume the possible changes in stroke directions [73] and which preserve the connections between different strokes. As one can see, these techniques increasingly take into account the properties of handprinted texts and are bound to improve the performance of the text processing systems.

The binarization algorithm should preferably learn the correspondence between the true or prototype binary character and the grayscale image from training data and utilize this correspondence during image processing. Statistical modeling methods try to create the models of prototype binary images and their mappings to grayscale images. Markov Random Fields (MRF) could be used for such modeling. In [13] the MRF consists of rectangular grid of prototype binary image patches, $x_{i,j}$. Two types of dependencies are learned using the training data: the probabilities of having some particular patch given a set of neighboring patches, $P(x_{i,j}, \{x_{i+\delta_i, j+\delta_j}\}_{\delta_i, \delta_j \in \{-1,0,1\}})$, and the probabilities of observing a grayscale patch $y_{i,j}$ for the underlying prototype patch, $P(y_{i,j}, x_{i,j})$. The binarization of grayscale image composed of the patches $\{y_{i,j}\}$ consists in finding the set of prototype patches $\{y_{i,j}\}$ which maximize both sets of probabilities. The presented algorithm has the desired properties of being able to learn from the training samples the possible local prototype shapes as well as the appearance of these shapes in the images. Thus, it implicitly takes into consideration the first four properties of the handprinted texts proposed before. The experiments confirm the superior performance of this method over non-trainable thresholding techniques, especially for low-quality handwritten text images.

Representations of Handprinted Text

The binary character image is one possible representation of the handprinted character, and many character recognition algorithms use it to extract features. But there are other representations which could be useful for feature extraction, specifically contours and skeletons.

Freman [26, 27] proposed using the chain codes for representing arbitrary curves. The curves on the image are the sequences of connected pixels, and in order to store these sequences in the computer storage, only the direction of next pixel with respect to previous one can be recorded. Each of 8 directions is encoded by 3 bits, and the total curve has around 3^n bits. Such encoding will be more effective than the traditional run-length encoding of the binary images for images containing few connected components.

Toussaint and Donaldson [92] traced the outer contours of the characters and used the positions of the local extrema points to represent a character as a

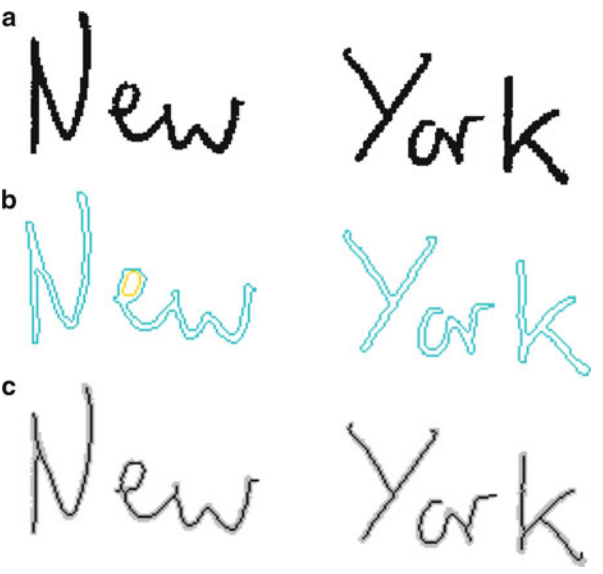

Fig. 11.3 Possible representations of handprinted texts: binary image (**a**), contour set (**b**), and skeleton (**c**)

short sequence of local extrema positions. The distance between two characters can be calculated as minimum edit distance between these sequences. As this work exemplifies, the contour representation of characters has a sequential or one-dimensional structure and allows different methods for extracting features or matching than two dimensional binary representation. For example, the works cited in section "Contour Moments" construct moments using one-dimensional contour functions, which arguably are more effective in encoding character shapes than the two-dimensional moments discussed in section "Image Moments."

The third possible representation of the handprinted character shapes is based on the notion that they were created by a pen movements, and it is possible to derive the trajectory of the center of the pen from it. There are different definitions and names for this central trajectory of the pen, with possibly one of the first one being the *medial axis* [10]. The most popular and simple methods of obtaining the medial axis, or *skeleton*, of the character are based on the *thinning* process, in which the boundary pixels of the shape are iteratively removed following some predetermined rules [1, 7, 38]. Figure 11.3 gives an example of the three possible representations discussed; the skeleton extraction method is based on [7].

Both contour and skeleton representations of characters allow the extraction of directional features or curvature features in form of slope of contour or skeleton interval. Although it is possible to extract such features from binary images or even original grayscale images, contour and skeletons make the process simpler. They also make simple the detection of the possible strokes or arcs or the end points. The skeletons also allow easy segmentation methods and the detection of the stroke intersections.

Segmentation

▸Chapter 8 (Text Segmentation for Document Recognition) provided an overview of general segmentation methods applicable to different types of texts. As it was mentioned in this chapter, one of the properties of the handprinted texts is the frequent separation of characters, and possibly an easier approach to segmentation can be used. Still, as Fig. 11.1 illustrates, it is typical for handprinted texts to have at least some of the characters to be connected or touched. The segmentation of handprinted texts, therefore, is a necessary step during recognition. Although, it would be desirable to have a recognizer, which would spot or recognize characters or words directly from the unsegmented texts, apparently such recognizers do not exist or do not have satisfactory performance yet.

General two approaches to character segmentation exist. One approach performs an explicit segmentation procedure and tries to understand the structure of the character elements near the segmentation point. The other approach is performing segmentation implicitly during word recognition; possible segmentation points or separation boundaries could have a large variety, and a particular segmentation point is determined by the best scores of recognized characters surrounding it. Cursive word recognition algorithms, such as HMMs, typically rely on this method. The set of possible segmentation boundaries could be simply represented by vertical lines, sets of two points in upper and lower contours, or by a single point in a skeleton representation of the word. Casey and Lecolinet [14] contains a review of segmentation methods along with their more detailed categorization.

The explicit segmentation algorithms are probably more powerful, since they try to utilize the knowledge about specific structures of handprinted texts. Fujisawa et al. [28] considered different possibilities on how two digits could touch each other, e.g., the stroke end of one digit is touching the middle of a stroke or arc of another digit and similar possible scenarios. Depending on the contour profiles, these scenarios could be separated, and specific segmentation could be performed for each scenario. Madhvanath et al. [67] considered another set of features, such as the closeness of upper and lower contours, the closeness of both of them to the baseline, and the separation of ascenders and descenders. The segmentation points obtained with this method are shown in Fig. 11.6.

It might be also possible to utilize some machine learning techniques to separate the true segmentation points from false ones. For example, [58] uses a neural network accepting as input set features extracted from a small region around candidate segmentation point and verifies whether this is valid segmentation point.

There are also some works which do not perform the segmentation of touching characters but try to classify the sets of touching characters. For example, Ciresan [18] is training a convolutional neural network to recognize the pairs of digits. The digit string recognition system tries to match a particular component to a single digit or to the pair of touching digits and selects the better match as recognition result. The technique is clearly limited since it would not allow the recognition of three touching digits. Also, it is doubtful that this algorithm has

better performance than explicit segmentation algorithms utilizing the patterns of connections in touching characters.

