2011 International Conference on Document Analysis and Recognition

A Fast Appearance-Based Full-Text Search Method for Historical Newspaper Images

Kengo Terasawa*†, Takahiro Shima* and Toshio Kawashima*
*Graduate School of Systems Information Science, Future University Hakodate
116–2 Kamedanakano, Hakodate, Hokkaido, 041–8655 Japan
Email: {kterasaw,g2109021,kawasima}@fun.ac.jp
†PRESTO, Japan Science and Technology Agency
4–1–8 Honcho, Kawaguchi, Saitama, 332–0012 Japan

***Abstract*—This paper presents a fast appearance-based full-text search method for historical newspaper images. Since historical newspapers differ from recent newspapers in image quality, type fonts and language usages, optical character recognition (OCR) does not provide sufficient quality. Instead of OCR approach, we adopted appearance-based approach; that means we matched character to character with its shapes. Assuming proper character segmentation and proper feature description, full-text search problem is reduced to sequence matching problem of feature vector. To increase computational efficiency, we adopted pseudo-code expression called LSPC, which is a compact sketch of feature vector while retaining a good deal of its information. Experimental result showed that our method can retrieve a query string from a text of over eight million characters within a second. In addition, we predict that more sophisticated algorithm could be designed for LSPC. As an example, we established the Extended Boyer-Moore-Horspool algorithm that can reduce the computational cost further especially when the query string becomes longer.**

***Keywords*-string matching; word spotting; historical document images; Locality-Sensitive Pseudo-Code; Boyer-Moore-Horspool algorithm**

I. Introduction

This paper presents a fast appearance-based full-text search method for historical newspaper images. With the recent development of digital technology, many historical documents are digitized by image scanner and are exhibited for public in image format. Even though efficient indexing or retrieval techniques are indispensable to enhance the utilization of such digital archives, most of retrieval techniques are designed for machine-readable texts rather than images themselves. However, to convert historical newspaper images into machine-readable texts is still a difficult task. Since historical newspapers differ from recent newspapers in image quality, type fonts and language usages, optical character recognition (OCR) does not provide sufficient quality. To construct electronic text with sufficient quality, we need manual corrections by professionals, but it is time and effort consuming work. By such reasons, many valuable documents, except for extremely valuable documents, are not converted to machine-readable texts but stored as images. Under these circumstances, to develop efficient indexing or retrieval techniques for old newspapers is surely important.

In this paper, we introduce the system we have developed for full-text searching of Japanese old newspapers. In cooperation with our civic library, we have applied our system to old newspapers during 1878–1884. In the proposed technique, we adopted pseudo-code expression called LSPC (Locality-Sensitive Pseudo-Code), which is a kind of discretization of high-dimensional vectors. The LSPC discretize vectors into a list of integers with less loss of information compared with usual vector quantization. In our previous study [1], we have demonstrated that LSPC is useful for word spotting of handwritings when used together with CDP (Continuous Dynamic Programming). On the other hand, this paper is the first that reports LSPC is practically applicable to a dataset of large amount of old newspaper images. If each character is properly separated in advance, sequential scanning method is fast enough for full-text searching with LSPC.

We applied our system to a dataset of old newspaper of 3462 pages, and verified that our system can retrieve a query string from a text of over eight million characters within a second while keeping no less than 80% recall ratio.

Since our system is purely appearance-based in the sense that it does not require any prior learning about the particular language, it is, even though originally intended to apply to Japanese documents, ready to apply to any languages as far as character segmentation is possible.

A. Related Work

To the aim of full-text search of historical documents, the most common approach is to attempt OCR and convert images into machine-readable texts. He and Downton [2] adopted off-the-shelf OCR system and post-processing in order to convert more than 500,000 archive card indexes of museum into digital libraries. Drira *et al.* [3] proposed a novel diffusion filter and improved the accuracy of OCR for old printed documents. Kluzner *et al.* [4] developed word-based adaptive OCR for historical books that can tolerate old fonts by gathering word classification technique and state-of-the-art OCR engine.

