

Script and Language Identification in Degraded and Distorted Document Images

Shijian Lu and Chew Lim Tan

Department of Computer Science

National University of Singapore, Kent Ridge, Singapore, 117543

{lusj, tancl}@comp.nus.edu.sg

Abstract

This paper reports a statistical identification technique that differentiates scripts and languages in degraded and distorted document images. We identify scripts and languages through document vectorization, which transforms each document image into an electronic document vector that characterizes the shape and frequency of the contained character and word images. We first identify scripts based on the density and distribution of vertical runs between character strokes and a vertical scan line. Latin-based languages are then differentiated using a set of word shape codes constructed using horizontal word runs and character extremum points. Experimental results show that our method is tolerant to noise, document degradation, and slight document skew and attains an average identification rate over 95%.

Introduction

Script and language identification is normally the first step for multilingual OCR and multilingual information retrieval. Traditionally, script and language identification is frequently addressed in natural language processing areas where the identification is carried out based on the character-coded text (Cavnar & Trenkle 1994) or OCR results (Lee *et al.* 1996). N -grams, which represent n consecutive text symbols, are frequently exploited for script and language identification. With the promise of paper-less office and the proliferation of digital libraries, more and more degraded and distorted document images of different scripts and languages are archived. Script and language identification in these degraded and distorted document images accordingly become a new challenge.

A few script identification techniques have been reported to determine scripts in scanned document images. The reported methods can be classified into three categories, namely, statistics based approaches, token based approaches, and texture based approaches. For statistics based approaches, the distribution of horizontal projection profile is frequently exploited (Ding *et al.* 1997; Pal *et al.*

2002). For document images printed in Roman script, the horizontal projection profile of text normally holds two local peaks at the x -line and baseline positions. But for oriental scripts, horizontal projection profile distributes more evenly from the top to the bottom of text lines. Besides, the distribution of upward concavity (Lee *et al.* 1996; Spitz 1997) is also proposed for script identification.

Texture based methods instead utilize the visual difference of document texture. In the work by Jain *et al.* (1996) Chinese and Roman are differentiated through convolving texture discrimination masks. Later, Tan (1998) constructs textural features using the rotation invariant multi-channel Gabor filter. Similarly, Busch *et al.* (2005) identify scripts based on texture features obtained through the gray-level co-occurrence matrix and wavelet analysis. Compared with the statistics based approach, texture-based approach needs no text segmentation or character component labeling, but the identification process is normally much slower due to the time-consuming texture measurement and analysis. In addition to texture and statistical image features, text tokens (Hochberg 1997) that are specific to different scripts are also exploited for script identification.

A few language identification techniques have also been reported to differentiate languages in scanned document images. Currently, most language identification techniques focus on Latin-based languages. Unlike various scripts that hold different alphabet structures (Spitz, 1997; Hochberg 1997) and texture features (Tan 1998; Busch 2005; Jain *et al.* 1996), all Latin-based languages are printed in the same set of Roman letters and so have similar texture features. Under such circumstance, letter sequences, which are generally organized in differently patterns (words) in different languages, are widely utilized for Latin-based language identification.

The reported language identification methods normally start with a character categorization process, which classifies character images into several categories based on a number of character shape features. For example, the work in (Spitz 1997; Nobile *et al.* 1997; Suen *et al.* 1998) proposes to group character and other text symbol images into six, ten, and thirteen categories, respectively. Based on the character categorization results, word shape codes (*WSC*)

are then constructed and languages are finally identified according to WSC frequency of a single word, word pair, and word trigram. As a common drawback, the performance of these methods depends heavily on character segmentation results. For degraded documents that contain a large quantity of broken or touching characters, the constructed WSC are far from the real ones.