The words in handprinted texts are usually well separated, and one could face the problem of determining the boundaries of the words and distinguishing them from the possibly separated characters inside the words. This problem is somewhat easier than the segmentation of the word into characters. The algorithms performing word segmentation could use such information as typical sizes of the characters, intervals between characters, or intervals and lengths of strokes or ligatures [79].

Feature Extraction

Current section presents some techniques in extracting features from character images. The literature on this topic is numerous and additional analysis can be found in some survey works [3, 12, 17, 72, 93]. The discussion of this section emphasizes two major directions for feature extraction: global and local. Global features are extracted from the whole character images by some unified approach, and they are usually complementary to each other. For example, the moments are extracted by the same approach and might reflect different frequencies or directions of moment functions. Whereas each moment taken separately has little meaning, the full set of moments can be used to reconstruct the original character image. Local features are typically extracted in some local neighborhood, or a zone, of the character image; the full feature vector for a character is a concatenation of feature vectors extracted for each zone. The features themselves are usually more simple in this case. They might be an average stroke direction, a histogram of gradients, the bits accounting for the presence of a stroke end or a corner in the zone, etc. Overall, one can hypothesize that the approaches extracting local features are more powerful since they exploit the characteristics of handprinted text to a greater degree.

Global Image Features

Projection Histograms

One of the simplest and possibly earliest methods of extracting the features includes the counting of foreground pixels at different horizontal or vertical pixel rows of the character image [55]. These numbers form the projection graphs, or histograms, which can be used as features themselves or could serve for extracting some secondary features. Although it might be proved that the set of projection histograms taken for different angles is sufficient to reconstructing original character image similar to the reconstruction method of computer tomography (CT) scans, the applications in character recognition do not go so far and might use only a limited projection set for features.

The projections are typically used for segmentation methods rather than for recognition. As an example of using projections for recognition, Wang et al. [100] construct the projections originating from the center of moments of the character for different angles. The Fourier moments are then extracted from the projection

histogram and used as features for character recognition. In general, the set of projections does not seem to reflect many properties of the handprinted characters, and therefore this type of features might not have good performance, and as a result, was used rarely for recognition.

Image Moments

The image of character can be represented as a function in two-dimensional space: $I(x, y)$, where x and y are the coordinates of the pixels and I represents the value of the intensity of the pixel (e.g., it can be assumed that the value 0 represents white pixel and 1 represents black). The set of functions defined on the same image area A, e.g., $L^2(A)$, forms a vector space with inner product:

$$< f_1, f_2 >= \iint_{x,y \in A} f_1(x, y) f_2(x, y) dx dy.$$

For any orthonormal basis $\{v_1, \ldots\}$ in this vector space, the image function can be expressed as a linear sum of basis functions:

$$I(x, y) = \sum_i a_i v_i(x, y) \text{ , where } a_i = \iint_{x,y \in A} I(x, y) v_i(x, y) dx dy.$$

Then, the original image function $I(x, y)$ can be approximated by the sum of first N terms:

$$||I(x, y) - \hat{I}_N(x, y)|| \xrightarrow[N \to \infty]{} 0 \text{ , where } \hat{I}_N(x, y) = \sum_{i=1}^{N} a_i v_i(x, y).$$

Depending on the choice of the orthonormal basis, different approximations of the image function $I(x, y)$ are possible. The multiple proposed methods of this type usually choose the basis functions of some predetermined form. Possibly one of the first works on applying the methods of basis function decomposition for character images by Hu [41] considered polynomial basis functions of increasing orders, and extracted coefficients were called *moments*.

$$M_{ij} = \iint_{x,y \in A} x^i y^j I(x, y) dx dy$$

The polynomials $x^i y^j$ in the above formula do not represent an orthonormal basis, and thus, the reconstruction of the original image by means of moments M_{ij} is not straightforward. Nevertheless, they can be used as features for character classifiers as any other mapping of image pixel values into single values. As an example of the usefulness of these features, the centroid of the image is expressed by first-order moments M_{10} and M_{01}, and the direction of the principal axis of the image can be expressed by means of second-order moments, M_{20}, M_{11}, and M_{02} [41]. The Gram-Schmidt orthonormalization procedure can convert the basis of moments

into the orthonormal basis of Legendre polynomials, and the reconstruction of the image function will be simpler in this case [90]. But it is not certain if the coefficients obtained with Legendre polynomials will have any advantage over moments when used as features for character recognition. Effectively, the new coefficients are obtained by the algebraic transformation of moments and thus have similar discriminating power as moments. Also, the formulas for calculating moments are simpler, and this might be the reason why moments are used more frequently than Legendre polynomial coefficients.

The other type of orthonormal functions which were extensively studied for character recognition are the Zernike polynomials, and the corresponding coefficients were named *Zernike moments* [4, 6, 51, 52, 90]. The Zernike polynomials are defined on the circle area and can be expressed with the help of polar coordinates [51]:

$$V_{nm}(x, y) = V_{nm}(\rho, \theta) = R_{nm}(\rho)e^{im\theta},$$

where n and m are integers, $n \geq 0$, $|m| \leq n$, $n - |m|$ is even and

$$R_{nm}(\rho) = \sum_{s=0}^{(n-|m|)/2} (-1)^s \frac{(n-s)!}{s!\left(\frac{n+|m|}{2} - s\right)!\left(\frac{n-|m|}{2} - s\right)!} \rho^{n-2s}.$$

Intuitively, $R_{nm}(\rho)$ represents an oscillating function along radial direction with the number of oscillations increasing with n, and term $e^{im\theta}$ represents angular oscillations. Polynomials with larger n an m reflect finer details of the image. Figure 11.4 illustrates the approximation of character images by the Zernike moments of different orders (the order is the largest n of participating Zernike polynomials). The polynomials of orders 10–15 are able to approximate the original character images to a sufficient degree, and thus, the corresponding coefficients can be used as features for character recognition. Note that the number of coefficients is bigger than the order of moments, e.g., polynomials of order 10 have 36 coefficients.

Note that other constructions of orthonormal basis functions are possible, which might be well applicable to character recognition. For example, [49] investigates similarly defined on circle area, orthogonal Fourier-Mellin moments and asserts that they outperform Zernike moments.

One of the frequently cited advantages for traditional moments and Zernike moments is that they could be used to construct the scale and rotation invariants [51,90]. Such property would be useful if character images had random orientations, as, for example, characters on topographic maps. But for many available datasets and applications, the character orientation is given. For such well-oriented characters the performance of moment invariants is worse than the performance of original moments [4, 51].

