

Identification of Latin-based Languages through Character Stroke Categorization

Shijian Lu, Linlin Li, Chew Lim Tan

Department of Computer Science, School of Computing
National University of Singapore, Kent Ridge, 117543, Singapore
{lusj, lilinlin, tancl@comp.nus.edu.sg}

Abstract

This paper presents a language identification technique that detects Latin-based languages of imaged documents without OCR. The proposed technique detects languages through the word shape coding, which converts each word image into a word shape code and accordingly transforms each document image into an electronic document vector. For each Latin-based language under study, a language template is first constructed through a corpus-based learning process. The underlying language of the query document is then determined based on the similarity between the query document vector and multiple constructed language templates. Compared with the reported methods, the proposed language identification technique is fast, accurate, and tolerant to text segmentation error caused by noise and various types of document degradation. Experimental results show some promising results.

1. Introduction

With the proliferation of digital libraries, an increasing number of document images of different languages are being produced. Optical character recognition (OCR), which converts imaged text into electronic text, is required to facilitate the management of these produced document images. Though most generic OCR systems are equipped with multiple OCR engines, manual routing is still required to switch the incoming documents to the proper OCR engine. Under such circumstance, a fast and robust language identification functionality is required to switch document images automatically according to the underlying languages.

Several language identification techniques have been reported. The earlier work follows a character shape coding approach, which assigns an electronic code to each labeled character and accordingly converts each word image into a word shape token. For example, the works in [2, 3, 4, 5] pro-

pose to first group character and other text symbol images into seven, ten, six, and thirteen categories, respectively. The techniques following this approach commonly assume that characters are correctly segmented. Besides, the coding process is normally quite slow due to the connect component labeling involved. To overcome the heavy reliance over text segmentation, some work [1, 6] have been reported to avoid the character coding process and capture the shape of an entire word image directly. The coding schemes following this approach are tolerant to text segmentation errors. However, the coding process is still slow due to the connected component labeling.

In this paper, we present a language identification technique that detects Latin based languages based on the observation that documents of the same language normally share a large amount of high-frequency language-specific stop words. A novel word shape coding scheme is proposed, which converts each word image into a word shape code and so each document image into a document vector. Languages are then determined based on the similarity between the converted document vector and multiple pre-constructed language templates. The proposed technique is tolerant to character segmentation errors and requires no connected component labeling at the same time.

2. Word Shape Coding

This section presents a novel word shape coding scheme. Given a document image, text lines and word images are first segmented. Character strokes are then classified into eight categories. Each word image is thus converted into a word shape code based on the categorization result.

2.1 Text Line and Word Segmentation

Words and text lines need to be located first. We locate words and text lines through the analysis of the horizontal and vertical projection profiles. As Figure 1 shows, for

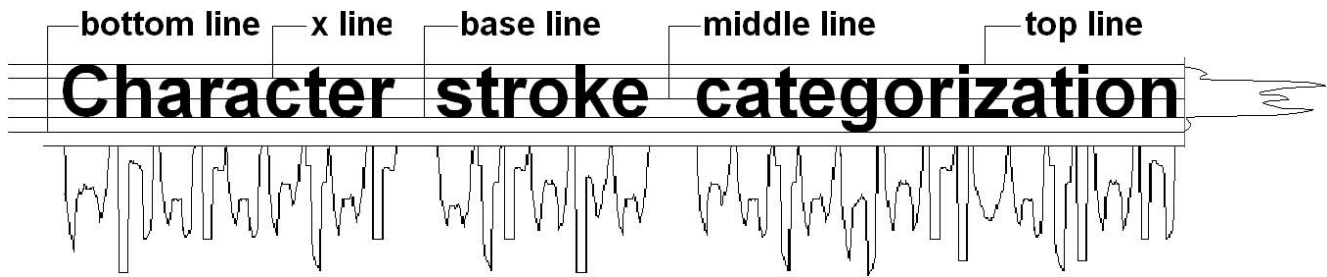


Figure 1. The detection of words and text lines through the projection profile analysis and the illustration of the top line, bottom line, x line, base line, and middle line of text.

Roman document images, the horizontal projection profile normally shows two peaks at the x line and base line positions. Besides, due to the inter-word blanks, there normally exist a number of zero-height sections in the vertical projection profile of text lines. Words and text lines can thus be located based on the peaks and the zero-height sections of the horizontal and vertical projection profiles. The middle line of text can also be located based on the detected x lines and base lines shown in Figure 1.