On the other hand, the idea we have adopted is to avoid character recognition. Instead, we used encoded feature vectors that describe the appearance of the character shapes

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.277

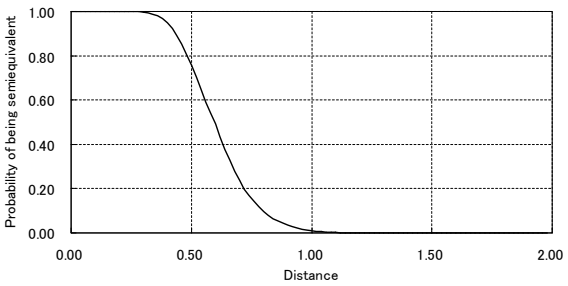

Figure 1. An example of the probability of two pseudo-codes to be semiequivalent with respect to the distance between two vectors.

without character recognition. The idea of vector encoding for document image is found in "Transmedia Machine" [5]. It is a string matching technique based on a pseudo-code for printed English documents. Imura and Tanaka [6] proposed transmedia-based full-text indexing method using an M-tree. C. L. Tan *et al.* [7] developed a document retrieval method using n-gram analysis for the sequence of the class classified by the appearance of the character. Marinai *et al.* [8] used SOM-based clustering for character-like coding and used string edit distance for word retrieval.

II. Locality-Sensitive Pseudo-Code (LSPC)

The most important basis of the proposed method is Locality-Sensitive Pseudo-Code (LSPC) [1]. In this section we first describe the characteristics of LSPC. And then briefly describe how to realize such characteristics. A more detailed description can be found in [1].

A. Outline of LSPC

LSPC is a technology where real vectors can be converted into pseudo-code expressions without significant loss of their information. In contrast to feature vector usually consists of high dimensional vector, LSPC is a list of integers with its length smaller than the dimension of original vector. Even though LSPC is a compact expression than original vector, it still keeps sufficient information of the original vector.

The outstanding characteristics of LSPC is the binary relation called 'semiequivalent.' It is defined that if one of their elements are the same, two code is regarded to be semiequivalent. The above allows the following summary of the definitions of LSPC.

Definition 1: For a d-dimensional vector $\boldsymbol{v}$, LSPC $C(\boldsymbol{v})$ is given as L-tuples of integers:

$$\boldsymbol{v} \in \mathbb{R}^d \mapsto C(\boldsymbol{v}) = (c_1(\boldsymbol{v})\ c_2(\boldsymbol{v})\ \cdots\ c_L(\boldsymbol{v}))^T \in \mathbb{N}^L.$$

Definition 2: Two LSPC $C(\boldsymbol{v}) = \{c_i(\boldsymbol{v})\}$ and $C(\boldsymbol{u}) = \{c_i(\boldsymbol{u})\}$ are regarded to be semiequivalent iff $\exists i$ s.t. $c_i(\boldsymbol{v}) = c_i(\boldsymbol{u})$.

Using this binary relation, we can estimate the distance between original vectors. Figure 1 represents the virtue of LSPC. This figure plots the probability of two pseudo-codes to be semiequivalent with respect to the distance between two vectors. As seen in the figure, it is sensitive to the distance in a sense that if distance between two vector is small, LSPC almost necessarily be semiequivalent, while if distance is large, LSPC almost never be semiequivalent.

Note that to examine two codes are semiequivalent or not requires less computational cost compared to directly calculating the distance between two vectors. This fact is main advantage of LSPC.

B. Composition of LSPC

This subsection describes how to realize the characteristics discussed in the preceding subsection.

The hash value obtained from the Locality-Sensitive Hashing (LSH) [9]–[11] is used for each element of LSPC. LSH is a well-known algorithm for approximate nearest-neighbor-search problem. LSH is based on the family of functions called LSH family defined as follows:

Definition 3: For a domain S of the vector set, LSH family $\mathcal{H} = \{h : S \to U\}$ is a family of functions that satisfies for any $p, q \in S$,

$$\text{if } d(p,q) \le r_1 \text{ then } \Pr_{\mathcal{H}}[h(q) = h(p)] \ge p_1,$$
$$\text{if } d(p,q) \ge r_2 \text{ then } \Pr_{\mathcal{H}}[h(q) = h(p)] \le p_2,$$

where, $d(p,q)$ represents the distance between vectors p and q, and it has to satisfy inequalities $p_1 > p_2$ and $r_1 < r_2$.