The important consideration in the choice of basis functions for character recognition is their ability to approximate the character images with possibly fewer terms. The visual inspection of moment polynomials suggests that these functions

Fig. 11.4 Original character images and their representation using Zernike moments of orders 5, 10, 15, and 20

can account for the variations in different character images, and the experiments on character recognition confirm that moments, especially Zernike moments, can have sufficiently good performance. But, the construction of the moment polynomials does not depend on the training character sets and might not use the particular handprinted character properties which have been outlined before. For example, the moments do not consider the possibly small stroke width relation to the character size; the character "e" in Fig. 11.4 arguably has better approximation than character "A" due to bigger stroke width. The moments also do not pay attention to structural elements of the characters, strokes, and arcs. At the same time, due to smoothness of polynomial functions, they could represent the similarly smooth foreground regions of the characters quite well. Note that the moments can be used not only for recognizing character images but also any arbitrary smooth shapes and have been used well in these other applications.

Karhunen-Loève (KL) Transform

Instead of using predefined functions for image decomposition, one might want to learn the orthonormal basis from the training data. Karhunen-Loève (KL) transform considers the training images as points in $N = n_1 * n_2$ dimensional space (n_1 and n_2 are the dimensions of the images) and finds a basis $\{v_1, \ldots, v_N\}$ which minimizes the approximation errors of representing training images by their projections into k-dimensional subspace spanned by the first k basis vectors:

$$min||\sum_i I_i - \hat{I_i^k}||\text{, where } \hat{I_i^k} = \sum_{j=1}^{k} a_{ij} v_j \text{ and } a_{ij} =< I_i, v_j > .$$

The solution to this problem is given by the eigenvectors, or principal components, of covariance matrix constructed with the help of training images I_i. The KL transform has been proposed [97] and used extensively for face detection and face recognition.

The use of KL transforms for handprinted character recognition has also been investigated. Grother [35] and Grother and Candela [37] report satisfactory results

on using it for handprinted digit classification. They also suggest that around 50,000 training samples are needed to overcome the difference in the performance on training and testing sets and thus to achieve a full potential of this method of feature extraction. The algorithm has been subsequently released in NIST form processing software [33].

But, when compared to other methods of feature extraction, the KL transform is not performing as well, and consequently it was used rarely for character recognition. The reason for this might be the great variety of character shapes, which KL transform fails to model properly. Indeed, it assumes that the character images occupy some linear subspace in the space of all possible two-dimensional images. Whereas for face images this assumption seems to hold, this is not the case for character images. For example, the average or mean face, used during KL transform, is usually well-defined image representing somewhat average person's face. But the mean image of digits [35] is just an obscure blob with no structure.

Contour Moments

As it was mentioned in section "Representations of Handprinted Text," the contour representation of the character is equivalent to the binary representation but has a one-dimensional nature instead of two dimensional. Consequently, instead of constructing two-dimensional moments of the image, the one-dimensional moments of the contours could be used. In one of the works utilizing this method, Cheng and Yan [16] used one-dimensional function of the distance of contour points from the centroid of the image and extracted Fourier coefficients of this function. Clearly, it is possible to use other one-dimensional functions of contours for deriving features. Contour profile technique looks at either x or y coordinate function of the contour points, and extracts moments from these functions. These and similar shape representation methods are reviewed in [72].

Although the contour moment techniques could be more efficient in representing characters and deriving features than image moment methods, essentially they have the same strengths and drawbacks with respect to utilizing the properties of handwritten characters – they account for smoothness of foreground regions and boundaries, but not for other properties. Therefore, their performance most probably will be worse than the performance of other methods.

Localized Features

One of the main ideas in character recognition is splitting the area of the character image into some subimages and determining the presence of particular features in each of these subimages. Trier et al. [93] uses the term *zoning* for this technique and assumes that most feature extraction methods have zoning varieties. The extraction of features from image zones begins from earliest works on character recognition. Bakis et al. [5] searched for the presence of 4-pixel configurations, roughly representing edges and angles of the characters, in 6 zones of the character images.

Tou and Gonzalez [91] looked for the structural elements, such as strokes arcs or loops, in eight segments of character image.

The technique continues to be used widely in for handprinted character and word recognition. For example, different combinations of gradient and structural binary or float number features are extracted in either 9 or 16 zones of contour representation of character in [86] and used for neural network classification algorithm. The performance of this and similar algorithms is very close to the optimal, and misclassified cases cause the difficulties for human experts as well. The character and character-based word recognizers utilizing this technique have been successfully used for real-life applications [83]. The hierarchical zoning technique is investigated in [46], where the zones possessing foreground regions are expanded to a greater degree and serve to calculate more features. There is some research addressing the problem of finding optimal zoning configurations from the training data [45].

The technique utilizes the property of the uniform distribution of structural elements of characters inside their bounding box and the usual separation of such features. For example, when a 3×3 zoning is used for digit "0," one might get 8 zones on the boundary detecting a presence of some structural elements, say arcs, and each zone will have only one such element detected. Thus, there is little confusion between detected features, and the overall structure of "0" is well represented by the extracted zone features. Without zoning technique, it is difficult to account for the fifth property of handprinted texts proposed in section "Properties of Handprinted Text."

Structural Features

The structural features reflect the elements constituting the handwriting: strokes, arcs, end points, intersections and corners of them, loops, ascenders and descenders, etc. The use of these elements provides possibly a shortest description of a character. Earlier approaches utilized the strings of structural elements for syntactical matching [91]. The assumption for these approaches is that the sequences of elements follow a particular well-defined grammar, and each character follows its own grammar rules. It might also be possible to define the structural elements through the composition of more primitive elements, such as chain code elements or the sides of a polygon approximating the boundary, thus expanding the grammar.

The weaknesses of approaches relying on structural features include the possibly complex structure of characters resulting in complex grammars, the imprecision in extracting the structural elements, and the difficulty in accounting for variations in the positions of the elements. Some of these difficulties were addressed in the later approaches representing the structural elements in the graph and performing elastic matching of the graphs [62, 98].

Elastic matching accounts for the variations in the positions of the elements. But the detection of structural elements can contain errors and should be addressed for better performance as well. Xue and Govindaraju [103] perform the matching of

structural elements with the help of HMMs, which cast the problem of imprecise matching into statistical framework. HMMs have two possible ways to account for such variations: by the variations in the explored paths of the hidden states and by the variability of the observed elements for the same hidden state.

Still, the most effective way to utilize the structural elements is by recording their presence in different zones. For example, Favata and Srikantan [25] construct a vector of bits which are set to 1 if a particular structural element (line, corner, or concavity) is present in 1 of 16 zones of the character image. In this case, it is possible to obtain a fixed length feature vector and perform matching by calculating the distance between two vectors or by applying classification algorithm.

Recognition Approaches

Character Recognition

The character recognition presents a sufficiently difficult problem for machine learning and pattern classification algorithms to continue being in the focus of interest for a research community. The number of the classes is relatively large: 10 classes for digit recognition, 26 or 52 for alphabetic characters, and 36 or 62 for alphanumeric. Note that the English alphabet is implied here; some classes can be grouped together due to little difference in their appearance ("0" and "O," "1," "i," and "l"). Each character might exhibit a significant difference in how it is written by different people at different times, and, at the same time, different characters might have very similar appearance which is frequent source of confusion even for human readers. These properties lead to relatively big intra-class and small interclass distances between extracted feature vectors and hinders the correct classification. Some learning algorithms can also experience difficulties due to unequal numbers of training samples, which is caused by different frequencies of characters in the texts.

