Chapter 4

Writer Identification Using Junction Features

Abstract

In this chapter, we propose a novel junction detection method in handwritten images, which uses the stroke-length distribution in every direction around a reference point inside the ink of texts. Our proposed junction detection method is simple and efficient, and yields a junction feature in a natural manner, which can be considered as a local descriptor. We apply our proposed junction detector to writer identification by Junclets which is a codebook-based representation trained from the detected junctions. A new challenging data set which contains multiple scripts (English and Chinese) written by the same writers is introduced to evaluate the performance of the proposed junctions for multi-script writer identification. Furthermore, two other common data sets are used to evaluate our junction-based descriptor. The performance of writer identification indicates that junctions are important atomic elements to characterize the writing styles. The proposed junction detector is applicable to both historical documents and modern handwritings, and can be used as well for junction retrieval.

4.1 Introduction

Singular structural features [Simon and Baret, 1991] are informative elements in visual patterns. Especially, where curvilinear lines form a cross, there exist small, informative areas. Such crossing regions, or junctions in this chapter, are of primary importance for character perception and recognition. The junctions can be categorized into different types such as L-, T-(or Y-) and X-junctions [Parida et al., 1998] according to the number of edges they connect, or the number of branches they have. Fig. 4.1 shows several artificial junctions. Given a combination of them, people can easily recognize the corresponding character. For example, given the junction set \{(a),(b),(c)\} in Fig. 4.1, the character ‘A’ will pop up in our brain. Similarly, the combination of \{(d),(e)\} results in the character ‘F’, putting the set \{(d),(e),(f)\} together will form the character ‘E’, \{(e),(g)\} will form character ‘H’, \{(a),(h),(a)\} will be character ‘M’, and the different arrangement \{(h),(a),(h)\} will be character ‘W’. From this example we can conclude that junctions are important atomic elements for some English characters, and such atomic elements are shared between different characters. For instance, the junction (e) in Fig. 4.1 is shared between ‘H’, ‘E’ and ‘F’.
4. Writer Identification Using Junction Features

Figure 4.1: Artificial junctions. The circle dots are the center points of the junction.

Junctions are also prevalent in handwritten scripts for languages that use the Roman alphabets, some of which have inherent junctions. Since Chinese characters are composed of line-drawing strokes, they naturally contain many junction points (Fan and Wu, 2000). Characters in other scripts probably also contain junctions, such as Arabic characters (Bulacu et al., 2007). Junctions are often the consequence of overwritten curved traces of handwriting, or are the consequence of connecting strokes between characters. Junctions reflect the local geometrical and structural features around the singular, salient points in handwritten texts. Hence, it is natural to use junctions in handwritten document analysis. Liu et al. (Liu et al., 1999) have shown the efficiency of using fork points on the skeletons for Chinese character recognition. It has also been used to extract features for Arabic handwriting recognition (see the survey (Lorigo and Govindaraju, 2006)).

In this study, we take the assumption that junction shapes are not guaranteed to be identical for different writers. Furthermore, even the same characters written in different historical periods contain different junction shapes. Generally, the differences are from three aspects: First, the length of branches of the junctions are variant. Second, the angles between each branch are also different between different writers or in different periods. Third, the type of junctions might be changed. We believe that those differences are caused by individual writing habits which can be considered as one type of biometric feature. Such features can be used for writer identification.

Based on the observations that junctions are prevalent in handwritten documents and they are different when generated by different writers as mentioned above, we propose an approach to detect junctions in handwritten documents and evaluate the performances of using these detected junctions for writer identification. The contributions of this chapter are summarized as follows: (1) We propose a simple yet effective method for junction detection
4.2 Related work

In handwritten documents, our junction detector yields a junction feature, which can be considered as a mid-level feature representation. Furthermore, a new representation of handwritten documents is proposed based on the detected junctions, termed as Junclets, which are the primitive junctions of the document. The main advantage of the proposed method compared to junction detection in line-drawing images in (Pham et al., 2014) is that our proposed junction method can yield a junction feature in a natural manner. In addition, the benefit of the proposed Junclets compared to COntected-COmponent COntour (CO3) (Schomaker and Bulacu, 2004) and Fraglets (Bulacu and Schomaker, 2007) is that it does not rely on any segmentation or line detection which are challenging problems in document images, especially in historical documents where a connected component may span several lines due to touching ascenders and descenders.

