Pre-Processing and Feature Extraction

Global Features

Luiz S. Oliveira and Marisa Morita

Federal University of Parana (UFPR)
Department of Informatics (DInf)
R. Rua Cel. Francisco H. dos Santos, 100, Curitiba, PR, Brazil
lesoliveira@inf.ufpr.br

1 Introduction

This report describes the activities performed in the last month, which were devoted to the implementation of the global features for month recognition. Besides feature extraction, some routines of pre-processing and segmentation were also implemented. These routines, which have been proved efficient for word recognition, will be used for other feature sets we plan to develop latter in the project.

The source code is available in the SVN repository. Before compiling the code make sure to update the PPROJ variable in the Make_orand_globfeat file available in the directory ./tools/orand/batch/feature/gen and also to update the defines in the global.h file, available in the directory ./tools/orand/inter/include. To compile the source code use

```
make -f Make_orand_globfeat
```

And to execute the program, type

```
./orand_globfeat feat1 globf
```

The program generates three different files for training, validation and testing, respectively. Those files are located in the directory ./tools/orand/param/train/feat1 and ./tools/orand/param/test/feat1.

In the next sections we describe the main modules implemented to detect and extract the feature vector based on global primitives.

2 Pre-Processing

As we will see later in this report, the feature extraction technique is based on a segmentation algorithm that detects some key points in the upper and lower contour. These points serve as anchor for the feature extraction process. It has been demonstrated [1] that the segmentation algorithm works better for slant free words. For this reason we have implemented a simple, but efficient, slant correction technique. It corrects the slant using a shear transformation defined in the following Equation

$$\begin{cases} x' = & x - (y \times \tan(\theta)) \\ y' = & y \end{cases}$$

Another pre-processing that has been proved useful for character recognition is the smoothing. In our implementation we have used the technique proposed in [2], which passes a 3×3 mask (Figure 1) over the image starting in the lower right corner and processes each row moving upwards row by row. The pixel in the center of the mask is the target. Pixels overlaid by squares marked “X” are ignored. If the pixels overlaid by the squares marked “=” all have the same value, i.e., all zero, or all one, then the target pixel is forced to match them, otherwise it is not changed. This test is done four times for each target pixel, once for each possible rotation of the mask. The result is that single-pixel indentations in all edges are filled and single-pixel bumps are removed

X	X	X
=	T	=
=	=	=

Figure 1: 3 × 3 mask used for smoothing.

3 Reference Lines

Ascenders, descenders, and loops are good primitives to recognise handwritten words. The discriminative power can be boosted if the location information is added to the feature vector. To be able to do that, before detecting such primitives it is necessary to define some reference lines. In our case, we have defined five reference lines using simple density histograms: (1) median line, (2) upper baseline, (3) lower baseline, (4) upper line, and (5) lower line, which delimit the upper, median, and lower regions. Usually, the median region contains the lowercase letters and the upper and lower regions possess the ascenders and descenders. Figure 2 depicts those lines.

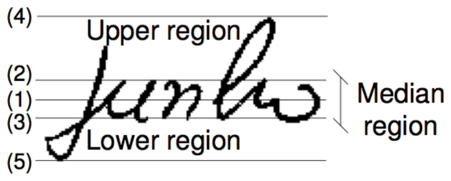

Figure 2: Five reference lines used for feature extraction [3]

4 Segmentation into Graphemes

Segmenting a word into characters is a very complicated task, mainly due to the variability of the handwriting. With this in mind, for segmentation-based word recognition techniques it is important to design a segmentation algorithm that produces several segmentation points where the optimal ones are determined during recognition.

The algorithm we have used uses contour information, loops, reference lines and some heuristics to identify the segmentation points. As output, the algorithm provides a sequence of graphemes where each one may be a correctly segmented, under-segmented or an over-segmented character. Besides the segmentation points produced by the algorithm, a word may have some natural segmentation points. An example of the different segmentation points in presented in Figure 3.