The choice of the classification method depends on the nature of extracted features. The set of structural features could have variable size and discrete values which reflect important connections between more primitive features. The classification approaches which are suitable for such features include syntactical matching [77], decision trees [87], elastic matching of graphs [98], and HMMs [103]. The methods in this category could have complex implementations with many parameters requiring heuristic adjustments [87] in addition to using the training samples.

Other types of features, including moments and localized features, take a form of a fixed-length feature vector. For such features, a variety of classification methods operating in N-dimensional feature space could be applied. The methods in this category are generally simpler than methods for matching structural features, and generic implementations of classification algorithms could be utilized with no changes. The parametric classification methods represent the classification decision boundaries or the class densities by some functions depending on parameters and search for the parameters best satisfying training data. As an example, Naive

Bayes classification assumes a normal distribution of class densities with diagonal covariance matrix and the variances determined by the strengths of the features; the classification is done according to Bayes decision rule. Parametric methods typically require a small amount of training data and have been used in earlier handprinted character classification works [5]. The disadvantage of parametric methods is the limited form of decision boundary and, consequently, the possible inability to correctly classify complex patterns.

Whereas the decision surface functions for parametric methods have a predetermined and limited form, the nonparametric methods are more flexible and could approximate the arbitrary decision surface with the help of the large number of training samples. The frequently used nonparametric methods include nearest neighbor classifiers [25], Bayesian classification by modeling the densities of the character classes [39], neural networks [29, 60], and support vector machines (SVM) [61, 65]. Given the improved performance of computers, the nonparametric classifiers became dominant for handprinted character recognition.

Nearest neighbor methods [21] present one of the simplest and possibly most frequently used classification methods for handprinted character recognition. Grother and Candela [37] compare the performance of few types of nearest neighbor, neural network, linear, and quadratic classifiers on the handprinted digit classification problem, where the features are extracted by KL transform. Overall, the nearest neighbor classifiers, as well as probabilistic neural networks operating on similar principle, showed better performance than all other classifiers on this task. As an example of such recognizer, consider the character recognizer based on k-nearest neighbor matching [25]. It records the presence of gradient, structural, and concavity features in 4×4 zones of the character images as a binary 512-dimensional feature vector. The calculation of the distance between tabulated training samples and unknown sample gives bigger weights to set bits in both feature vectors. The variety of distance calculation methods and the methods to increase the matching have been proposed as well [9].

The k-nearest neighbor classification method has obvious drawbacks – large training template storage space and long matching time – but these drawbacks are becoming less important with increased computing hardware capabilities. Its advantages include the ability to perform well with large dimensional feature vectors and non-floating point features [25]. But its main advantage is probably the ability to account well for the diversity of handprinted character shapes: if a particular character shape is not yet tabulated and incorrectly recognized by current training samples, it could be added to the training set, thus allowing future correct classification of similar shapes. The error rate of k-nearest neighbor classifier approaches the error rate of the optimal Bayes classifier with the increased number of training samples. In general, it might be more effective to increase the size of the training set, say by 50 %, than to search for other classification methods in order to achieve the desired error rate reduction. Note that the proper feature extraction is necessary for well performing k-nearest neighbor classifier; if trained directly on the image pixels or low level features, neural networks and SVMs might achieve a better performance [61, 64, 65].

Character-Based Word Recognition

The words are composed from characters, and it is natural to build the word recognition system on the basis of character recognizers. Although it might be possible to perform word recognition based on some global features [68], such as the number of detected strokes, loops, ascenders, or descenders, the performance of such recognizers would be limited. One of the possible approaches to word recognition is presented on Figure 11.5. It is the approach based on the oversegmentation of word image – few consecutive segments are combined together to generate a hypothetical character image which is matched by included character recognizer. All possible combinations of segments are considered and matched to appropriate characters from lexicon words.

Many word recognition algorithms follow this workflow [11, 15, 22, 24, 53, 64, 71, 78, 99]. The numerical strings can also be processed by the similar method [74]. Even if the algorithm does not have an explicit segmentation step, it might perform an implicit segmentation. Some of the HMM-based word recognition approaches [71, 82] use vertical lines to separate the observation frames, which directly corresponds to the segmentation by the same lines.

One of the techniques, which most word recognizers employ, is the use of dynamic programming matching methods resulting in the speedup in the search for best fitting correspondence of segments and characters [53]. The technique basically keeps in memory the results of matching the consecutive subsequences of segments to the subsequences of characters in the lexicon words. Then, the matching of longer subsequences of segments will utilize the results of previous matchings of shorter subsequences. If an HMM is used in word recognizer, a forward-backward algorithm of the same complexity is employed for matching.

In general, there is little difference in the explicit dynamic programming word recognizers [53] and HMM-based recognizers [22]. Both of the approaches could be trained to recognize separate characters and combine the results of character matches into word recognition result. It might be possible to utilize similar features in both approaches, and the time and memory requirements are the same. Probably,

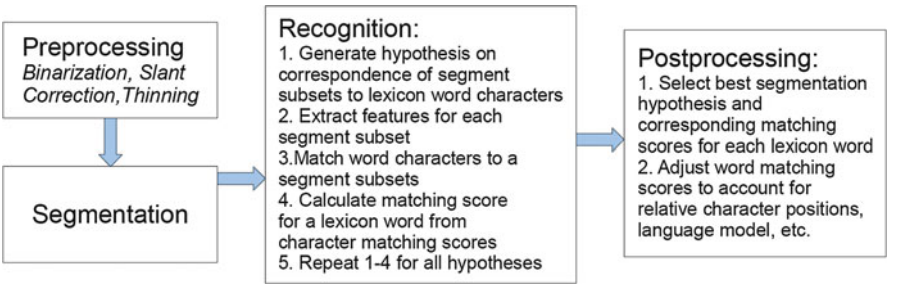

Fig. 11.5 General workflow of a segmentation-based handprinted word recognition algorithm

