藏文历史文献粘连字丁串数据库及改进的滴水算法

# 1.简介

班禅作品全集

The complete works of Panchen Lama

字丁串的切分研究在字丁识别系统中具有至关重要的作用，是一个比较传统但仍未完全解决的重要课题，这方面的研究工作从二十世纪八十年代就已经开始。目前，对于数字、英文、中文的切分识别已经取得了不错的效果，在邮政编码识别、银行支票阅读、文本识别等方面有着重要的应用。在字丁切分识别研究中，很少有学者关注藏文。在字丁串的切分任务中，粘连字丁串的处理是最为核心也是最复杂的问题。对于粘连字丁串的切分可大致分为两大类，第一类隐式切分（Implicit segmentation），以滑动窗口的方式从左到右遍历，得到一组序列，然后基于动态规划或隐马尔科夫模型等，选择最佳的识别效果。第二类显示切分（Explicit segmentation），这种切分方式又可进一步分为两类，一类是弱切分，另一类为过切分。（1）弱切分，弱切分算法主要特征是考虑局部信息，最终只会产生一条切分路径，适用于粘连不严重的情况，代表算法有投影[4][5]、滴水算法[6]、蓄水池算法[10]等。这些算法切分效率高，在粘连不严重的情况下能够取得不错的效果。投影在粘连严重（投影峰值点不在粘连区域）的情况下会造成切分错误，滴水算法及其改进算法主要是模拟水滴在重力作用下落的过程，水滴下落的路径形成切割路径，初始降落点的选择对于是否能正确切分具有决定性作用。蓄水池算法认为在字丁串粘连处会形成坑槽，即蓄水池，根据所要切分字丁的特点制定合适的规则将水库穿透即可得到最佳分割路径。（2）过切分，过切分算法会考虑所有可能存在的切分路径，会产生多条切分路径。可大致分为三类：基于前景分析；基于背景分析；基于组合分析。首先利用前景、背景提取轮廓和骨架信息，然后提取所需的特征点，基于这些特征点在粘连字丁串上得到多个候选组合，最后结合字丁识别器、几何模型、语言模型，筛选出最佳的分割路径。相对于弱切分算法，过切分算法切分准确度高，在粘连严重的字丁串中仍能表现出很好的性能。过切分算法核心思想是将粘连的字丁串切分成多个小部件，这样得到的小部件是单个字丁或单个字丁的一部分。原则上所得到的小部件会小于经隐式切分所形成的候选部件，通过组合不同的小部件，形成候选字丁模式，结合语言模型、几何信息对候选模式进行打分或者动态规划，最后得到最优切分路径。

在过切分方法中，常对粘连字丁串的前景骨架、轮廓和背景骨架进行分析，也有学者从多个角度对粘连字丁串进行综合分析，试图找到最佳的切分路径。可将粘连字丁串切分方式大致分为三种：基于前景分析；基于背景分析；基于组合分析。

Digitalization of historical documents can protect literature and improve reading efficiency. Through an optical character recognition system, we can get the content of the literature. OCR system generally includes image preprocessing, character segmentation and character recognition. The research on the segmentation of the touching character string plays an essential role in character segmentation. It is a traditional but not yet fully solved problem, and related researches have been started since the 1980s. At present, the segmentation about touching character strings (usually digital, letters and Chinese character) has achieved satisfactory results, which has important applications in ZIP code recognition, bank check reading and text recognition. In this field, few scholars pay attention to Tibetan historical document.