The existence of such a family of functions is guaranteed for the L_1 space [9], for the L_s space of any $s \in (0,2]$ [10], and for any dimensional unit hypersphere [12].

A function that belongs to a family of functions $\mathcal{H}$ can be regarded as a hash function in a sense that it maps a vector $\boldsymbol{v} \in \mathbb{R}^d$ onto the integer $h(v) \in \mathbb{N}$. The basic idea of LSPC is to use the hash values of LSH function as a pseudo-code.

Definition 4: LSPC $C(\boldsymbol{v})$ for a vector $\boldsymbol{v}$ is defined as:

$$C(\boldsymbol{v}) = \{\ g_1(\boldsymbol{v}), g_2(\boldsymbol{v}), \ldots, g_L(\boldsymbol{v})\ \}$$
$$g_i(\boldsymbol{v}) = \{\ h_{i1}(\boldsymbol{v}), h_{i2}(\boldsymbol{v}), \ldots, h_{ik}(\boldsymbol{v})\ \}$$

where h_{ij} is a function randomly chosen from the LSH family $\mathcal{H}$. According to the definition above, $C(\boldsymbol{v})$ is composed of kL integers. However, for practical purposes, we use expression of $g_i(\boldsymbol{v}) = h_{i1}(\boldsymbol{v})M^{k-1} + h_{i2}(\boldsymbol{v})M^{k-2} + \cdots + h_{ik}(\boldsymbol{v})$ and regard $g_i(\boldsymbol{v})$ as a single integer, where M is the maximum value of h_{ij}. In this way, $C(\boldsymbol{v})$ can be described by L integers.

III. Proposed method

The target material of this study is "Hakodate Shinbun," a local newspaper published in 1878–1908 (Fig. 2). Out of the entire archives, the publications during 1878–1884 are used in this study. It consists of 3,462 pages. This section describes how we have processed this material.

Figure 2. Historical Newspaper Image: "Hakodate Shinbun" published in 1878–1884

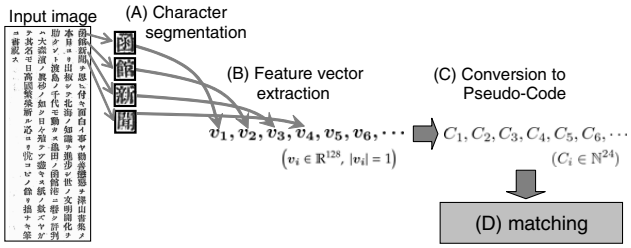

Figure 3. Outline of our method

Figure 3 shows the outline of the proposed method. Our method consists of several distinct steps: (A) Character segmentation; (B) Feature vector extraction; (C) Pseudo-code conversion; and (D) Matching of pseudo-code. Among these steps, (A) is a preprocess and is material-dependent. The main contribution of this paper lies in (C) and (D).

A. Character Segmentation

While character recognition of old newspaper is a highly difficult task, we may say that the difficulty of character segmentation of old newspaper is relatively moderate. Especially for Japanese document, character segmentation is easier than for Latin document because the size of character is mostly constant. The difficulties lie in those issues such that layout analysis, denoising, and so on.

Fortunately, since our material "Hakodate Shinbun" uses mostly fixed daily layout, we could use the knowledge of the fixed layout in the process. More specifically, we first applied Hough Transform to detect outer frames of a paper. Cutting on outer frames, most pages were separated into three block images. Each block image was further processed, including masthead detection, ruled lines removal, ruby characters removal, and denoising. After these preprocess, we applied a commercial OCR software [13] just for character segmentation. Note that this OCR software yielded terrible result for character recognition, but satisfactory result for character segmentation. When we manually evaluated the output using some sample pages, the accuracy of character recognition was 71%, while accuracy of character segmentation was 91%.

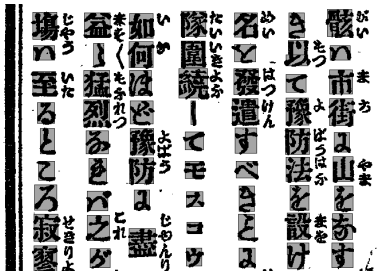

Figure 4. Result of character segmentation

We applied above process to the 3,462 pages of the material newspaper database. The result of this process yielded 8,594,920 segmented characters (Fig. 4).