2.2 Character Stroke Categorization

We classify character strokes into eight categories for the word shape coding. The desired character strokes are first detected through scanning each located text line at the middle line position from left to right. If the horizontal scan line intersects a text pixel, all text pixels within the same column between the top line and bottom line labeled in Figure 1 are determined as part of the desired character stroke. Consequently, a desired character stroke pertains to all text pixels within a rectangle specified by a black run scanned by the middle line of text in horizontal direction and the top line and bottom line in vertical direction.

For the three word images in Figure 1, the black pixels in Figure 2 show the detected desired character strokes, which can be then classified into eight categories as follows:

- 1): Curved/skewed strokes having text pixels lying below the base line (curved/skewed strokes of “y”, “g”, ...).
- 2): Curved/skewed strokes lying between the x line and base line (curved/skewed strokes of “o”, “c”, ...).
- 3): Curved/skewed strokes having text pixels lying above the x line (curved/skewed strokes of “X”, “Z”, ...).
- 4): Vertical straight strokes stretching below the base line (vertical straight strokes of “p”, “q”, ...).
- 5): Vertical straight strokes lying between the x line and base line (vertical straight strokes of “r”, “n”, ...).
- 6): Vertical straight strokes stretching above the x line (vertical straight strokes of “b”, “h”, ...).
- 7): Vertical straight strokes lying between the x line and

Table 1. The codes of Roman letters and numbers 0-9 under the proposed coding scheme.

Codes	Characters	Codes	Characters
25	a	62	bk
2	cexszY	26	d
6	ftEFIJLPT14	11/14	g
65	h	7	i
8	j	555	m
55	nu	22	ovAVÄ
42	p	24	q
5	r	222	w
12	y	63	BDKR
3	Z23579	33	CSGOQUXôÔ068
66	HÜ	6226	M
626	N	2222	W
37	âä	77	üûñ

base line and having text pixels lying above the x line (vertical straight strokes of “i”, “ü”, ...).

8): Vertical straight strokes stretching below the base line and having text pixels lying above the x line (vertical straight strokes of “j”).

The vertical straight strokes in categories 4, 6, and 8 can be easily differentiated from those curved/skewed ones in categories 1, 2, and 3 because they contain at least one column of solid text pixels stretching above the x line or below the base line. The differentiation between the vertical straight strokes in 5 and 7 and those curved/skewed ones in 1, 2, and 3 is based on the compactness of the desired character strokes evaluated as follows:

$$T = \frac{N_t}{W * H} \quad (1)$$

where W and H refer to the stroke width (length of a black run) and the x height (distance between x line and base

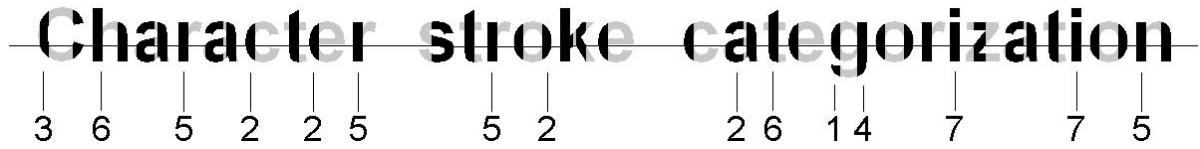


Figure 2. The desired character strokes highlighted by pixels of the black color and the corresponding character stroke codes.

line of text). N_t gives the number of text pixels within the rectangle specified by the W and H . For vertical straight strokes in categories 5 and 7, text pixels almost fill up the rectangle specified by the black run and the x height. But for curved/skewed strokes, there normally exists a large proportion of background pixels within that rectangle. The two characters “r” and “o” within the word image “stroke” in Figure 2 illustrate the two types of character strokes.

2.3 Word Shape Coding

We denote the desired character strokes by the category labels described in the last subsection. Figure 2 shows the codes of some desired character strokes of the three word images. Each character can thus be converted into a digit sequence through the concatenation of the codes of its desired character strokes from left to right. Table 1 shows the codes of Roman letters and numbers from 0 to 9. It should be noted that character “g” may be coded either by “11” in Roman font or by “14” in Arial font. However, other characters/character sequences seldom generate the “11” and “14” in general case. Therefore, “11” and “14” can be treated as equivalent during the word shape coding.

Based on the character stroke coding results described above, a word image can thus be converted into a digit sequence through the concatenation of the codes of the characters it contains from left to right. For the three word images in Figure 2, the corresponding word shape codes can accordingly be represented by three digit sequences including “3365255252625”, “26522622”, and “22562142257225672255, respectively.