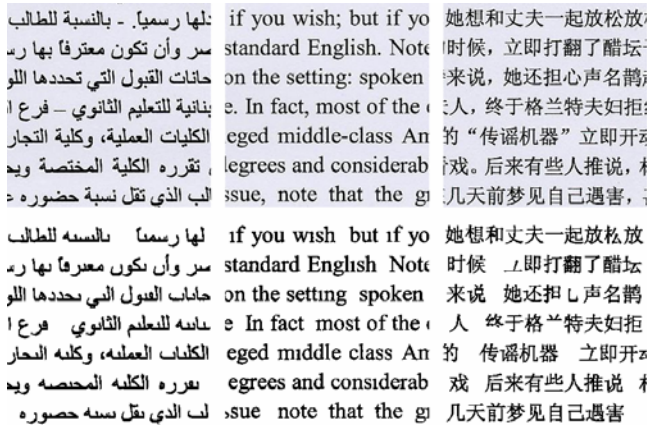


Figure 1: Three document patches in Arabic, Chinese, and English and the preprocessing results

We propose to identify scripts and languages through document vectorization, which transform each document image into a document vector that characterizes the shape and frequency of the contained character and word images. For each script and language studied, we first construct a vector template through a learning process. Script and language of the query image are then determined based on the distances between the query document vector and multiple constructed script or language templates. In our proposed method, we vectorize document images using the number of character runs and character extremum points, which are both tolerant to noise, document degradation, and slight skew. We first exploit the density and distribution of vertical intersections to identify six frequently used scripts including Arabic, Chinese, Hebrew, Japanese, Korean, and Roman. For document images printed in Roman, eight Latin-based languages including English, German, French, Italian, Spanish, Portuguese, Swedish, and Norwegian are then differentiated based on the WSC constructed using the proposed character or word shape features.

Document Preprocessing

Some preprocessing operations are required before the ensuing script and language identification. Firstly, document text must be segmented from the background and we adopt Otsu's (Otsu 1978) global thresholding technique for document binarization. After document binarization, segmented document components are then labeled through connected component analysis. For each labeled document component, information including component size, cen-

teroid, and pixel list can accordingly be determined and stored for later image processing. Lastly, preprocessing is finished through two rounds of size filtering.

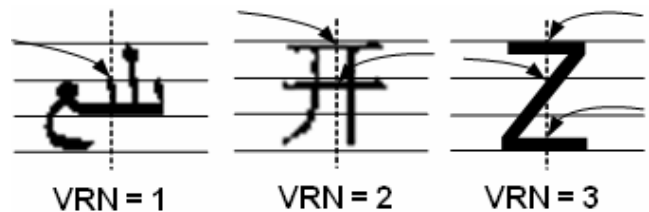


Figure 2: The number and position of vertical runs.

Noise of small size is first removed through the first round of size filtering. We set the threshold at 10 because nearly all labeled character components contain much more than 10 pixels. The second round filtering further removes noise of bigger size and small document components such as punctuations, the top parts of i and j , and character ascent and descent (such as \hat{a} and \ddot{u}) associated with extended Roman alphabets. Filtering threshold is determined as the size of the k^{th} document component so that the between class variance $\sigma_s(k)$ of document component size histogram reaches maximum:

$$\sigma_s(k) = \frac{\varphi(k) \cdot \sum_{i=1}^S i \cdot p(i) + \sum_{i=1}^k i \cdot p(i)}{\varphi(k) \cdot (1 - \varphi(k))} \quad (1)$$

where S refers to the maximum size of document components after the first round of size filtering. $p(i)$ gives the normalized density of component size histogram and $\varphi(k)$ gives the zeroth-order cumulative moment of the histogram. They are determined as:

$$p(i) = n(i) / N, \quad \varphi(k) = \sum_{i=1}^k p(i) \quad (2)$$

where N denotes the number of document components after the first round of size filtering. $n(i)$ gives the number of document components with size equal to i .

For the three document patches on the top of Figure 1, the three patches on the bottom give the preprocessed document images. Currently, most language identification methods depend heavily on small document components such as punctuation and character ascent and descent (\hat{a} and \ddot{u}) for character shape coding. Unfortunately, degraded documents normally contain noise with size similar to these small components, which results in coding inaccuracy. As our method does not require these small components, we just remove them together with noise and so produce a cleaner text image for ensuing feature detection.

Script Identification

We identify scripts based on the density and distribution of vertical character runs. For each script studied, a script template is first constructed through a learning process.