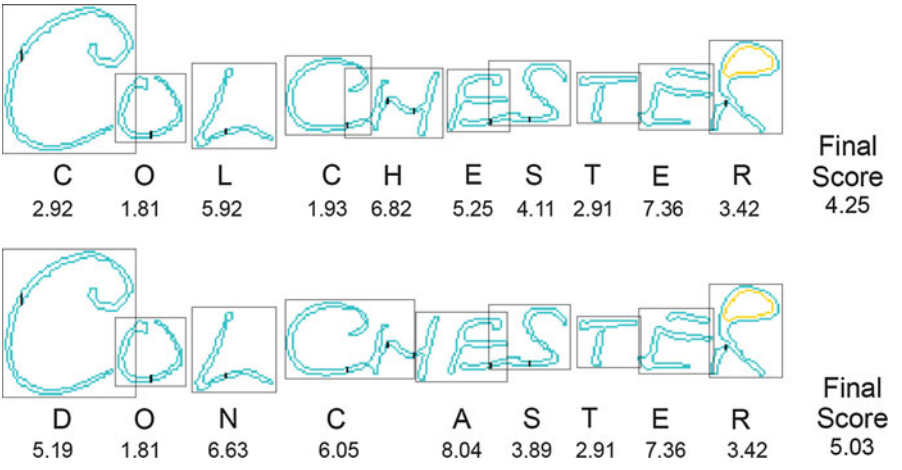

Fig. 11.6 Two results of matching the word image to lexicon words "Colchester" and "Doncaster" by segmentation-based word recognizer [53]. The character matching scores represent the distances from the feature vectors extracted from character subimages to the nearest prototype feature vectors of corresponding training characters. The final matching score is the root mean square of character scores

the main difference appears during the training procedure – an HMM training implicitly accounts for finding best segmentation points separating characters with the objective that segmented characters are most easily recognized, but the segmentation points in the dynamic programming algorithms are determined by the external segmentation algorithm. The characteristics of the segmentation point are automatically included in the matching scores for HMM, but dynamic programming algorithms should incorporate this information explicitly. It is not well ascertained if one type of recognizer performs better than another, and the possible comparison results [71] might be greatly influenced by the different choices of preprocessing steps, features, or matching score calculations in two types of recognizers.

Figure 11.6 presents the results of matching the same image to the two lexicon words by the word recognizer of dynamic programming type [53]. The figure also provides the matching scores for the characters in these words and the segment assignments to the characters. The interesting theoretical question arises with regard to these results: what is the optimal way to combine character scores into word recognition scores? The Bayesian approach suggests that the matching score for a lexicon word should be constructed in form of the geometric mean

$$S \cong \sqrt[k]{\prod_{m=1,\ldots,k} s_m}$$

if the character matching scores s_m are proportional to the likelihood of matching character c_m to the corresponding subimage: $s_m \cong p(\text{subimage}_m|c_m)$ [94].

HMM approaches usually follow the above formula for matching score calculation, but the dynamic programming algorithms might not. As in example of Fig. 11.6, the word matching score is the root mean square of character matching scores. It turns out that taking root mean square for this word recognizer is equivalent to constructing geometric mean of character likelihoods, but even better results could be achieved by training character recognizer to deliver scores proportional to $p(\text{subimage}_m|c_m)$ [94].

Background Modeling

The last two properties of handprinted texts outlined in section "Properties of Handprinted Text" describe the properties of character subimages inside the word image: there is frequently a space or connecting stroke from one character to the next, and the sizes and positions of characters in the same word should be proportional. Most of the word recognizers do not consider the relative sizes and positions of characters, and the final matching score is constructed only from the matching results of each character [53]. The relative sizes and positions of characters can be represented as additional matching scores accounting for the connections between neighboring characters, e.g., for two consecutive letters "a" and "b" of the lexicon word, and for the currently matched subimages of these letters, sub_a and sub_b, one might derive the connection probability $P(\text{"ab"}|\text{sub}_a \cup \text{sub}_b)$ and use this probability in word recognizer's score [30].

The distances between characters are partially accounted for during the segmentation procedure for segmentation-based approaches and during the matching for HMM-based approaches. But, more information could be obtained from, for example, the size of the gap between characters or some other distance measures between them. The gap information might be essential during recognition of phrases, where the gaps between words are more pronounced than the gaps between characters. El-Yacoubi et al. [23] introduce *space HMM* to account for the spaces between the words in an HMM-based phrase recognizer. Park and Govindaraju [79] consider a more detailed and explicit gap treatment in a segmentation-based phrase recognizer, accounting for both inter-character and interword gaps and modifying the matching scores depending on the relationships between gaps and average distances between characters.

The property of character geometry relative to other characters in a handprinted word has also got the attention of researchers. The simple bounding box relationships such as "the average height of character 'f' should be around 1.5 of the average character height" have been considered in [102]. Gader et al. [30] investigated a more complicated method of extracting special *transition* and *bar* features and using a neural network to find the likelihood of two neighboring characters of particular types, such as "ascender, ascender" and "ascender, descender." But the reported performance was similar to a simpler approach relying only on the information about character bounding boxes.

Combinations of Character and Word Recognizers

One of the usual techniques to improve the performance of handprinted character and word recognition system is to incorporate the matching results of few recognizers. Preferably, the combined recognizers would utilize different features extracted from the images and thus will have complementary information useful for fusion. Usually, the character and word recognizers produce match confidences for a finite number of characters or lexicon words, and, therefore, they could be considered as classifiers. Thus, the problem of combining character and word recognition results can be cast as a general problem of classifier combinations [57]. Some of the frequently used combinations methods include voting techniques [59], Dempster-Shafer theory [101], or a variety of combination rules, such as sum, product, min, and max rules [56].

It is possible to distinguish three types of recognition outputs: a single choice of matching class, a ranking order of classes, or a matching score or confidence assigned to each class [101]. The combination methods could deal only with one type of such outputs, e.g., voting methods use single choices of classifiers, Borda count method accepts rank information, and combination rules use matching scores. Since the matching scores deliver the most information about the recognition results, the use of combination rules [56] is generally favored by the researchers. The choice of a particular combination rule, e.g., sum or product, depends on the assumptions on what the matching scores really mean. For example, if the product rule was optimal for the scores of some recognizer, then the sum rule would have been optimal on the logarithms of those scores. As a result, the use of combination methods typically involves some procedure to normalize the matching scores, i.e., to convert them to some prespecified form such as the posterior probability of the class [44]. The efficient normalization procedure might require some analysis on the strength of particular recognizer on a given class set [34].

Despite the large number of published works in the classifier combination field, there is still no consensus on the optimal or, at least, the best performing approach for combining the results of character and word recognizers. For example, it was noted [40] that the rank-based combination methods can have better performance than the matching score combination methods, which apparently utilize more available information. Thus, the conversion of the matching scores to the ranks and performing some rank combinations, e.g., by Behavior Knowledge Space method [42], could potentially give better results than trying to utilize matching scores with combination rules [56].

The recent results in classifier combination research pinpoint the problem to the dependencies of matching scores assigned to different classes [96]. For example, a character recognizer might produce high matching scores for a good quality image and character classes “a,” “d,” and “o,” but low matching scores for a poor quality image and same character set. If the score for “a” is higher than the scores for other characters in both situations, then the rank information will correctly classify image as “a.” But the high matching score for incorrect character “d” in the first

case and low matching score for correct character "a" in the second case might confuse the combination algorithm and result in misclassification. Thus, the optimal combination algorithm should include the model of how the matching scores were generated, but it seems that it is hard to construct such model for each combined recognizer. Instead, it is possible to use some statistical measures derived from the sets of matching scores assigned to different classes during combination in order to account for such dependencies and to achieve a better performance than either rank or traditional rule-based approaches [95].