4.2 Related work

In natural images, junctions are often detected based on template matching, contours, or gradient distributions. The template-based method for junction detection has been proposed in (Parida et al., 1998), in which the junction detection problem is formulated as one of finding the parameter values of the junctions that yield a junction which best approximates the template data by minimizing an energy function. The energy function has two parts: scale and location of junctions in images and the junction parameters, which are the number of wedges, wedge angles and wedge intensities. In (Sinzinger, 2008), a novel junction detector is proposed by fitting the neighborhood around a point to a junction model, which segments the neighborhood into wedges by determining a set of radial edges. Two energy functions are used for radial segmentation, and junctions with the most energy are selected as junction candidates, followed by junction refinement to suppress the junctions on the straight edges. The contour-based approach (Maire et al., 2008) considers junctions as points at which two or more distinct contours intersect, and junctions are localized based on the combination of local and global contours using the global probability of boundary (gPb) (Martin et al., 2004). Finally, a probability of junction operator is designed to compare the keypoints found by junctions to those detected by the Harris operator. Recently, Xia et al. (Xia et al., 2014) introduced a novel meaningful junction detection method based on the a contrario detection theory, called a contrario junction detection (ACJ). The strength of a junction is defined as the minimum of the branch strengths which is a measurement of the consistency of the gradient directions in an angular sector. Junctions are detected whose strength is greater than a threshold which is estimated by the a contrario approach. Compared to other methods, this approach requires fewer parameters, and is able to inhibit junctions in textured areas.

In (Pham et al., 2014), junctions are computed by searching for optimal meeting points of median lines in line-drawing images. There are three main steps in this method: (1) region of support determination by the linear least squares for 2-junctions, and crossing-points in skeleton lines for n-junctions, where n = 3, 4. (2) distorted zone construction by a circle
centered at candidate junction points whose diameter is equal to the local line thickness. (3) extracting the local topology which is a set of skeleton segments linked with a connected component distorted zone and junction optimization.

Su et al. (Su et al., 2012) propose a method for junction detection in 2D images with linear structures. The Hessian information and correlation matrix measurements are combined to select the candidate junction points. The potential junction branches of candidate junctions are found, based on the idea that the linear structure should have a higher intensity compared to the background in structured images. Then the locations of the junction centers are refined using template fitting at multiple scales. One disadvantage of this method is that it can only detect junctions with three or more branches.

4.3 Junction detection

In a handwritten image, a junction is defined as a structure \( J \) on the text strokes, with a center point and several separated branches, which can be formulated as (Xia et al., 2014): 
\[
\{ p, r, \{ \theta \}_{m=1}^{M}, S(\theta_m) \},
\]

\( p \) is the center point inside the ink, \( r \in N \) is the scale of the junction, \( \{ \theta_1, \ldots, \theta_M \} \) are the \( M \) branch directions around \( p \) which are corresponding to stroke directions, \( M \) is the order of the junction which is always set to 2, 3 or 4, corresponding to L,Y or X-junctions. \( S(\theta_m) \) is the strength of the branch with direction \( \theta_m \). An example of a junction with three branches is shown in Fig. 4.2. In this chapter, the discrete set \( D \) of possible directions \( \theta_m \) is defined as:
\[
D = \left\{ \frac{2\pi k}{N}; k \in \{0, \ldots, N - 1\} \right\}
\]  
(4.1)

Here, \( N \in N \) is the number of directions we considered, which is set to 360 in all experiments in this chapter.

According to the above definition, there are three main procedures of the proposed junction detection approach. Firstly, the candidate center point \( p \) is detected. After that, the strength of a branch in every direction in the discrete set \( D \) is obtained. Finally, candidate branches are found on local maximum directions, followed with several junction refinement

![Figure 4.2: A junction with three branches. Each branch lies on the skeleton line of the stroke.](image-url)
4.3. Junction detection

4.3.1 Pre-processing

The input of our method is a binary document image with skeleton lines. The Otsu thresholding algorithm (Otsu, 1975) is applied in this chapter, which is widely used for modern handwritings (Bulacu and Schomaker, 2007; Brink et al., 2012). Other robust binarization methods, such as the AdOtsu (Moghaddam and Cheriet, 2012) method, could be used for degraded documents depending on the application. Any skeleton extraction method can be used, because we only use the skeleton line for junction candidates detection, not for feature computation.

4.3.2 Detection of candidate junctions

The junctions in handwritten scripts are generated by the crossings of strokes in handwritten documents, hence it is reasonable to select the fork points obtained from the skeletonization process as the candidate center points. A fork point which is always related to a skeleton image is the location where at least three branches of line segments meet. Therefore, as mentioned in (Pham et al., 2014), this approach can only find the $M$-junctions ($M \geq 3$).