3 Language Identification

This section presents the proposed language identification technique. Given a query document image, the corresponding document vector is first built according to the proposed word shape coding scheme. The language of the query document image is then determined based on the similarity between the query document vector and multiple language templates, which are pre-constructed through a corpus-based learning approach.

3.1 Document Vector Construction

Based on the word shaping coding scheme described in the last section, a document image can be converted into a document vector through the coding of the contained word images one by one. In the proposed method, each document vector element is composed of two components, namely, a word shape component corresponding to a word shape code and a word frequency component, giving the frequency of the corresponding word image as follows:

$$DV = [(WSC_1 : WON_1), \dots, (WSC_N : WON_N)] \quad (2)$$

where N refers to the number of unique words within the studied document image. WSC_i and WON_i denote the word shape and word frequency components of the i^{th} document vector element.

The document vector construction process can be summarized as follows. Given a word shape code converted from a word image within the studied document, the corresponding document vector is searched for the element with the same word shape code. If such element exists, the word occurrence number of that element is increased by one. Otherwise, a new document vector element is created and the corresponding word shape code and word occurrence number are initialized with the converted word shape code and one, respectively. The conversion process terminates automatically when all word images have been converted and examined as described above. Finally, the word frequency component is normalized by the number of words within the studied document image.

3.2 Language Template Construction

For each language under study, we construct a language template that takes the same format as the document vector in Equation (2). In particular, we construct language templates by using those high-frequency language-specific stop words based on the observation that documents of the same language normally share a large amount of high-frequency stop words. Four document corpora are created, which are composed of a large number of words collected from different sources including the Internet, proceedings, and some

Table 2. Some selected most frequent words and the corresponding word shape and word frequency separated by “:”.

Language	Stop words	language template elements
English	the, of . . .	6652:0.0502, 226:0.0193 . . .
French	de, la . . .	262:0.0236, 625:0.0334 . . .
German	der, die . . .	2625:0.0348 2672:0.0244 . . .
Italian	di, e . . .	267:0.0375, 2:0.0319 . . .

electronic books.

Language templates are constructed through the transliteration of training words one by one. Starting with an empty language template, each training word within a specific document corpus is first converted into a word shape code according to the coding scheme in Table 1. The language template is then searched for the element with the same word shape code. If such element exist, the corresponding word occurrence number is increased by one. Otherwise, a new language template element is inserted where the word shape and word frequency components are set as the converted word shape code and one, respectively. The learning process terminate until all words within that document corpus have been converted and examined.

After the accumulation process described above, the collected language template elements are sorted according to the word occurrence numbers. Finally, the language template is constructed by using the first K (100 in our system) most frequent elements, most of which correspond to those high-frequency stop words of the corresponding language. For the four Latin-based languages under study, Table 2 shows a few language template elements where column two shows the selected words and column three shows the corresponding word shape codes and word frequency information (normalized by the number of training words).

3.3 Language Identification

Given a query document of one of four languages under study, the underlying language can thus be determined based on the cosine similarity between the query document vector and multiple constructed language templates:

$$sim(Q, LT) = \frac{\sum_{i=1}^N Q_i^{WON} \cdot LT_i^{WON}}{\sqrt{\sum_{i=1}^N (Q_i^{WON})^2 \cdot \sum_{i=1}^N (LT_i^{WON})^2}} \quad (3)$$

where N refers to the size of the language template LT under study and Q denotes the query document vector. LT_i^{WON} denotes the word frequency component of the i^{th} language template element. Q_i^{WON} is determined as follows. For each element within the LT , Q is search for the

Table 3. Collision rates of words of the same and different languages.

	English	French	German	Italian
English	0.0096	0.0252	0.0087	0.0049
French	0.0252	0.0418	0.0061	0.0142
German	0.0087	0.0061	0.0107	0.0028
Italian	0.0049	0.0142	0.0028	0.0113

element with the same word shape component. If such element exist, Q_i^{WON} is set as the corresponding word frequency component. Otherwise, Q_i^{WON} is simply set at zero. Finally, the underlying language of the query document is assigned to the language template with the largest cosine similarity evaluated in Equation (3).

4. Experiments

This section presents experimental results. 80 documents are prepared where every 20 are printed in one of four languages under study. The 80 test documents contain at least 15 text lines each and texts are printed in different fonts including Arial, Roman, Verdana, and so on.