B. Feature Vector Extraction

Each separated character was then fed to feature extraction process. As a feature vector, we used gradient distribution features [1] as the same as our previous study. The segmented image was divided into smaller 4×4 regions, which were used to form feature vectors of 128 dimensions, just similar to the descriptor utilized in SIFT [14].

C. Feature Vector to Pseudo-Code Conversion

In this process each feature vector was converted into LSPC. The feature vectors we have created in the precedent process have 128 dimensionality, and each component of the vectors was a positive real number, and the norm of each vector was normalized to unity. Such conditions allows us to apply Spherical LSH (SLSH) [12] in this LSPC conversion process.

To be more exactly, suppose the unit hypersphere of $\mathbb{R}^{128}$ space, and suppose the regular orthoplex (it is a generalization of the octahedron to higher dimensional space) inscribed in a unit hypersphere. When we randomly rotate (with random rotation matrix A) the orthoplex and partition the hypersphere so that all points on the hypersphere belong to the nearest vertex of the rotated orthoplex, since the orthoplex in $\mathbb{R}^{128}$ has 256 vertices, this process construct a map $\boldsymbol{v} \in \mathbb{R}^{128} \mapsto h_A(\boldsymbol{v}) \in \{0, 1, \cdots, 255\}$. However, since all component of $\boldsymbol{v}$ is restricted to positive value, we can consolidate the diagonal vertices without loss of generality. Thus, the map now could be expressed as $\boldsymbol{v} \in \mathbb{R}^{128} \mapsto h_A(\boldsymbol{v}) \in \{0, 1, \cdots, 127\}$.

The LSPC uses this map kL times and construct the pseudo-code $C(\boldsymbol{v}) = \{g_1(\boldsymbol{v}), g_2(\boldsymbol{v}), \cdots, g_L(\boldsymbol{v})\}$, where $g_i(\boldsymbol{v}) = \sum_{j=1..k}(h_{ij}(\boldsymbol{v}))^j$.

From the preparatory experiment, we determined the parameters $L = 24$ and $k = 3$. Since the range of $h_A(\boldsymbol{v})$ is 0 to 127, the range of $g_i(\boldsymbol{v})$ is 0 to $128^3 - 1 = 2097151$. Thus, the

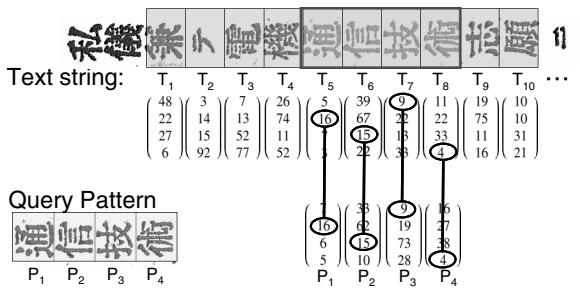

Figure 5. String matching of LSPC. Note that the pseudo-code is displayed as $\mathbb{N}^4$ in this figure, while the pseudo-code we actually adopted was $\mathbb{N}^{24}$.

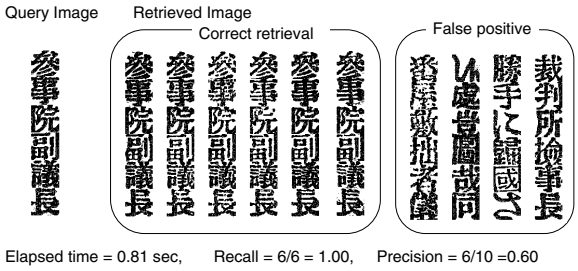

Figure 6. Search result of the query keyword of 6 character length

document images were converted to the sequence of LSPC. This conversion saves the space, that is, while original 128 dimensional vector requires 128 double variables, i.e., $128 \times 8 = 1024$ bytes per vector, the converted pseudo-code requires 24 long int variables, i.e., $24 \times 4 = 96$ bytes per code.

D. String Matching of Pseudo-Code

Now we already have sequence of LSPC. The remaining problem is a string matching of LSPC. The problem is formulated as follows (see also Fig. 5).

Definition 5: Given LSPC string P of length n and LSPC string T of length $m(> n)$. The string matching problem of LSPC is to search for all starting position i of a pattern P in a text T such that $P(\xi) \sim T(i+\xi-1)$ for all $\xi = 1, 2, ..., n$.