We use the method proposed in (Brink et al., 2012) to detect the 2-junction candidate center points. Given a point $p_i$ on the skeleton line, two nearby pixels $p_{i-e}$ and $p_{i+e}$ can be found which have a distance of $e$ pixels from $p_i$ in preceding and succeeding directions. The leg from $p_{i-e}$ to $p_i$ forms an inbound angle $\varphi_1$, and the leg from $p_i$ to $p_{i+e}$ forms an outbound angle $\varphi_2$, as illustrated in Fig. 4.3(a). Then the angle on $p_i$ defined by $p_{i-e}$ and $p_{i+e}$ can be estimated by: $\varphi_{2\pi}(p_i) = \min(\|\varphi_2 - \varphi_1\|, 2\pi - \|\varphi_2 - \varphi_1\|)$. Fig. 4.3(b) shows the value $\varphi_{2\pi}(p_i)$ of different positions from A to B on the line of Fig. 4.3(a). We can see that there is a local minimum where the curvature of the curve is higher (note that a small value corresponds to a high curvature). This point is considered as the candidate 2-junction center $p$. More details will be presented in the following sections.

The detected 2-junction candidate center points, combined with fork points (i.e., points on skeleton lines with at least three 8-connected neighbors), are treated as the detected candidate junction center points. Finally, we randomly remove one of the points when they are close neighbors which is defined as the Manhattan distance of the center points is less than 4.

4.3.3 Branch strength of a junction

The strength of a junction branch $S(\theta)$ will be defined as a measurement of the probability of the direction $\theta$ being one of the branches of the junction. The branch strength is an important measurement for finding the potential branches. It has been computed in different ways for different types of images. For example, in linear structure images, the average...
4. Writer Identification Using Junction Features

![Diagram](image)

Figure 4.3: Computing the angle of a curve. Figure (a) shows how to compute the angle on point \( p_i \) and (b) shows the computed angle on the curve from \( A \) to \( B \).

Intensity is used as the branch strength measurement in gray-scale images [Su et al., 2012]. The consistency of a gradient distribution in a wedge region is used to find the potential branches in natural images [Sinzinger 2008, Xia et al. 2014]. In handwritten documents, junctions are always formed by the intersection of strokes. Therefore, it is natural to use the features of strokes to describe the strength in every direction.

The underlying idea for potential branch detection is that each branch should be one of the strokes which forms the junction, and the corresponding stroke length should be higher than the stroke lengths in neighboring directions. We can therefore consider the stroke length as the branch strength. There are some possible ways to compute such stroke lengths, such as searching the ink pixels following a ray in a certain direction, similar as [Epshtein et al., 2010]. In this chapter, we use a simple and efficient method based on Bresenham’s algorithm [Hearn and Baker 1997] to compute the length of the stroke inspired by [Brink et al., 2012].

Given a reference point (junction candidate center point) \( p = (x, y) \), one end point \((x_e, y_e)\) is found in the direction \( \theta \) by:

\[
\begin{align*}
x_e &= x + l \cdot \cos(\theta) \\
y_e &= y + l \cdot \sin(\theta)
\end{align*}
\]  

(4.2)

Here, the parameter \( l \) signifies the maximum measurable length, and restricts the search space. The length of the stroke can be measured by the trace length on the Bresenham path [Hearn and Baker 1997] starting from reference point \( p = (x, y) \) towards the end point \((x_e, y_e)\). The trace stops if a background (white) pixel \((x_b, y_b)\) is hit and the trace length \( \text{len}(\theta) \) is then computed as the distance from \( p = (x, y) \) to this background pixel \((x_b, y_b)\) by the Euclidean measure:

\[
\text{len}(\theta) = \sqrt{(x - x_b)^2 + (y - y_b)^2}
\]  

(4.3)
4.3. Junction detection

Figure 4.4: Illustration of length computation. The point $(x, y)$ is the reference point (Junction candidate center point), and the point $(x_e, y_e)$ is the end point with direction $\theta$. The point $(x_b, y_b)$ is the first background pixel that is hit when following a Bresenham path from the start point $(x, y)$ to the end point $(x_e, y_e)$. The length of the stroke on direction $\theta$ is the distance from the point $(x_e, y_e)$ to the point $(x_b, y_b)$.

Figure 4.5: An illustration of junction points and the strength of the branch in each direction. The middle figures show the length distribution in polar coordinates, and the right figures show the distribution in a linear coordinate, from 0 to 360 degrees.

Fig. 4.4 gives an illustration of this method.