Script Identification

Scripts can therefore be determined based on the distance between the VRV of the query image and the six constructed VRT . We evaluate the distances between the query VRV and the six constructed VRT using Bray Curtis distance, which has a nice property that its value always lies between 0 and 1. The distance between the query document vector and the i^{th} script template is:

$$BCD_i = \frac{\sum_{j=1}^N |VRV_j - VRT_j^i|}{\sum_{j=1}^N |VRV_j| + \sum_{j=1}^N |VRT_j^i|} \quad (5)$$

where VRV_j represents the j^{th} element of the query VRV . VRT_j^i corresponds to the j^{th} element of the i^{th} VRT . As a result, query image is assigned to the script with the smallest Bray Curtis distance.

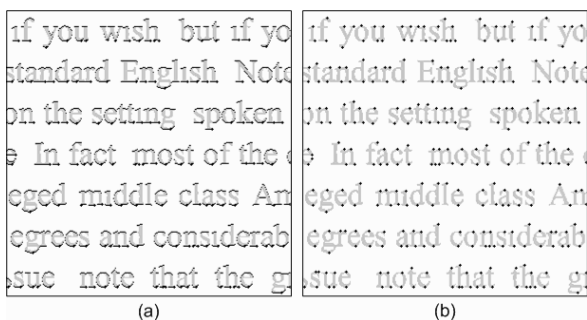


Figure 4: (a) Extracted upward and downward text boundaries; (b) detected character extremum points

Latin-based Language Identification

We identify Latin-based languages using a word shape coding scheme, which transforms each document image into a document vector that characterizes the shape and frequency of the contained word images.

Word Shape Feature Extraction

We exploit two word shape features for word shape coding. The first refers to character extremum points detected from the upward and downward text boundary. The second is the number of horizontal word runs, which counts the intersections between character strokes within a word image and the middle line of text.

For each labeled document components, the upward and downward boundaries can be determined using a vertical scan line that traverses across the document component from the top to the bottom. The first and last character pixels of each scanning round corresponding to the highest and lowest pixels are determined as upward and downward boundary points. For the document patch in English in Figure 1, Figure 4(a) shows the extracted upward and

downward text boundary where texts are printed in light gray for highlighting.

For each labeled document component, its upward or downward boundaries actually form an arbitrary curve that can be characterized by a function $f(x)$. Character extremum points can accordingly be defined as the extrema of the function $f(x)$. For the text boundaries in Figure 4(a), Figure 4(b) shows detected character extremum points. It should be clarified that some single pixel concave and convex along text boundary may affect the extremum point detection. Therefore, these concave and convex with a single pixel must be removed beforehand.

Character extremum point is tolerant to most segmentation errors and slight document skew. For word image “*language*” in Figure 5(a), characters “*g*”, “*u*”, and “*a*” are falsely connected after the binarization process. With traditional character shape coding technique (Spitz 1997; Nobile *et al.* 1997; Suen *et al.* 1998), these three characters will be treated as one and the resulting WSC will be totally different from the real one. But character extremum point is able to capture the word shape correctly while characters are connected as shown in Figure 5(a). Similarly, local extremum points can also be detected correctly in the presence of slight skew as illustrated in Figure 5(b).

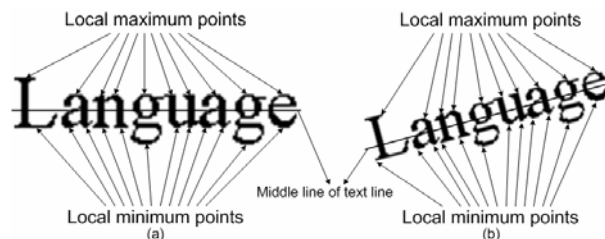


Figure 5: Character extremum points and horizontal word runs in degraded and distorted document image.

Horizontal run number (HRN) refers to the number of intersections between character strokes within a word image and the related middle line of text. For example, the HRN of word image “*the*” is 4 because there are 4 intersections between the character strokes and the related middle line. Similar to the character extremum point, HRN is also tolerant to segmentation errors and slight document skew. For the word image “*language*” in Figure 5, 14 horizontal runs can be correctly detected in the presence of segmentation errors and slight document skew.

Word Shape Coding

We classify character extremum points into three categories based on their positions relative to the x -line and baseline of text. The first category pertains to the extremum points lying far below the baseline. The second category pertains to the ones lying within the region between the x -line and the baseline. The third category pertains to the ones lying far above the x -line. We denote the three categories as 1, 2, and 3, respectively. For document images in

Latin-based languages, the x -line and baseline roughly determine the position of character extremum points.