Conclusion

The reviewed methods of feature extraction and character and word recognition are among the most typical and frequently used, but the list is surely not complete. The written character and word recognition research area is one of the most popular and a great number of works have been published on the topic. The different techniques of preprocessing, feature extraction, and recognition could be combined in many ways, and performance of existing algorithms could be usually improved in multiple ways.

Among this variety, it should be important to understand what makes a particular algorithm weak or strong. In order to do it, this chapter tried to relate the reviewed algorithms to the properties of the handprinted texts. The processing of handprinted texts seems to favor the algorithms which extract the stroke information, use structural features, and utilize the locality of features. The handprinted word recognition could be facilitated by having some character segmentation algorithm, which learns the typical connection patterns between characters. It is also important to have some method for verifying the relative sizes of found characters and their positions.

Many published algorithms fail to take into account all the properties of handprinted texts and thus probably do not have the optimal performance. One of the reasons might be the complexity of implementation and not guaranteed performance improvements. For example, in order to extract some structural features, like a presence of strokes, the algorithm has to binarize image, convert it to chain code representation or extract skeleton, and apply some dedicated algorithms determining if a sequence of points in the chain code or skeleton represents a stroke. Such process is more difficult than, for example, extracting moment features or performing KL transform. The construction of the segmentation algorithm, either heuristic or trainable, is even more complicated. But the utilization of simple properties, such as character size and position distributions, is easy to implement and should be utilized more frequently.

There seems to be an increasing tendency to use advances in machine learning algorithms instead of heuristic processing and feature extraction of handprinted images. The kernel methods, such as support vector machines, can be used for feature extraction or for classification, with the input consisting of original or downsampled pixel image. For word recognition, the use of HMMs seems to become a mainstream approach. The advantage of these methods is the precise

accounting for the variability in the training data, and it is possible that they would be able to construct proper internal representation of characters and account for the presented properties of handprinted texts.

Cross-References

▸A Brief History of Documents and Writing Systems
▸Continuous Handwritten Script Recognition
▸Document Analysis in Postal Applications and Check Processing
▸Imaging Techniques in Document Analysis Processes
▸Page Segmentation Techniques in Document Analysis
▸Recognition of Tables and Forms
▸Text Segmentation for Document Recognition

References

1. Ahmed M, Ward R (2002) A rotation invariant rule-based thinning algorithm for character recognition. IEEE Trans Pattern Anal Mach Intell 24(12):1672–1678
2. Anderson RG (1968) Syntax-directed recognition of hand-printed two-dimensional mathematics. PhD thesis, Harvard University
3. Arica N, Yarman-Vural FT (2001) An overview of character recognition focused on off-line handwriting. IEEE Trans Syst Man Cybern Part C Appl Rev 31(2):216–233
4. Bailey RR, Srinath M (1996) Orthogonal moment features for use with parametric and non-parametric classifiers. IEEE Trans Pattern Anal Mach Intell 18(4):389–399
5. Bakis R, Herbst NM, Nagy G (1968) An experimental study of machine recognition of hand-printed numerals. IEEE Trans Syst Sci Cybern 4(2):119–132
6. Belkasim SO, Shridhar M, Ahmadi M (1991) Pattern recognition with moment invariants: a comparative study and new results. Pattern Recognit 24(12):1117–1138
7. Bernard TM, Manzanera A (1999) Improved low complexity fully parallel thinning algorithm. In: Proceedings of the international conference on image analysis and processing, Venice, 1999, pp 215–220
8. Bernstein MI (1964) Computer recognition of on-line, hand-written characters. Technical report RM-3753-ARPA, The RAND Corporation
9. Bin Z, Srihari SN (2004) Fast k-nearest neighbor classification using cluster-based trees. IEEE Trans Pattern Anal Mach Intell 26(4):525–528
10. Blum H (1967) A transformation for extracting new descriptors of shape. In: Wathen-Dunn W (ed) Models for the perception of speech and visual form. MIT, Cambridge, pp 362–380
11. Bozinovic RM and Srihari SN (1989) Off-line cursive script word recognition. IEEE Trans Pattern Anal Mach Intell 11(1):68–83
12. Bunke H, Wang PS-P (1997) Handbook of character recognition and document image analysis. World Scientific, Singapore
13. Cao H, Govindaraju V (2009) Preprocessing of low-quality handwritten documents using Markov random fields. IEEE Trans Pattern Anal Mach Intell 31(7):1184–1194
14. Casey RG, Lecolinet E (1996) A survey of methods and strategies in character segmentation. IEEE Trans Pattern Anal Mach Intell 18(7):690–706
15. Chen MY, Kundu A, Srihari SN (1995) Variable duration hidden Markov model and morphological segmentation for handwritten word recognition. IEEE Trans Image Process 4(12):1675–1688

16. Cheng D, Yan H (1998) Recognition of handwritten digits based on contour information. Pattern Recognit 31(3):235–255
17. Cheriet M, Kharma N, Liu CL (2007) Character recognition systems: a guide for students and practitioners. Wiley-Interscience, Hoboken
18. Ciresan D (2008) Avoiding segmentation in multi-digit numeral string recognition by combining single and two-digit classifiers trained without negative examples. In: 10th international symposium on symbolic and numeric algorithms for scientific computing (SYNASC'08), Timisoara, pp 225–230
19. Cohen E, Hull JJ, Srihari SN (1991) Understanding text in a structured environment: determining zip codes from addresses. Int J Pattern Recognit Artif Intell 5(1–2):221–264
20. Cohen E, Hull JJ, Srihari SN (1994) Control structure for interpreting handwritten addresses. IEEE Trans Pattern Anal Mach Intell 16(10):1049–1055
21. Dasarathy BV (1991) Nearest neighbor (NN) norms : NN pattern classification techniques. IEEE Computer Society, Los Alamitos
22. El-Yacoubi A, Gilloux M, Sabourin R, Suen CY (1999) An HMM-based approach for off-line unconstrained handwritten word modeling and recognition. IEEE Trans Pattern Anal Mach Intell 21(8):752–760
23. El-Yacoubi MA, Gilloux M, Bertille JM (2002) A statistical approach for phrase location and recognition within a text line: an application to street name recognition. IEEE Trans Pattern Anal Mach Intell 24(2):172–188
24. Favata JT (1996) Character model word recognition. In: Fifth international workshop on frontiers in handwriting recognition, Essex, pp 437–440
25. Favata JT, Srikantan G (1996) A multiple feature/resolution approach to handprinted digit and character recognition. Int J Imaging Syst Technol 7(4):304–311
26. Freeman H (1961) On the encoding of arbitrary geometric configurations. IRE Trans Electron Comput EC-10(2):260–268
27. Freeman H (1974) Computer processing of line-drawing images. ACM Comput Surv 6(1):57–97. 356627
28. Fujisawa H, Nakano Y, Kurino K (1992) Segmentation methods for character recognition: from segmentation to document structure analysis. Proc IEEE 80(7):1079–1092
29. Fukushima K (2003) Neocognitron for handwritten digit recognition. Neurocomputing 51(0):161–180
30. Gader PD, Mohamed M, Chiang J-H (1997) Handwritten word recognition with character and inter-character neural networks. IEEE Trans Syst Man Cybern Part B Cybern 27(1): 158–164
31. Garris MD, Dimmick DL (1996) Form design for high accuracy optical character recognition. IEEE Trans Pattern Anal Mach Intell 18(6):653–656
32. Garris MD, Wilkinson RA (1992) Handwritten segmented characters NIST special database 3, National institute of standards and technology, Gaithersburg, Maryland, USA
33. Garris MD, Blue JL, Candela GT, Dimmick DL, Geist J, Grother PJ, Janet SA, Wilson CL (1994) NIST form-based handprint recognition system, Technical Report NISTIR 5469, National institute of standards and technology, Gaithersburg, Maryland, USA.
34. Govindaraju V, Slavik P, Xue H (2002) Use of lexicon density in evaluating word recognizers. IEEE Trans Pattern Anal Mach Intell 24(6):789–800
35. Grother P (1992) Karhunen Loève feature extraction for neural handwritten character recognition. Proc SPIE 1709(1):155
36. Grother PJ (1995) NIST special database 19-handprinted forms and characters database, National institute of standards and technology, Gaithersburg, Maryland, USA
37. Grother PJ, Candela GT (1993) Comparison of handprinted digit classifiers. (U.S.), National institute of standards and technology, Gaithersburg, Maryland, USA
38. Guo Z, Hall RW (1992) Fast fully parallel thinning algorithms. CVGIP Image Underst 55(3):317–328
39. Hinton GE, Dayan P, Revow M (1997) Modeling the manifolds of images of handwritten digits. IEEE Trans Neural Netw 8(1):65–74