为了比较不同算法的效率，避免因数据集不同造成的影响，分别有学者建立了粘连字丁串数据库。其中有代表性的有：粘连手写数字串数据库、中文粘连字丁串数据库（CASIA-HWDB-T）。表一显示了两个数据库的规模及相关的存储信息。粘连手写数字串数据库中的数据是根据NIST SD19中单个的数字合并得到的，作者选取了2000图片进行合并得到了273,452张粘连手写数字串。考虑到人工合成粘连字丁串，可能与真实手写字丁串有差异，许亮根据已经标记好的脱机手写文本数据库CASIA-HWDB，抽取了其中的所有粘连字丁串构成了新的数据库。CASIA-HWDB-T包含56,469个粘连字丁串，其中大部分是单粘连丁符串，1,818个是多粘连字丁串。

在这篇文章中，参考文献【】【】，我们通过对藏文历史文献进行连通区域分析，建立了藏文历史文献粘连字丁串数据库。它由X张两个字丁的粘连和X张三个字丁的粘连数据组成，存在多种粘连方式。另外，我们在我们的数据库上利用改进的算法进行了一些实验。

Most of the time, researchers use different dataset to verify the new algorithm. Finally, the algorithm proposed by researchers can display good performance in their datasets. It is not accurate to evaluate the performance of different algorithms on different datasets. To compare the efficiency and performance of different algorithms and avoid the impact of different databases, some scholars have established the touching character string benchmarking database. Handwritten touching digital database(HWD-TD) and offline Chinese touching character string database (CASIA-HWDB-T) are representative. Table 1 shows the size of two databases and related storage information. HWD-TD contains several different kinds of touching and it was generated by connecting 2,000 images of isolated digits extracted from the NIST SD19. Considering the authenticity of artificial synthesis touching character strings, Xu Liang extracted touching character string from CASIA-HWDB and formed a new database. CASIA-HWDB-T contains 56469 touching character string, most of which are single-touch character, and the 1818 are multi-touch character.

Inspired by the work of [] and [], we established a synthetic database of Tibetan touching character string(TB-HST-TCS), which generated from Tibetan history document (The complete works of Panchen Lama). TB-HST-TCS was built by Connected Component Analysis, which was composed of XXX images of two-touching character string and XXXX images of three-touching character string. Three different algorithms have been carried out on our database. In the following chapter, we will introduce our database in detail.

Table 1. The size and related storage information about HWD-TD and CASIA-HWDB-T. CASIA-HWDB-T is further divided according to the number of touching points and the number of touching character string.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Database | | String | Type | Synthesis | Annotated Information |
| HWD-TD | | 273,452 | Digitals | Ture | Labels, Segmentation Point |
| CASIA-HWDB-T | HWDB-ST-P | 48,536 | Chinese, Digits and Letters | False | Labels, Segmentation Point,  Average Stroke Width and String Height |
| HWDB-ST-M | 6,115 |
| HWDB-MT | 1,818 |

# 2.数据库

To the best of our knowledge, no similar database about Tibetan handwritten touching character string have been built so far. Next, we will introduce the collection and annotation information of the database.

到目前为止，还没有公开发表过任何关于藏文粘连字丁串的数据库。下面，我们将介绍我们数据库的建立及标注信息。

## 2.1采集数据

藏文文字通常以一个字母为中心，其余字母以此为基础前后附加和上下叠写，组合成一个完整的字表结构。一个藏字最多可由6个辅音字母组成，如图1所示。在进行藏文切分识别时通常以一个字丁作为识别单位（上下叠写构成的单位）。我们对藏文历史文献《》进行扫描，得到原始的扫描图像。在粘连字丁串抽取中，首先对图像进行人工切分成行图像；然后对行图像进行二值化和降噪处理；最后通过连通区域分析建立了我们的粘连字丁串数据库的原始数据集。

The Tibetan language is usually centered on a single character. The rest characters are written on the top, bottom, left and right sides of the basic character, and all character merge into a complete word. A Tibetan word can be made up of 6 consonant letters. When segmented and recognized Tibetan characters, we usually use the vertical unit as a basic element. We scanned the Tibetan historical documents (The complete works of Panchen Lama) and obtained the original scanned images. In the touching character string extraction, we first manually segment the image into row images; Then binarize row images and reduce noise point. In the binarization process, we mark the foreground pixel to 0 and the background pixel is 1. Due to the cause of the ink diffusion and illumination, we deleted the outliers with pixels less than 30 in foreground pixels; At last we collected the original dataset through Connected Component Analysis.