After computing the stroke length $\text{len}(\theta)$ at direction $\theta$, the strength of a branch can be defined as:

$$ S(\theta) = \text{len}(\theta), \quad \theta \in D $$

(4.4)
The larger the strength of a branch in direction $\theta$, the more likely it is that the branch corresponding to one of the strokes formed the junction. Fig. 4.5 gives an illustration of the computed strength distribution in each direction in $D$ given a reference point.

The strength of a branch computed by Eq. 4.4 is affected by the width of the stroke. The line with maximum length of the stroke is not parallel with the direction of the stroke, but shifted to the diagonal line, see Fig. 4.6(a). This fact results in two local maxima around the stroke direction. One possible solution is to use a smoothing filter to remove the noise, which is applied in (Su et al., 2012). However, designing such filter is difficult because it is related to the width of the stroke.

One observation is that the direction of the stroke line lying on the maximal value of the distance transform (DT) map (Meijster et al., 2000) of the binary stroke is the same as the direction of the stroke, see Fig. 4.6(c). Based on this observation, the strength of a branch is weighted by the summed value of the distance transform $v_{dt}(\theta)$ in direction $\theta$ on the Bresenham path. The value $v_{dt}$ in each direction is normalized by dividing the sum of $v_{dt}(\theta)$ among all the directions in $D$.

$$S(\theta) = w(\theta) \ast \text{len}(\theta), \quad \theta \in D$$

Here, $w(\theta) = v_{dt}(\theta)/\sum_{\theta \in D} v_{dt}(\theta)$. The local maximum point of the weighted strength reflects the direction of the stroke, see Fig. 4.6(d).
Based on the strength of a junction defined as Eq. 4.5, a fast and simple algorithm for junction detection can be developed. Given a point $p$ and the strength $S(\theta)$, $\theta \in D$, the potential branches can be found in direction $\theta$ where the strength function $S(\theta)$ reaches a local maximum. The set of these local maxima can be computed efficiently by non-maximum suppression (NMS) (Neubeck and Van Gool, 2006; Xia et al., 2014).

### 4.3.4 Final junction refinement

Sources of noise are easily introduced in the binarization, skeletonization or other pre-processing operations (see Fig. 4.8). Therefore, some post-processing steps are needed to refine the detected junctions.

1. Remove the branches with short lengths. A branch whose length is less than a certain threshold is discarded, $len(\theta) < \lambda W_{stroke}$. Here, $\lambda$ is a parameter to control the minimum of the length of the branches. For example, the dashed branch in Fig. 4.8(a) can be considered as noise and will be removed by this refinement.

2. Remove overlapping branches. If the distance of two directions of branches $d_{2\pi} = |\theta_i - \theta_j|$ is smaller than $\Delta$, the branch with smaller branch length $len(\theta)$ is discarded. For example, the dashed branch in Fig. 4.8(b) is too closed to the other branch, and will be removed.

3. Suppress the junction on a straight line. We remove the 2-junctions whose branches are opposite ($d_{2\pi}(\theta_1, \theta_2 + \pi) < \Delta$). Fig. 4.8(c) shows an example of a junction that lies on a straight line and will be removed by this constraint.

There are two parameters to refine the junctions: $\lambda$ which controls the minimum length of branches and $\Delta$ which controls the overlapping and straight lines of the branches. The minimum value of $len(\theta)$ is equal to $W_{stroke}$, hence $\lambda$ should be greater than 1. However, if $\lambda$ is too large, meaningful branches will be deleted. Fig. 4.7 shows the detected percentages of different types of junctions with different parameter values of $\lambda$ and $\Delta$ in the CERUG-EN data set (see Data Set section). We suggest to use $\lambda \in [1, 2]$, and in the experiments we fixed it as $\lambda = 1.5$. For the parameter $\Delta$, we suggest the value $\Delta = 0.1\pi$ because it is quite stable in the range $[0.02\pi, 0.2\pi]$.

The last issue is the scale $r$ of the detected junction. Unlike other works (Sinzinger, 2008; Xia et al., 2014; Pham et al., 2014; Su et al., 2012), in our approach the scale $r$ is not involved in the procedures of junction detection. In order to make the method complete, we just set the scale of the junction as the minimum of the length of the branches.

$$r = \min\{len(\theta_m)\} \quad \text{(4.6)}$$

### 4.3.5 Junction feature

Template-based methods for junction detection involve similarity computation between the candidate junctions and the templates (Su et al., 2012). Building an efficient similarity
measurement is a challenging problem. Two types of junction descriptors were proposed in (Wang, Bai, Wang, Liu and Tu, 2010) based on shape context approaches (Belongie et al., 2002) for object recognition. In that paper the method concatenates the shape context features equally sampled on the contour segments of junctions. In this chapter, we consider the normalized distribution of the stroke-length in each direction in $D$ as a feature for the junction $J_i$, which can be defined as:

$$F(J_i) = \{f_0, \ldots, f_{N-1}\}$$

(4.7)

Here, $f_i = \text{len}(\theta_i)/\sum_{j=0}^{N-1}\text{len}(\theta_j)$ is the normalized length of direction $\theta$ and $N = 360$. The dimension of the feature is equal to $N$, which is the number of directions we considered. The last column of Fig. 4.5 gives two examples of the junction features.