Here $P(\xi)$ represents the ξ-th code of P, and $S(i) \sim T(j)$ represents that code $S(i)$ and code $T(j)$ are semiequivalent.

To solve this problem, the most primitive method is a sequential scanning (also referred to as "Naive method" or "Brute-Force method"), which checks all position in the text T whether an occurrence of the pattern P starts there or not. This method can be easily implemented in the complexity of $O(nm)$ code-to-code comparison. In the experiment in the following section, we adopted this method.

IV. Searching Performance

This section describes the experimental result of our method. Experiments were performed on a computer with 64 bit operating system, Intel Xeon X5460 CPU (3.16 GHz), and 16GB memory.

For the evaluation, we chose five keywords from the entire text, all of which appear at least five times in the text. The length of the keywords varies from 5 characters to 10 characters. One of the keywords and its search result is displayed in Fig. 6.

We evaluated the method in terms of the processing time and the searching accuracy. For the measure of accuracy, we adopted recall score and precision score, which are defined as follows:

Table I
Searching Performance of Sequential Scan

Keyword Length	Time [sec]	Recall [%]	Precision [%]
5 characters	0.90	81.8	75.6
6 characters	0.81	85.7	25.5
7 characters	0.90	90.0	81.8
9 characters	0.91	82.4	82.1
10 characters	0.87	93.3	81.8

(full-text length = 8,594,920)

$$\text{recall} = \frac{\text{number of correctly retrieved strings}}{\text{number of correct occurrences}}$$

$$\text{precision} = \frac{\text{number of correctly retrieved strings}}{\text{number of retrieved strings}}$$

In precision evaluation, we manually checked all of retrieved strings whether it was correct or not. Recall evaluation is a bit difficult because we do not have entire ground-truth text and do not know the number of correct occurrences. Instead of complete survey, we have adopted sampling strategy, i.e., we picked out 5–18 occurrences as a test occurrence for each keywords, and calculated how many of the test occurrences was correctly retrieved.

The result is summarized in Table I. For all keywords, the processing time was less than a second. We presume the difference between 0.81–0.91 second came from the difference of the frequency of semiequivalent codes for each keyword. The recall scores were more than 80% for all keywords. The precision scores were also good except for one keyword (6 character-length keyword). The reason of the low score came from the fact that many confusingly similar codes existed for this keyword. In our future work, we are planning to improve the precision by adopting two step strategy, i.e., first pick out the candidates by LSPC alone, and narrow down the candidate using the original feature vectors.

Note that the accuracy of character segmentation was not perfect (91% to be exact), and this could cause the degradation of both recall and precision scores. The scores displayed in Table I was the scores endured over such degradation factors.

Table II
SEARCHING PERFORMANCE OF EX-BMH

Keyword Length	Sequential Scan [sec]	Ex-BMH [sec]
5 characters	0.90	1.23
6 characters	0.81	1.08
7 characters	0.90	1.01
9 characters	0.91	0.86
10 characters	0.87	0.81

(full-text length = 8,594,920)

V. FURTHER SOPHISTICATED ALGORITHM

By the virtue of discretization, we may consider some more sophisticated string matching algorithms for LSPC sequence, that cannot applied for feature vector sequence. As an example, this section introduce the Extended Boyer-Moore-Horspool (Ex-BMH) algorithm [15] and its performance in terms of processing time. In contrast to we theoretically verified the efficiency of Ex-BMH algorithm in [15], this paper is the first report that verify the efficiency of Ex-BMH algorithm by actual time measurement when applied to a large dataset of the document images.

The Ex-BMH algorithm is an extension of BMH algorithm [16] which is well known string matching algorithm for its practical efficiency. The Ex-BMH algorithm broadened the target of BMH algorithm from usual character string into LSPC string by modifying the skip function. In this paper, we only present the experimental result. The detailed algorithm description is found in [15].

Table II represents the measured processing time of Ex-BMH algorithm applied to the same dataset as in the previous section. By the nature of BMH algorithm, the processing time becomes shorter as the keyword length becomes longer. As seen in the table, Ex-BMH is really efficient if the keyword length is longer than 9 characters.