In the Connected component analysis, considering the overlapping of vowels and stroke break, we combined the algorithm proposed by XX to merge the component. The four nearest neighbor pixels were used to mark row images, and we save the borders and pixels of each connected area. We can mark the four ends of the boundary as , , , respectively. We can assume that the boundary information of two connected regions are (, , , ) and (, , , ), where less than . According to the formula (1) , (2) and (3), we can calculate , and . represents the length of the overlapping of two components. represents the total length of the two components. represents the distance of the centroid of the two components.

Whether the two connected components are merged by metric, where W1 and W2 denote the width of two connected components, respectively.

If > 0, it means that two connected components can be merged. After the whole row images processing is completed, the ratio of the length to width of the average character is calculated. If the length to width ratio of a character is 1.3 times greater than the mean value, it is initially determined to be touching character string. Figure 1 shows the touching character string extracted from row images. As can be seen, the initially extracted touching character string is inaccurate. In subsequent annotation, we will filter out the real touching character string and carry out related markers.



Figure 1. Some images extracted from Tibetan history document, which contains incorrect image. Overlapping characters are marked with red rectangles and the single character is marked by a blue rectangle.

在连通区域分析时，考虑到字丁存在上下叠写的情况，结合XX提出的算法，对联通区域进行合并。首先通过四邻近对行图像进行连通区域标记，并保存每个连通区域的边框和区域内的像素，记：上、下、左、右边界分别为、、、，假设两个连通区域边界分别为（，，，）和（，，，），且<，按照公式（1）（2）和（3），计算，，和，其中ovlp表示两个字丁重叠的长度，span表示两个字丁总的长度，dist表示两个字丁的质心的距离。

两个连通区域是否合并由nmovlp度量，其中w1和w2分别表示两个连通区域的宽度。

若nmovlp > 0，则表示两个连通区域需要合并。将整个行图像处理完成后，统计字丁长宽比的均值。若某个字丁的长宽比大于均值的1.3倍，则初步判定为粘连字丁串。图2展示了从行图像中提取的粘连字丁串。可以看出，初步提取得到的字丁串是不准确的，在后续的标记中，我们会筛选出粘连的字丁串并进行相关的标记。

## 2.2数据库数据

Data annotation

经过进一步的筛选，我们删除了单个字丁和重叠字丁，确定了最终的数据集。通过我们的观察，数据库中绝大部分为两个字丁的粘连。不同的字丁串存在一处或两处粘连，根据粘连形式的不同，我们将藏字粘连形式分为了六类，表格展示了粘连字丁串不同的粘连形式及对应的例子。根据粘连点和粘连字丁个数，我们将数据库分为三个子集：单粘连字丁串、单粘连多字丁串、多粘连字丁串，图展示了三个子数据库的粘连字丁串图像。

After further screening, we removed the single character and overlapping character, and determined the final dataset. All the characters and punctuation in Tibetan language are aligned according to the baseline. This feature is helpful for the segmentation and recognition of Tibetan character. In the segmentation of handwritten digits or Chinese characters, the type of touching will be further classified. And we divided the touch type into 6 categories based on the baseline position and touch style, as shown in Table 2. Through our observation, most of the images in the database are two touching character string. We partition TB-HST-TCS into two sub databases according to the number of character in touching character string: TB-HST-TCS-T and TB-HST-TCS-M. Each image in TB-HST-TCS-T contains two characters and another one is composed of more than two characters, as depicted in Figures 2(a) and 2(b).

Table 2. Touching type of single-touching Tibetan character pair.

|  |  |  |  |
| --- | --- | --- | --- |
| Position | Touching Stroke Relation | Examples | Rate (%) |
| Above Baseline |  |  | 1.37 |
| On Baseline |  |  | 22.36 |
| Below Baseline |  |  | 76.27 |

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |

Figure 2. Examples of touching character string samples extracted from the database: (a) each image contains two characters and; (b) each image contains more two characters.