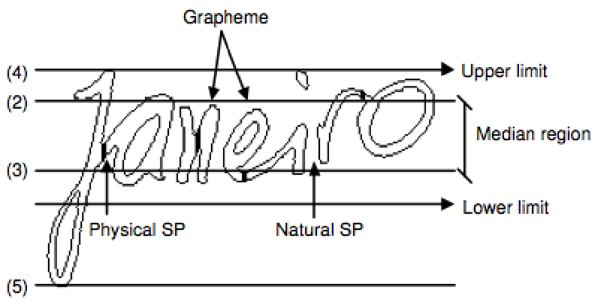

Figure 3: Types of segmentation points that may generate graphemes [3]

5 Feature Extraction

Ascenders, descenders, and loops are detected through the upper contour maxima, lower contour minima, and secondary contours, respectively. These features are usually located in the upper and lower regions, respectively. Because of the variability of the handwriting, it is quite difficult to detect such regions, though.

After detecting the aforementioned features, they are classified into big or small primitives according to their position (maxima and minima) in such regions. The region of small ascenders (descenders) is situated between the upper (lower) limit and this limit decreased (increased) by

40% of the median region height. The remaining of the upper (lower) region is considered the area of big ascenders (descenders). In both cases, the threshold we have used was chosen based on experimentation carried out on the validation set. Therefore, if an upper contour maximum (lower contour minima) is located in the small or big region, it corresponds to a small or big ascender (descender) respectively.

Loops can be identified in any of the three regions (upper, median, and lower) depending on the position of their gravity centres. Loops belonging to the median region are encoded in two ways (big or small). The big loops are those ones greater than the half of the median region height. Thus, the combination of the foregoing primitives plus a primitive that determines if a grapheme does not contain ascender, descender, and loop produces a 20-symbol alphabet. Table 1 describes the symbols that compose the global feature vector.

Table 1: Description of the 20-symbol alphabet

Symbol	Description
O	Big loop in the median region
o	Small loop in the median region
H	Big ascender
h	Small ascender
B	Big descender
b	Small descender
L	Big ascender and a loop in the upper region
l	Small ascender and a loop in the upper region
Z	Big or small descender and a loop in the superior region
-	Absence of ascender, descender, and loops
D	Big ascender and a big or small loop in the median region
d	Small ascender and a big loop in the median region
s	Small ascender and a small loop in the median region
Q	Big descender and a big or small loop in the median region
q	Small descender and a big or small loop in the median region
K	Big or small ascender, a loop in the upper region, and a big or small loop in the median region
G	Big or small descender, a loop in the upper region, and a big or small loop in the median region
M	Big or small ascender and descender
F	Big or small descender and a loop in the upper or lower region
x	Big or small descender and a big or small loop in the median region

Figure 4 shows an image with its respective global feature vector.

6 Conclusions

In this report we have described the first feature set developed to recognise handwritten words. It is a segmentation-based feature set that takes into account holistic features such as ascenders,

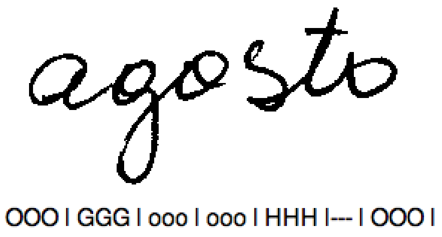

Figure 4: Month image with its respective global feature vector

descenders and loops. Our next activity consists in implementing and training a HMM-based classifier and train it using the described feature set.

References

[1] M. Morita, R. Sabourin, F. Bortolozzi, and C. Y. Suen, "Segmentation and recognition of handwritten dates: an hmm-mlp hybrid approach," *International Journal on Document Analysis and Recognition*, vol. 6, no. 4, pp. 248–262, 2004.

[2] N. W. Strathy, "A method for segmentation of touching handwritten numerals," M.S. thesis, Concordia University, Montreal-Canada, September 1993.

[3] Marisa Morita, *Automatic Recognition of Handwritten Dates on Brazilian Bank Cheques*, Ph.D. thesis, Ecole de Technologie Superieure, Universite du Quebec, 2003.