There are several advantages of the proposed junction feature:

1. It is a scale-invariant descriptor, which measures the context of a reference point...
inside the ink of texts. It is also easy to extend to a rotation-invariant descriptor by permuting the feature vector starting from some estimated angles, instead of horizontal directions.

(2) It contains the normalized ink width of the junction $J_i$, which can be estimated as:

$$w_{stroke} \approx f_{\theta_{\text{min}}} + f(\theta_{\text{min}} + \pi)$$ (4.8)

Here, $\theta_{\text{min}} = \text{argmin}_i \{f_i\}$ is the minimum value of the junction feature. The ink width has been shown as a powerful source of information for stroke determination (Newell and Griffin, 2014).

(3) It also contains the normalized ink length on each branch direction of the junction.

(4) To the best of our knowledge, it is the first local descriptor in handwritten document analysis which can be used for matching, and recognition like SIFT (Lowe, 2004) does in natural images.

### 4.4 Writer identification

Although a wide variety of features, local or global, have been proposed in the literature to distinguish writing styles, they are finally transformed into global features based on the bag-of-visual-words framework. Using global features to represent handwritten samples is simple and efficient for computation.

We build the probability distribution of the junctions as a global feature for each writer based on a learned codebook, which we termed as *Junclets*, with a similar framework as the traditional approaches to build a probability distribution of local patterns, such as connected-component contours (CO$^3$) (Schomaker and Bulacu, 2004), graphemes (Bulacu and Schomaker, 2007), writing fragments (Siddiqi and Vincent, 2010), and line segment codes (Ghiasi and Safabakhsh, 2013). Compared to the existing methods, our *Junclets* do not need any segmentation, which makes our method more stable and universal for any type of documents. Our basic idea is that the ensemble of junctions can capture the junction details of the handwritings, which reflect the writing styles of the author.

For each data set, the training set is generated by the junctions extracted from one of the handwritings from each writer. The Kohonen SOM 2D method is used to train the junction codebook, which is widely used in other works (Schomaker et al., 2007). We evaluate the performance with different sizes of the codebook in our experimental section. One trained codebook is shown in Fig. 4.9. The writer can be characterized by a stochastic pattern generator, producing a family of basic shapes, such as graphemes (Bulacu and Schomaker, 2007) or junctions in this paper. The individual shape emission probability is computed by building a histogram based on the trained codebook using the top 5 nearest codewords coding method inspired by LLC (Wang, Yang, Yu, Lv, Huang and Gong, 2010). The writer descriptor is computed by normalizing this histogram to a probability distribution that sums to 1.
4.5 Experimental results

4.5.1 Data sets

We used the handwritten documents from the existing widely used data sets, such as Firemaker (Schomaker and Vuurpijl 2000) and IAM (Marti and Bunke 2002), to evaluate the performance of the proposed method for writer identification. In modern times, more and more people can use more than one language, hence writer identification based on different languages is a new challenging problem, which has been studied in (Newell and Griffin).
4.5. Experimental results

Figure 4.10: Detected junctions in a historical document from the Monk system. Note that all the junctions are normalized into a fixed size in order to improve visualization.

2014) among Latin language, such as English, French, German and Greek. However, there is no research report about how the handwriting style is affected by different characters using different alphabets. For example, how the English text written by Chinese people is affected by the way of writing Chinese characters. In order to answer such questions, we collect a new data set which contains multiple scripts (English and Chinese) from the same Chinese writer, called the Chinese-English database of the University of Groningen (CERUG for short). Detailed information of the CERUG data set can be found in the previous chapter.

4.5.2 Junction analysis

This section aims at illustrating the proposed junction detection method with several more visual experiments on historical documents from the Monk system (Van der Zant et al. 2008). Fig. 4.10 presents the junction detection results. In these figures, the red circles and lines represent the junction region and the orientation of the branches respectively, and the white point in the center is the center point of the junction. Observe that our proposed method can accurately detect those junctions through their type, localization and scale in different types or layouts of handwritten historical scripts. We believe that the junctions are the basic elements or features which can be used for document and layout analysis and document classification.