藏文在书写时所有的字符、分字点和句末符都是根据基线对齐的，这个特点有助于藏文字符的切分和识别。

为了更好的评价算法切分的准确性，我们对粘连字丁串进行了标记，标注信息包括：粘连字丁串的类别，图像的高度、长度，平均笔画宽度，以及候选切分点坐标。通过连接候选切分点可构成最佳的切分路径。图像展示了一个粘连字丁串的标注信息，及在原图像上切分点的位置信息，我们将标注后的信息存储到了与原图像同名的XML文件中。

To accurately evaluate the efficiency of the segmentation algorithm, we have annotated the touching character string. The information of the ground truth includes the baseline (BL), the class label (CL), the height and width of image, the average stroke width (SW), and the coordinate of the candidate segmentation point. BL is an important parameter. The vowels are located above the BL, and other characters are under the BL. Using BL to divide the touching characters into two parts can help to improve the accuracy of segmentation. SW and CL are used to evaluate the accuracy of segmentation and recognition respectively. We saved the annotation information in an XML file with the same name as the original image. Figure XX depicts an example of an XML file for a touching character strings.

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |

Figure XX Example of (a) the annotated information, (b) the touching point (indicated by the red arrow), the baseline (in blue line). The tag TextRegion represents a segmentation path. If the touching character string has two adhesions, TextRegion will has four coordinate points.

## 2.3 数据分析

We have analyzed the database and counted the number of characters, touching points and Multi-touching (a segmentation path has multiple points) in each touching character string, as shown in the table 4. In our database, single-touching character string is about ten times than multi-touching character string. For TB-HST-TCS-M, each touching character string has 2.03 touching points and 3.11 characters.

Table.4 Statistics of TB-HST-TCS according to the number of character in touching character string, an overwhelming majority of which is single-touching character string.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Database | String | Multi-touching | Character | Touch point |
| TB-HST-TCS-T | 5,844 | 427 | 11,688 | 6,300 |
| TB-HST-TCS-M | 1,399 | 163 | 4,350 | 2,835 |
| Total | 7,243 | 690 | 16,038 | 9,135 |

In follow-up investigation, we found a common phenomenon. Due to the degradation of Tibetan historical documents’ quality, the strokes of character are broken, as shown in the figure 12. Because of this reason, when we annotate data, we spend a lot of time to identify adhesive character string. In your processing, you need to confirm a complete character, both character segmentation and recognition.