Combined with the number of horizontal word runs, a word image can thus be transformed to a digit sequence. The first part on the left of the digital sequence is coded based on character extremum points. The following part on the right instead counts the number of intersections between character strokes and the middle line of text. Take the sample word *the* as an example. The corresponding word shape code is 3322/4 where 3322 is coded using character extremum points and the following digit 4 refers to the number of horizontal word runs, respectively.

Each document image can thus be transformed into a word shape vector (WSV) with each element recording a unique WSC . The WSV construction process can be described as follows. For each WSC translated from a word within the studied document image, the corresponding WSV is searched for the element with the same WSC . If such element exists, the occurrence number of the corresponding WSC is increased by one. Otherwise, a new WSV element is created where the WSC component is decided as that newly coded WSC and the word occurrence number is initialized as 1 instead.

A word frequency vector (WFV) can be constructed as well based on the occurrence numbers of the coded WSC . WFV must be normalized first to facilitate the ensuing language identification:

$$WFV_i = \frac{ON_i}{N_w} \quad (6)$$

where WFV_i gives the i^{th} element of the normalized WFV . N_w denotes the number of word images within the studied image and ON_i is the occurrence number of the i^{th} WSC .

Table 2: The number of the WSC learned from the training images (EN-NR: English, French, German, Italian, Spanish, Portuguese, Swedish, and Norwegian).

	EN	FR	GN	IT	SP	PT	SW	RW
EN	7632	1743	1548	1621	1573	1608	1273	1194
FR	1743	8467	1637	1573	1642	1617	1301	1248
GN	1548	1637	9136	1592	1527	1573	1168	1273
IT	1621	1573	1592	7921	1534	1592	1239	1215
SP	1573	1642	1527	1534	8346	1732	1264	1186
PT	1608	1617	1573	1592	1732	8271	1169	1253
SW	1273	1301	1168	1239	1264	1169	8749	1624
RW	1194	1248	1273	1215	1186	1253	1624	8937

Language Identification

We use the proposed word shape coding scheme for language identification. For each language studied, a word shape template (WST) and a word frequency template (WFT) are first constructed. Languages of query images are then determined based on the distance between the query vector and multiple learned language templates.

We construct WST and WFT through a learning process, which remember WSC and the related word frequency

based on a set of training images. 400 document images (every 50 printed in one studied language) are prepared for WST and WFT construction. Table 1 gives the learning results where diagonal items give the numbers of WSC of the eight studied languages and off-diagonal items give the numbers of WSC shared by two related languages. As Table 1 shows, the average WSC collision rate just reaches around 10%. Considering the facts that the learned WSC contain a large number of short frequently appeared words, the 10% collision rate is actually much higher than the real one. It may be reduced greatly after more sample images are trained and some longer WSC are remembered.

Languages can be determined based on WSC information alone. Treating WSV as a data set, languages of query images can be determined based on the number of WSC that are shared between the query WSV and the 8 WST :

$$HD_i = \frac{|WSV \cap WST_i|}{|WSV|} \quad (7)$$

where $|\cdot|$ give the size of a data set. The $|WSV|$, known as a normalization factor, removes the effect of document length difference.

Language can be alternatively determined based on similarity between the query WFV and multiple constructed WFT . We propose to evaluate such similarity using cosine measure:

$$CM_i = \frac{\sum_{j=1}^N WFV_j \cdot WFT_j^i}{\sqrt{\sum_{j=1}^N (WFV_j)^2 + \sum_{j=1}^N (WFT_j^i)^2}} \quad (8)$$

Script of the query image is accordingly determined as the one with the biggest similarity given above.

Table 3: The accuracy of script identification in relation to the number of lines of text processed.

No of Text Lines	No of Samples	Accuracy
12	67	98.51%
10	82	97.56%
8	97	95.88%
6	121	94.21%
4	168	91.07%
2	263	83.65%
1	326	76.54%

Experimental Results

Script Identification

276 testing documents are prepared to evaluate the performance of the proposed script identification method. The testing documents are collected from different sources and each contains around 30 text lines on average. All testing documents are captured using a generic document scanner

where scanning resolution is set at 200 ppi to simulate the document degradation. Experimental results show that the scripts of all 276 testing images are correctly identified.