4.5.3 Performance of writer identification

In this section, the performance of writer identification is presented using the detected junctions in CERUG, Firemaker and IAM data sets. We use the popular Top-1 and Top-10 identification rates to evaluate the performance of writer identification. Note that all the data sets contain two samples per writer and writer identification is performed in a “leave-one-out” manner. There are 105 writers in CERUG both in English and Chinese, 250 writers in Firemaker in Dutch, 650 writers in IAM in English.
4. Writer Identification Using Junction Features

Table 4.1: Performance of Junclets (codebook size is 400) on different data sets.

<table>
<thead>
<tr>
<th>database</th>
<th>top1</th>
<th>top10</th>
</tr>
</thead>
<tbody>
<tr>
<td>CERUG-CN</td>
<td>90.4</td>
<td>97.1</td>
</tr>
<tr>
<td>CERUG-EN</td>
<td>87.1</td>
<td>96.2</td>
</tr>
<tr>
<td>CERUG-MIXED</td>
<td>85.7</td>
<td>98.5</td>
</tr>
<tr>
<td>Firemaker</td>
<td>80.6</td>
<td>94.0</td>
</tr>
<tr>
<td>IAM</td>
<td>83.3</td>
<td>94.4</td>
</tr>
</tbody>
</table>

Table 4.2: Performance comparison of Fraglets and the proposed Junclets on writer identification.

<table>
<thead>
<tr>
<th></th>
<th>Firemaker</th>
<th>IAM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Top1</td>
<td>Top10</td>
</tr>
<tr>
<td>Fraglets (Bulacu and Schomaker, 2007)</td>
<td>75</td>
<td>92</td>
</tr>
<tr>
<td>Junclets</td>
<td>80.6</td>
<td>94.0</td>
</tr>
</tbody>
</table>

Performance of Junclets

In this section, we conduct the experiment to evaluate the performance of writer identification with different sizes of the codebook. Fig. 4.11 shows our results obtained on different data sets. It can be noticed that the writer identification rates (Top-1 and Top-10) slightly increase as the size of the codebook increases. The Junclets codebook spans a shape space by providing a set of nearest-neighbor attractors for the junctions extracted from the written samples. As the codebook size increases, the Junclets in the codebook contain more detailed information, and therefore can capture more details of junctions generated by the individual writer. Furthermore, the dimension of the probability distribution is higher for larger codebook sizes, which results in a higher performance. However, from Fig. 4.11 we can find that the performance is slightly degraded when the codebook size is 2500. This is quite natural as a larger size of the codebook results in a larger dimensionality of the representation space which is more sensitive to the variance within the documents from the same writer. For the results reported in this chapter, we used the codebook which contains 400 Junclets. Table 4.1 gives the writer identification performance on different data sets.

We also compare the results of our proposed junclets features with Fraglets (Bulacu and Schomaker, 2007) which are computed by generating a codebook at the grapheme level. Table 4.2 gives the performance of these two methods based on two common data sets. The codebook size of Junclets is 400, which is equal to the one of Fraglets used in (Bulacu and Schomaker, 2007). The results in Table 4.2 show that our proposed Junclets representation provides around 5% and 3% (Top1) better results than Fraglets on the Firemaker and IAM.
4.5. Experimental results

Figure 4.11: The Top-1 (left figure) and Top-10 (right figure) performance of different sizes of the codebook on different data sets.

data sets, respectively.

Performance of feature combinations

To demonstrate the benefits of our proposed junclets representation, we combine our method with other widely used features. In Table 5.4, we present the performance of other existing features and the combination with Junclets on the three data sets. The existing features we selected are: (1) Hinge [Bulacu and Schomaker, 2007]: the joint probability distribution of the orientations of two legs of two contour fragments attached at a common end pixel on the contours. (2) Quill [Brink et al., 2012]: the probability distribution of the relation between the ink direction and the ink width. (3) QuillHinge [Brink et al., 2012]: the combined feature from Hinge and Quill. There are several common parameters in Hinge and Quill: the leg length $r$, the number of ink width bins $p$ and the number of ink angle bins $q$. In their original works [Bulacu and Schomaker, 2007][Brink et al., 2012], those parameters were learned from the training data set. However, in this chapter, we simply fixed them as $r = 7$, $p = 40$, $q = 23$, because the aim is not to show the best performance of those features, but to provide the performance of them combined with Junclets.