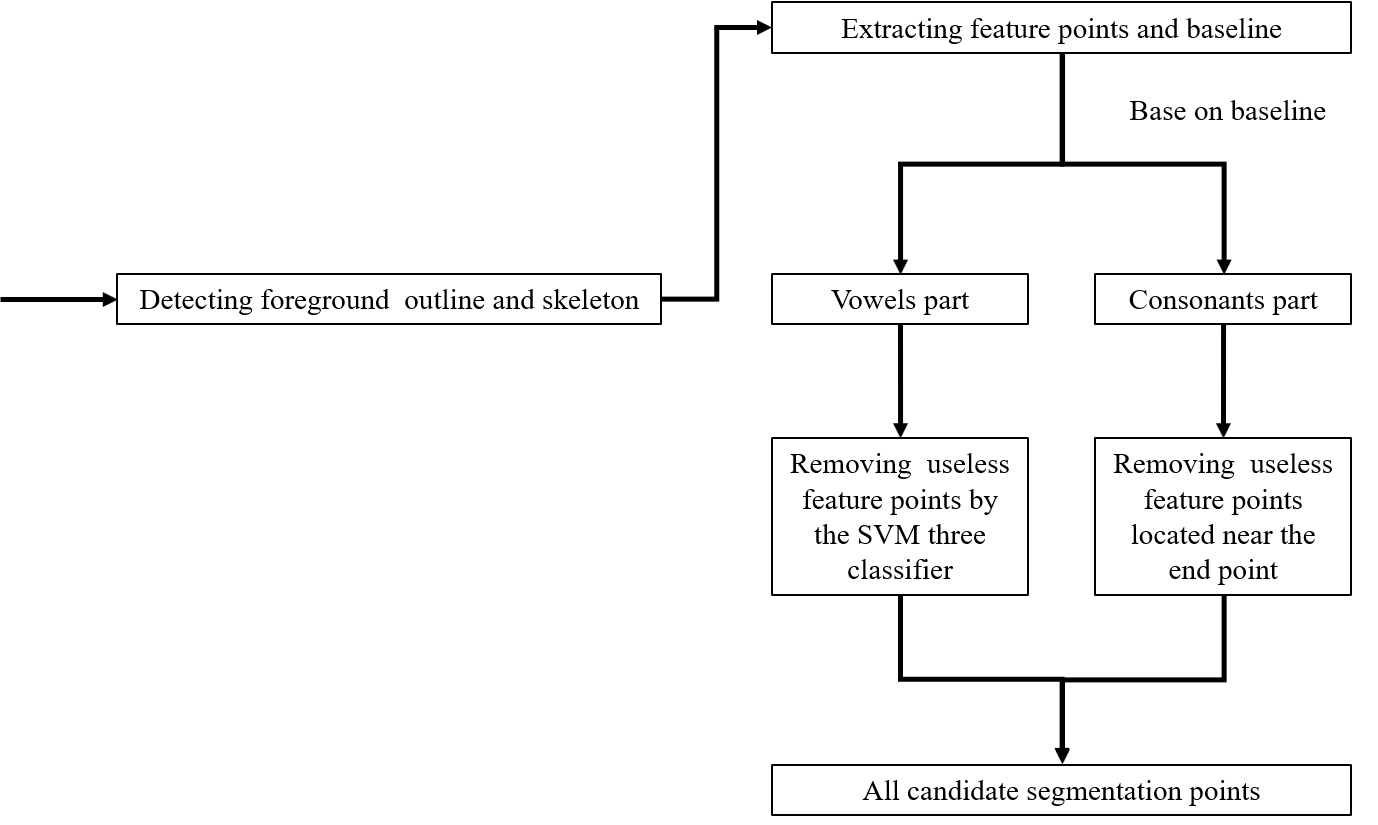
|  |  |
| --- | --- |
|  |  |

Figure 12. Example of the broken strokes in the touching character string (in the red ring).

# 3.算法

The segmentation algorithm can be roughly divided into two categories, implicit segmentation and explicit segmentation []. Implicit segmentation: the text line picture traverses from left to right with a narrow sliding window to obtain a feature sequence. Then based on the HMM of the statement, the character recognition and segmentation results of the entire text line are obtained. Explicit segmentation: according to feature points, the touching character string is cut into several parts. It can be further divided into two categories, one is weak-segmentation and the other is over-segmentation. The main feature of the weak segmentation algorithm is that only one segmentation path is generated, which is suitable for less severe contact. The representative algorithm includes vertical projection [4] [5], drip algorithm [6] water reservoir [10] and so on. The over-segmentation algorithm produces multiple segmentation paths. It can be roughly divided into three categories: foreground-based, background-based, and recognition-based [分类].

We have measured the performance of an over-segmented algorithm on this database for reference. The over-segmented algorithm is a foreground-based combined algorithm.