While the number of characters within query images becomes smaller, the script identification rate may deteriorate as the CIV of the query image cannot reflect the real density and distribution of centroid intersections. We create a set of document samples as given in Table 3 to test the script identification performance in relation to character numbers. All sample images in Table 3 are cropped from the 276 scanned full page document images described above. As Table 3 shows, the identification rate is quite high as the number of text lines is bigger than 4, otherwise the identification accuracy deteriorates quickly.

Table 4: Language identification results

	Detected Languages								Error
	EN	FR	GN	IT	SP	PT	SW	RW	
EN	48	2							2
FR		46		1	3				4
GN			50						0
IT		1		49					1
SP		2			47	1			3
PT		2				48			2
SW							48	2	2
RW			1				1	48	2

Language Identification

To test our proposed language identification method, we prepare 347 testing documents printed in eight studied Latin-based languages. Each document contains 1-40 text lines. Similarly, all testing images are scanned at resolution 200 ppi to simulate document degradation. Table 4 gives the identification results. As Table 4 shows, the language identification rate reaches over 95% on average.

Similar to script identification, the performance of the proposed language identification method is closely related to the number of words within the query images. We note that all testing images containing more than 50 word images are correctly identified. As the word number becomes smaller, the collided *WSC* may dominate the normalized the query *WSV* or *WFV* and this may result in identification errors. Experiments show 15 of the 16 falsely identified testing images in Table 4 contain 20 or less words.

Conclusion

This paper reports a script and language identification technique that differentiates scripts and languages in degraded and distorted document images. In our proposed method, scripts and languages are differentiated through document vectorization, which transforms each document image into an electronic document vector. We vectorize document images using stroke run numbers and character extremum points, which are both tolerant to noise, document segmentation errors, and slight document skew. Ex-

periments show the accuracy of the proposed identification method reaches over 95% on average.

Acknowledge

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

References

- Cavnar, W. and Trenkle, J. 1994. N-Gram Based Text Categorization. *3rd Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, 161-175.
- Lee, D. S., Nohl, C. R. and Baird, H. S. 1996. Language Identification in Complex, Un-oriented, and Degraded Document Images. *International Workshop on Document Analysis Systems*, Pennsylvania, 76-88.
- Spitz, A. L. 1997. Determination of Script and Language Content of Document Images. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 19(3): 235-245.
- Hochberg, J., Kerns, L., Kelly, P. and Thomas, T. 1997. Automatic Script Identification from Images Using Cluster-based Templates. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 19(2): 176-181.
- Tan, T. N. 1998. Rotation Invariant Texture Features and Their Use in Automatic Script Identification. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 20(7): 751-756.
- Busch, A., Boles, W. W. and Sridharan, S. 2005. Texture for Script Identification. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 27(11): 1720-1732.
- Ding, J., Lam, L. and Suen, C., Y. 1997. Classification of Oriental and European Scripts by Using Characteristic Features. *International Conference on Document Analysis and Recognition*, Ulm, GERMANY, 1023-1027.
- Jain, A., K. and Zhong, Y. 1996. Page Segmentation Using Texture Analysis. *Pattern Recognition*, 29(5): 743-770.
- Pal, U. and Chaudhury, B. B. 2002. Identification of Different Script Lines from Multi-script Documents. *Image and Vision Computing*, 20(13-14): 945-954.
- Nobile, N., Bergler, S., Suen, C. Y. and Khoury, S. 1997. Language Identification of Online Documents Using Word Shapes, *4th International Conference on Document Analysis and Recognition*, Ulm, GERMANY, 258-262.
- Suen, C., Y., Bergler, S., Nobile, N., Waked, B. and Nadal, C. P. 1998. Categorizing Document Images into Script and Language Classes, *International Conference on Advances in Pattern Recognition*, Plymouth, UK, 297-306.
- Powalks, R. K., Sherkat, N. and Whitrow, R. J. 1997. Word Shape Analysis for A Hybrid Recognition System, *Pattern Recognition*, 30(3): 421-445.
- Otsu, N. 1978. A Threshold Selection Method from Graylevel Histogram, *IEEE Transactions on System, Man, Cybernetics*, 19(1): 62-66.