We use the weighted combination method between the considered feature and Junclets as: $d = (1 - \lambda) d_c + \lambda d_{\text{junclets}}$, where $d_c$ is the distance of the considered features (one of Hinge, Quill and QuillHinge) and $\lambda$ is the mixing coefficient. In our experiments, we empirically set $\lambda = 0.2$ for Hinge, $\lambda = 0.7$ for Quill, and $\lambda = 0.6$ for QuillHinge. Here we use a high $\lambda$ value for Quill than Hinge because our proposed Junclets representation also contains the stroke width information. Therefore, the impact of Quill is low when combined with the Junclets.

As shown in Table 5.4, the combination of the existing features and our proposed Junclets outperforms both the previous features and the Junclets representation on the three data sets. One interesting observation is that the edge-based features, (Hinge, Quill and QuillHinge)
Table 4.3: The performance of different features and the combination with Junclets.

<table>
<thead>
<tr>
<th>Feature Description</th>
<th>CERUG-CN Top1</th>
<th>Top10</th>
<th>CERUG-CN Top1</th>
<th>Top10</th>
<th>CERUG-CN Top1</th>
<th>Top10</th>
<th>Firemaker Top1</th>
<th>Top10</th>
<th>IAM Top1</th>
<th>Top10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hinge (Bulacu and Schomaker, 2007)</td>
<td>90.8</td>
<td>96.2</td>
<td>12.3</td>
<td>30.0</td>
<td>84.7</td>
<td>95.7</td>
<td>85.8</td>
<td>95.8</td>
<td>86.6</td>
<td>95.2</td>
</tr>
<tr>
<td>Quill (Brink et al., 2012)</td>
<td>82.7</td>
<td>92.3</td>
<td>18.5</td>
<td>48.6</td>
<td>74.8</td>
<td>93.3</td>
<td>60.8</td>
<td>78.8</td>
<td>84.6</td>
<td>93.8</td>
</tr>
<tr>
<td>Quill-Hinge (Brink et al., 2012)</td>
<td>88.5</td>
<td>93.8</td>
<td>45.2</td>
<td>91.0</td>
<td>86.7</td>
<td>98.6</td>
<td>74.0</td>
<td>89.8</td>
<td>90.8</td>
<td>96.5</td>
</tr>
<tr>
<td>Junclets</td>
<td>90.4</td>
<td>97.1</td>
<td>87.1</td>
<td>96.2</td>
<td>85.7</td>
<td>98.5</td>
<td>80.6</td>
<td>94.0</td>
<td>83.3</td>
<td>94.4</td>
</tr>
<tr>
<td>Junclets+Hinge</td>
<td>94.2</td>
<td>97.1</td>
<td>39.1</td>
<td>76.7</td>
<td>95.2</td>
<td>98.6</td>
<td>89.8</td>
<td>96.0</td>
<td>90.6</td>
<td>96.7</td>
</tr>
<tr>
<td>Junclets+Quill</td>
<td>92.3</td>
<td>96.2</td>
<td>86.2</td>
<td>97.1</td>
<td>92.9</td>
<td>100</td>
<td>83.4</td>
<td>95.0</td>
<td>89.4</td>
<td>96.5</td>
</tr>
<tr>
<td>Junclets+Quill-Hinge</td>
<td>92.3</td>
<td>95.2</td>
<td>89.5</td>
<td>97.6</td>
<td>96.2</td>
<td>100</td>
<td>85.2</td>
<td>95.4</td>
<td>91.1</td>
<td>97.2</td>
</tr>
</tbody>
</table>

Table 4.4: The expectation and integrated rates of the line length distribution of three data sets.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Expectation</th>
<th>Firemaker</th>
<th>IAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CERUG-EN</td>
<td>20.3</td>
<td>19.6</td>
<td>19.0</td>
</tr>
<tr>
<td>Integrated rates (L &gt; 100)</td>
<td>0.3933%</td>
<td>0.0082%</td>
<td>0.0479%</td>
</tr>
</tbody>
</table>

do not achieve a good performance on the CERUG-EN data set compared to the Firemaker and IAM data sets. One partial reason is that English texts written by Chinese people have large straight lines compared to those written by native-speaker subjects. We performed an experiment to prove our assumption using a fast line detection method (LSD) (Von Gioi et al., 2010) to detect lines in handwritten document in the Firemaker, IAM and CERUG-EN data sets. A histogram of the line length from 0 to 300 is built based on the detected lines, and the expectation of this empirical distribution is obtained. We also compute the integrated rate which is defined as the sum of the line length probability with the condition that the length is greater than a threshold $T$. We set $T = 100$ in this experiment.