The flowchart of our algorithm is shown in figure 3. In section 1, foreground outline and skeleton are detected. In section 2, we detect the feature points on the outline by algorithm [] and detect baseline of touching character string. According to the Tibetan baseline, we divide the touching character string into two parts: vowels and consonants. In section 3, we will remove all the useless feature points. For the vowels part, we use feature points directly to segment vowels. We designed an Support Vector Machine (SVM) classifier to predict the probability that the image is a vowel. When the probability of each parts is acceptable, we keep this feature point, otherwise we delete it. For the consonant part, all the feature points located near the end point are deleted according to the skeleton information of the touching character string. In the end, we get all the candidate segmentation points. 

Firstly, we extract contour and skeleton information, and then extract the required feature points (such as corner point and fork point). Based on these feature points, we get multiple candidate combinations on the touching character string. Finally, we combine word recognition, geometry model and language model to select the best segmentation path. Compared with the weak-segmentation algorithm, which has more higher precision and more complexity.

# 4.实验

Experiments

To get better segmentation result, the touching pattern in the touching character string should be recognized at first. We extract the connected components by 8-connected regions for each image, and we delete components which width and height less than SW\*2. Figure 13 shows the details and candidate segment points by our algorithm. We can see that our feature points do not appear in pairs. We have chosen two ways to build segmentation paths. When two feature points are located on either side of the stroke, we connect the two feature points to form a segmentation path. In other cases, we cut the strokes directly to form a segmentation path.

|  |  |
| --- | --- |
|  |  |
| （a） | (b) |

Example of (a) foreground outline, skeleton and feature points, (b) segmentation path

We access the performance of the algorithm based on the distance (d) between a touching point and a candidate point. When d is less than a threshold , we think that the candidate point is a correct segmentation point. In our paper, we set equal to 1.4\*SW. We also calculate the recall rate R and the precision rate P [] to evaluate our algorithm, as following.

Table 12 reports the performance of the foreground-based combined algorithm on the proposed database. In our algorithm, we extracted the Tibetan baseline with an accuracy rate of 95%. Since we forcibly split vowels and consonants, the actual segmentation result is better than the calculated value.

Performance of the foreground-based combined algorithm on the database.

|  |  |  |
| --- | --- | --- |
| Database | R (%) | P (%) |
| TB-HST-TCS-T | 87.54 | 27.56 |
| TB-HST-TCS-M | 80.78 | 32.98 |
| Average | 86.60 | 30.25 |

Over-segmentation algorithm can achieve better segmentation results, but too many candidate points will bring expensive calculations. Table 13 reports the average number of candidate points generated by our algorithm and the time to process each file in Python.

The average number of candidate points and time to process each file.

|  |  |  |
| --- | --- | --- |
| Database | Average number | Time for each file (s) |
| TB-HST-TCS-T | 3.22 | 0.0905 |
| TB-HST-TCS-M | 5.21 | 0.1286 |
| Average | 3.60 | 0.0978 |

# 5．参考文献

1. Chen Y K, Wang J F. Segmentation of Single- or Multiple-Touching Handwritten Numeral String Using Background and Foreground Analysis[J]. Pattern Analysis & Machine Intelligence IEEE Transactions on, 2000, 22(11):1304-1317.
2. Mori, S., Suen, C. Y., & Yamamoto, K. (1992). Historical review of ocr research and development. *Proc IEEE,* *80*(7), 1029-1058.
3. Cheriet M, Kharma N, Liu C L, et al. Character Recognition Systems: A Guide for Students and Practitioners[M]// Character Recognition Systems: A Guide for Students and Practioners. 2008.
4. Chen Y K, Wang J F. Segmentation of single- or multiple-touching handwritten numeral string using background and foreground analysis[J]. Pattern Analysis & Machine Intelligence IEEE Transactions on, 2000, 22(11):1304-1317.

in this paper we describe a synthetic database com- posed of 273,452 handwritten touching digits pairs to assess segmentation algorithms. it contains several dif- ferent kinds of touching and it was generated by con- necting 2,000 images of isolated digits extracted from the nist sd19. in order to get a better insight on the proposed database and establish some parameters for further comparisons, we carried out experiments using four state-of-the-art segmentation algorithms.