4.1 Word Shape Coding Evaluation

The performance of the proposed word shape coding scheme is first tested. To identify languages properly, the converted word shape codes must be distinguishable between words of different languages. Besides, the coding scheme must be stable across the fonts, noise, and various types of document degradation. We test the proposed coding scheme using the 80 test documents described above. Table 3 shows the coding collision rate, which is defined as the ratio between the number of words sharing word shape codes with others and the number of words coded. As Table 3 shows, collision rates of words of different languages normally lie below 2%, which ensures that languages can be detected properly by the converted document vectors.

The stability of the proposed coding scheme is then tested by using the 80 test documents. For each of the 80 test documents, three document images are created to simulate various document degradation, which are corrupted by gaussian noise ($\sigma = 0.05$), salt & pepper noise (corruption percentage = 0.05) and low scanning resolution (200 dpi), respectively. Experimental results in Table 4 shows that over 90% word images are correctly coded in the presence of the noise and low image resolution.

Table 4. The coding accuracy of the proposed word shape coding scheme in relation to the noise and low image resolution.

	Gaussian	Pepper-Salt	Low Resolution
English	0.9527	0.9256	0.9224
French	0.9591	0.9238	0.9183
German	0.9613	0.9180	0.9209
Italian	0.9477	0.9213	0.9172

4.2 Language Identification Evaluation

The performance of the proposed language identification technique is evaluated by using document vectors of the three sets of document images (80 in each set). Experiments show that the average language identification reaches up to 97.08%. Table 5 shows the average similarity between document of the same and different languages. As Table 5 show, the similarities between documents of the same language (diagonal items) are much higher than those of different languages (off-diagonal items)

The proposed language identification technique depends heavily on the number of words within the query document. We therefore create a set of text line images through cropping from the three sets of document image described above. Experiments show that language can be detected correctly in most cases when query images contain more than six text lines (around 110 word images). However, the identification rate drops quickly below 80% when query document images contain less than three text lines.

4.3 Discussion

The proposed technique is superior to others in speed. Experiments show that the proposed technique is around 6-10 times faster than those reported in [2, 3, 4, 5]. This can be explained by the fact that the proposed coding scheme avoids the time-consuming connected component labeling process. In addition, the collision rate of the proposed coding scheme is much lower than those reported ones as well. As Table 1 shows, the proposed coding scheme classifies Roman letters and 0-9 into 26 categories, which are much bigger than the number of categories reported in [2, 3, 4, 5]. Furthermore, the proposed technique can be easily extended to handle documents of a new Latin language by constructing the corresponding language template beforehand.

Though the proposed technique is fast and accurate, a few limitations exist. Firstly, the proposed technique cannot handle italic text because vertical straight strokes cannot be differentiated from those curved/skewed ones properly. Under such circumstance, text styles need to be detected first

Table 5. The similarity between document vectors of same and different languages.

	English	French	German	Italian
English	0.8802	0.2728	0.2005	0.2148
French	0.3638	0.8233	0.1952	0.3501
German	0.2696	0.2454	0.8444	0.2146
Italian	0.3076	0.3273	0.2218	0.9181

before the word shape coding. Besides, it cannot handle short documents either as described above. We will study these issues in our future work.

5. Conclusion

This paper reports a fast and robust language identification technique. The proposed technique determines languages through the word shape coding, which converts each document image into a document vector. Latin-based languages are then determined based on the similarity between the converted document vectors and multiple pre-constructed language templates. Experiments show that the proposed technique is fast, accurate, and tolerant to noise and various types of document degradation.

6 Acknowledgement

This research is supported by the Agency for Science, Technology and Research (A*STAR), Singapore, under grant no. 0421010085.

References

- [1] S. Lu and C. L. Tan. Script and language identification in degraded and distorted document images. July 2006.
- [2] T. Nakayama. Modeling content identification from document images. *Fourth conference on Applied natural language processing*, pages 22–27, October 1994.
- [3] N. Nobile, S. Bergler, C. Y. Suen, and S. Khoury. Language identification of on-line documents using word shapes. *4th ICDAR*, pages 258–262, 1997.
- [4] A. L. Spitz. Determination of the script and language content of document images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(3):235–245, March 1997.
- [5] C. Y. Suen, S. Bergler, N. Nobile, B. Waked, C. P. Nadal, and A. Bloch. Categorizing document images into script and language classes. *International Conference on Advances in Pattern Recognition*, pages 297–306, 1998.
- [6] C. L. Tan, W. Huang, Z. Yu, and Y. Xu. Image document text retrieval without ocr. *24(6):838–844*, 2002.