From Table 4.4, we can conclude that (1) the expectation of the line length on the three data sets is almost the same, which means the handwriting samples in those data sets are under the same scale. (2) the integrated probability of line lengths greater than $T = 100$ of CERUG-EN is about 48 times and 8 times higher than the ones in Firemaker and IAM. The results demonstrate that the CERUG-EN is a challenging data set whose handwriting samples contain more long lines.

Multi-script writer identification

In this section, we look at the performance of writer identification between different scripts, especially between Chinese and English. We chose the first page of the CERUG-CN data set, and the first paragraph of the CERUG-EN and CERUG-MIXED data sets from each writer. The Junclets representation is computed based on the codebook trained from CERUG-MIXED because it contains both Chinese and English scripts. The performance of writer
Table 4.5: The performance when using multiple scripts.

<table>
<thead>
<tr>
<th></th>
<th>Chinese/English</th>
<th>Chinese/Mixed</th>
<th>English/Mixed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Top1</td>
<td>Top10</td>
<td>Top1</td>
</tr>
<tr>
<td>Hinge (Bulacu and Schomaker, 2007)</td>
<td>0.48</td>
<td>0.96</td>
<td>11.0</td>
</tr>
<tr>
<td>Quill (Brink et al., 2012)</td>
<td>2.4</td>
<td>24.9</td>
<td>12.9</td>
</tr>
<tr>
<td>QuillHinge (Brink et al., 2012)</td>
<td>0.96</td>
<td>12.9</td>
<td>15.3</td>
</tr>
<tr>
<td>Junclets</td>
<td>90.7</td>
<td>96.7</td>
<td>72.8</td>
</tr>
</tbody>
</table>

Table 4.6: Writer identification performance of different approaches on the IAM and Firemaker datasets.

<table>
<thead>
<tr>
<th>Approach</th>
<th>IAM</th>
<th>Firemaker</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Writers</td>
<td>Top-1</td>
</tr>
<tr>
<td>Wu et al. (Wu et al., 2014)</td>
<td>657</td>
<td>98.5</td>
</tr>
<tr>
<td>Siddiqi et al. (Siddiqi and Vincent, 2010)</td>
<td>650</td>
<td>89</td>
</tr>
<tr>
<td>Bulacu et al. (Bulacu and Schomaker, 2007)</td>
<td>650</td>
<td>89</td>
</tr>
<tr>
<td>Ghiasi et al. (Ghiasi and Salabakhsh, 2015)</td>
<td>650</td>
<td>93.7</td>
</tr>
<tr>
<td>Jain et al. (Jain and Doermann, 2011)</td>
<td>300</td>
<td>93.3</td>
</tr>
<tr>
<td>Proposed</td>
<td>650</td>
<td>91.1</td>
</tr>
</tbody>
</table>

Identification between different scripts is given in Table 4.5. These results show that our proposed Junclets representation achieves much better results, particularly between Chinese and English scripts, on which the Hinge, Quill and QuillHinge fail in this test.

**Comparison with other studies**

We summarized the results of several works about writer identification on IAM and Firemaker data sets in the literature in Table 4.6. Although it is not fair to compare them because some approaches used a subset of the IAM database, Table 4.6 still gives us a good basis for comparison and our proposed method combined with other features is comparable with others. Although the writer identification rate of our method does not achieve the state-of-the-art results, the proposed Junclets representation can work on the challenging CERUG-EN data set and performs writer identification between Chinese and English.

### 4.5.4 Junction retrieval

We present here an additional experiment at application level for junction retrieval. Junction retrieval, which is similar to word-image retrieval (Van Oosten and Schomaker, 2014), is
4. Writer Identification Using Junction Features

4.6 Conclusion

In this chapter, we have introduced a generic approach for junction detection in handwritten documents. The proposed method yields a junction feature in a natural manner, which can be considered as a local descriptor. We apply the detected junctions to writer identification using a compact representation, called Junclets. The proposed Junclets representation which is computed from a learned codebook achieves much better performance for writer identification, especially between English and Chinese scripts on our novel data set. Our proposed method is simple and computationally efficient, and it does not rely on any segmentation, and hence can be used for any type of handwritten documents.

In the first part of this thesis, we studied the writer identification problem by designing textural-based and grapheme-based features. We have found that textural-based features generated following the joint-feature distribution principle gives an improvement on five benchmark data sets studied in this thesis. The junction feature proposed in this thesis also provides a good performance for writer identification, especially for writer identification cross English and Chinese.
In the next part, we will study the historical document dating and localization problems by writer identity and classification. We will not only evaluate the performance of textural-based and grapheme-based features proposed in the first part, but also propose three grapheme-based features which are more powerful for capturing general handwriting style in certain period or location.
Part II

Historical document dating and localization