40. Ho TK, Hull JJ, Srihari SN (1994) Decision combination in multiple classifier systems. IEEE Trans Pattern Anal Mach Intell 16(1):66–75
41. Hu M-K (1962) Visual pattern recognition by moment invariants. IRE Trans Inf Theory 8(2):179–187
42. Huang YS, Suen CY (1995) A method of combining multiple experts for the recognition of unconstrained handwritten numerals. IEEE Trans Pattern Anal Mach Intell 17(1):90–94
43. Hull JJ (1994) A database for handwritten text recognition research. IEEE Trans Pattern Anal Mach Intell 16(5):550–554
44. Ianakiev K, Govindaraju V (2002) Deriving pseudo-probabilities of correctness given scores. In: Chen D, Cheng X (eds) Pattern recognition and string matching. Kluwer, Dordrecht/Boston
45. Impedovo S, Lucchese MG, Pirlo G (2006) Optimal zoning design by genetic algorithms. IEEE Trans Syst Man Cybern Part A Syst Hum 36(5):833–846
46. Park J, Govindaraju V, Srihari SN (2000) OCR in a hierarchical feature space. IEEE Trans Pattern Anal Mach Intell 22(4):400–407
47. Jayadevan R, Kolhe S, Patil P, Pal U (2011) Automatic processing of handwritten bank cheque images: a survey. Int J Doc Anal Recognit 15(4):1–30
48. Kamel M, Zhao A (1993) Extraction of binary character/graphics images from grayscale document images. CVGIP Graph Models Image Process 55(3):203–217. 167588
49. Kan C, Srinath MD (2002) Invariant character recognition with Zernike and orthogonal Fourier-Mellin moments. Pattern Recognit 35(1):143–154
50. Kaufmann G, Bunke H (2000) Automated reading of cheque amounts. Pattern Anal Appl 3(2):132–141
51. Khotanzad A, Hong YH (1990) Invariant image recognition by Zernike moments. IEEE Trans Pattern Anal Mach Intell 12(5):489–497
52. Khotanzad A, Hong YH (1990) Rotation invariant image recognition using features selected via a systematic method. Pattern Recognit 23(10):1089–1101
53. Kim G, Govindaraju V (1997) A lexicon driven approach to handwritten word recognition for real-time applications. IEEE Trans Pattern Anal Mach Intell 19(4):366–379
54. Kim G, Govindaraju V (1997) Bank check recognition using cross validation between legal and courtesy amounts. Int J Pattern Recognit Artif Intell 11(4):657–674
55. Kirsch RA, Cahn L, Ray C, Urban GH (1958) Experiments in processing pictorial information with a digital computer. In Papers and discussions presented at the December 9–13, 1957, eastern joint computer conference: Computers with deadlines to meet. ACM: Washington, D.C 221–229
56. Kittler J, Hatef M, Duin RPW, Matas J (1998) On combining classifiers. IEEE Trans Pattern Anal Mach Intell 20(3):226–239
57. Kuncheva LI (2004) Combining pattern classifiers: methods and algorithms. Wiley Inter-Science, Hoboken
58. Kurniawan F, Rehman A, Mohamad D (2009) From contours to characters segmentation of cursive handwritten words with neural assistance. In: International conference on instrumentation, communications, information technology, and biomedical engineering (ICICI-BME), Bandung, 2009, pp 1–4
59. Lam L, Suen CY (1995) Optimal combinations of pattern classifiers. Pattern Recognit Lett 16(9):945–954
60. Le Cun Y, Jackel LD, Boser B, Denker JS, Graf HP, Guyon I, Henderson D, Howard RE, Hubbard W (1989) Handwritten digit recognition: applications of neural network chips and automatic learning. IEEE Commun Mag 27(11):41–46
61. LeCun Y, Jackel LD, Bottou L, Cortes C, Denker JS, Drucker H, Guyon I, Muller UA, Sackinger E, Simard P, Vapnik V (1995) Learning algorithms for classification: a comparison on handwritten digit recognition. In: Oh JH, Kwon C, Cho S (eds) Neural networks: the statistical mechanics perspective. World Scientific, Singapore, pp 261–276
62. Lee RST, Liu JNK (2003) Invariant object recognition based on elastic graph matching: theory and applications. IOS, Amsterdam/Washington, DC