VI. CONCLUSION

This paper presents a fast appearance-based full-text search method for historical newspaper images. Experimental result showed that our method can retrieve a query string from a text of over eight million characters within a second. Since the LSPC technique is a kind of randomized algorithm, there is a trade-off between efficiency and accuracy. The proposed method increased the search speed in compensation for search accuracy, especially for precision scores. In our future work, we are planning to improve the precision by adopting two step strategy, i.e., first pick out the candidates by LSPC alone, and narrow down the candidate using the original feature vectors. Of course, in order to obtain the highest accuracy in compensation for high computational cost, to avoid using LSPC is a possible choice. The method should be chosen according to the practical applications.

Our future work also includes developing an efficient algorithm to realize Inexact Matching rather than Exact Matching. With such an advanced algorithm, it would be possible to develop a fast algorithm using LSPC, which is applicable to more difficult problems such as string matching of handwritten documents.

ACKNOWLEDGMENT

This work was supported in part by PRESTO of Japan Science and Technology Agency (JST). The experimental materials were provided by Hakodate City Central Library.

REFERENCES

[1] K. Terasawa and Y. Tanaka, "Locality Sensitive Pseudo-Code for Document Images," Proc. ICDAR2007, vol. 1, pp. 73–77, 2007.

[2] J. He and A. Downton, "Evaluation of a User-Assisted Archive Construction System for Online Natural History Archives," Proc. ICDAR2005, pp. 442–446, 2005.

[3] F. Drira, F. LeBourgeois, H. Emptoz, "Document Images Restoration by a New Tensor Based Diffusion Process: Application to the Recognition of Old Printed Documents," Proc. ICDAR2009, pp. 321–325, 2009.

[4] V. Kluzner, A. Tzadok, Y. Shimony, E. Walach, A. Antonacopoulos, "Word-Based Adaptive OCR for Historical Books," Proc. ICDAR2009, pp. 501–505, 2009.

[5] Y. Tanaka and H. Torii, "Transmedia Machine and its keyword search over image texts," Proc. RIAO'88, pp. 248–258, 1988.

[6] H. Imura and Y. Tanaka, "An Indexed Full-Text Search Method of Printed Document Images with an M-tree," Proc. RIAO'10, pp. 72–75, 2010.

[7] C. L. Tan, W. Huang, Z. Yu, Y. Xu, "Imaged Document Text Retrieval without OCR," IEEE Trans. on PAMI, vol. 24, no. 6, pp. 838–844, 2002.

[8] S. Marinai, E. Marino, G. Soda, "Indexing and Retrieval of Words in Old Documents," Proc. ICDAR2003, vol. 1, pp. 223–227, 2003.

[9] A. Gionis, P. Indyk, R. Motwani, "Similarity Search in High Dimensions via Hashing," Proc. 25th Int. Conf. on Very Large Data Base, VLDB1999, pp. 518–529, 1999.

[10] M. Datar, P. Indyk, N. Immorlica, V. Mirrokni, "Locality-Sensitive Hashing Scheme Based on p-Stable Distributions," Proc. Symposium on Computational Geometry 2004, pp. 253–262, 2004.

[11] A. Andoni and P. Indyk, "Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions," Proc. Symposium on Foundations of Computer Science, FOCS'06, pp. 459–468, 2006.

[12] K. Terasawa and Y. Tanaka, "Sperical LSH for Approximate Nearest Neighbor Search on Unit Hypersphere," Proc. 10th Workshop on Algorithms and Data Structures, WADS2007, LNCS 4619, pp. 27–38, 2007.

[13] MDTOCR v.7.0, Media Drive Corporation, Japan

[14] D.G. Lowe, "Distinctive image features from scale-invariant keypoints," Int. J. Computer Vision, vol. 60, no. 2, pp. 91–110, 2004.

[15] K. Terasawa , T. Kawashima, Y. Tanaka, "The Extended Boyer-Moore-Horspool Algorithm for Locality-Sensitive Pseudo-Code," Proc. 6th Int. Conf. on Computer Vision Theory and Applications, VISAPP 2011, pp. 437–441, 2011.

[16] R. N. Horspool, "Practical fast searching in strings", Software – Practice & Experience, vol. 10, issue 6, pp. 501–506, 1980.