63. Liu C-L, Nakashima K, Sako H, Fujisawa H (2003) Handwritten digit recognition: benchmarking of state-of-the-art techniques. Pattern Recognit 36(10):2271–2285
64. Liu C-L, Sako H, Fujisawa H (2004) Effects of classifier structures and training regimes on integrated segmentation and recognition of handwritten numeral strings. IEEE Trans Pattern Anal Mach Intell 26(11):1395–1407
65. Liu C-L, Fujisawa H, Marinai S (2008) Classification and learning methods for character recognition: advances and remaining problems. In: Marinai S, Fujisawa H (eds) Machine learning in document analysis and recognition. Volume 90 of studies in computational intelligence. Springer, Berlin/Heidelberg, pp 139–161
66. Madhvanath S, Govindaraju V, Srihari SN (1996) Reading handwritten phrases on U.S. census forms. Int J Imaging Syst Technol 7(4):312–319
67. Madhvanath S, Kim G, Govindaraju V (1999) Chaincode contour processing for handwritten word recognition. IEEE Trans Pattern Anal Mach Intell 21(9):928–932
68. Madhvanath S, Kleinberg E, Govindaraju V (1999) Holistic verification of handwritten phrases. IEEE Trans Pattern Anal Mach Intell 21(12):1344–1356
69. Marti UV, Bunke H (2002) The IAM-database: an English sentence database for offline handwriting recognition. Int J Doc Anal Recognit 5(1):39–46
70. Milewski R, Govindaraju V, Bhardwaj A (2009) Automatic recognition of handwritten medical forms for search engines. Int J Doc Anal Recognit 11(4):203–218
71. Mohamed M, Gader P (1996) Handwritten word recognition using segmentation-free hidden Markov modeling and segmentation-based dynamic programming techniques. IEEE Trans Pattern Anal Mach Intell 18(5):548–554
72. Loncaric S (1998) A survey of shape analysis techniques. Pattern Recognit 31(8):983–1001
73. Oh I-S (1995) Document image binarization preserving stroke connectivity. Pattern Recognit Lett 16(7):743–748
74. Oliveira LS, Sabourin R, Bortolozzi F, Suen CY (2002) Automatic recognition of handwritten numerical strings: a recognition and verification strategy. IEEE Trans Pattern Anal Mach Intell 24(11):1438–1454
75. Otsu N (1979) A threshold selection method from gray-level histograms. IEEE Trans Syst Man Cybern 9(1):62–66
76. Palumbo PW, Swaminathan P, Srihari SN (1986) Document image binarization: evaluation of algorithms. Proc. SPIE 697:278–285
77. Parizeau M, Plamondon R (1995) A fuzzy-syntactic approach to allograph modeling for cursive script recognition. IEEE Trans Pattern Anal Mach Intell 17(7):702–712
78. Park J (2002) An adaptive approach to offline handwritten word recognition. IEEE Trans Pattern Anal Mach Intell 24(7):920–931
79. Park J, Govindaraju V (2002) Use of adaptive segmentation in handwritten phrase recognition. Pattern Recognit 35(1):245–252
80. Plamondon R, Srihari SN (2000) Online and off-line handwriting recognition: a comprehensive survey. IEEE Trans Pattern Anal Mach Intell 22(1):63–84
81. Plötz T, Fink G (2009) Markov models for offline handwriting recognition: a survey. Int J Doc Anal Recognit 12(4):269–298
82. Senior AW, Robinson AJ (1998) An off-line cursive handwriting recognition system. IEEE Trans Pattern Anal Mach Intell 20(3):309–321
83. Setlur S, Lawson A, Govindaraju V, Srihari S (2002) Large scale address recognition systems truthing, testing, tools, and other evaluation issues. Int J Doc Anal Recognit 4(3):154–169
84. Sezgin M, Sankur B (2004) Survey over image thresholding techniques and quantitative performance evaluation. J Electron Imaging 13(1):146–168
85. Srihari SN, Shin Y-C, Ramanaprasad V, Lee D-S (1996) A system to read names and addresses on tax forms. Proc IEEE 84(7):1038–1049
86. Srikantan G, Lam SW, Srihari SN (1996) Gradient-based contour encoding for character recognition. Pattern Recognit 29(7):1147–1160
87. Suen CY, Nadal C, Legault R, Mai TA, Lam L (1992) Computer recognition of unconstrained handwritten numerals. Proc IEEE 80(7):1162–1180

88. Suen CY, Lam L, Guillevic D, Strathy NW, Cheriet M, Said JN, Fan R (1996) Bank check processing system. Int J Imaging Syst Technol 7(4):392–403
89. Tappert CC, Suen CY, Wakahara T (1990) The state of the art in online handwriting recognition. IEEE Trans Pattern Anal Mach Intell 12(8):787–808
90. Teague MR (1980) Image analysis via the general theory of moments. J Opt Soc Am 70(8):920–930
91. Tou JT, Gonzalez RC (1972) Recognition of handwritten characters by topological feature extraction and multilevel categorization. IEEE Trans Comput C-21(7):776–785
92. Toussaint GT, Donaldson RW (1970) Algorithms for recognizing contour-traced handprinted characters. IEEE Trans Comput C-19(6):541–546
93. Trier ØD, Jain AK, Taxt T (1996) Feature extraction methods for character recognition-a survey. Pattern Recognit 29(4):641–662
94. Tulyakov S, Govindaraju V (2001) Probabilistic model for segmentation based word recognition with lexicon. In: 6th international conference on document analysis and recognition (ICDAR 2001), pp 164–167, Seattle. IEEE Computer Society
95. Tulyakov S, Govindaraju V (2008) Use of identification trial statistics for combination of biometric matchers. IEEE Trans Inf Forensics Secur 3(4):719–733
96. Tulyakov S, Wu C, Govindaraju V (2010) On the difference between optimal combination functions for verification and identification systems. Int J Pattern Recognit Artif Intell 24(2):173–191
97. Turk M, Pentland A (1991) Eigenfaces for recognition. J Cogn Neurosci 3(1):71–86
98. Uchida S, Sakoe H (2005) A survey of elastic matching techniques for handwritten character recognition. IEICE Trans Inf Syst E88-D(8):1781–1790
99. Verma B, Blumenstein M (2008) Fusion of segmentation strategies for off-line cursive handwriting recognition. In: Verma B, Blumenstein M (eds) Pattern recognition technologies and applications: recent advances. IGI Global, Hershey
100. Wang K, Yang YY, Suen CY (1988) Multi-layer projections for the classification of similar Chinese characters. In: 9th international conference on pattern recognition, Rome, 1988, vol.2, pp 842–844
101. Xu L, Krzyzak A, Suen CY (1992) Methods of combining multiple classifiers and their applications to handwriting recognition. IEEE Trans Syst Man Cybern 22(3):418–435
102. Xue H, Govindaraju V (2002) Incorporating contextual character geometry in word recognition. In: Proceedings of eighth international workshop on frontiers in handwriting recognition, Niagara-on-the-Lake, 2002, pp 123–127
103. Xue H, Govindaraju V (2006) Hidden Markov models combining discrete symbols and continuous attributes in handwriting recognition. IEEE Trans Pattern Anal Mach Intell 28(3):458–462
104. Ying L, Srihari SN (1997) Document image binarization based on texture features. IEEE Trans Pattern Anal Mach Intell 19(5):540–544

Further Reading

Researchers typically do not separate the tasks of recognizing handprinted data from more cursive handwritten data. As a result, many published algorithms can be applied to both types of data, and it might be hard to say if a particular algorithm is more suitable to deal with handprinted or cursive handwritten characters and words. Thus, the image enhancement and binarization techniques described here and in ▶Chap. 4 (Imaging Techniques in Document Analysis Processes) can be used with both types of text. The extensive overview of feature extraction methods mostly related to handprinted characters is presented in [93]. The character and word recognition methods reviewed in [3, 17, 80] are mostly applicable to handprinted data as well. The methods oriented more towards a cursive handwritten data are described in [12, 81].