ELSEVIER

Pattern Recognition Letters 22 (2001) 545–550

Pattern Recognition Letters

www.elsevier.nl/locate/patrec

Detecting duplicates among symbolically compressed images in a large document database

Dar-Shyang Lee, Jonathan J. Hull *

Ricoh California Research Center, Document Analysis Group, 2882 Sand Hill Road, Suite 115, Menlo Park, CA 94025-7054, USA

Abstract

The detection of duplicate images is a useful means of indexing a large database of documents. An algorithm for duplicate document detection is proposed in this paper that operates directly on images that have been symbolically compressed using techniques related to the ongoing JBIG2 standardization effort. This paper describes a hidden Markov model (HMM) method that recognizes the text in an image by deciphering data from the compressed representation. Experimental results show that it can recover better than 90% of the text in compressed document images and that this is sufficient to identify duplicates in a large database. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Duplicate detection; Document indexing; Symbolic compression; IM^3

1. Introduction

Duplicate documents can be a significant problem in large collections. Ideally, a duplicate detection algorithm can find both exact duplicates, which have exactly the same content, and partial duplicates which have a large percentage of their text in common. Locating exact duplicates could reduce the storage required for a large database. Finding partial duplicates would allow users to easily find other versions of a given document.

Since document images are usually stored and transmitted in compressed formats, considerable run-time advantages are realized by performing the matching process directly on compressed images (Lee and Hull, 1999a,b). One technical challenge is the extraction of meaningful information from compressed data. *Symbolic* compression schemes preserve much of the structure in a document image thereby facilitating feature extraction. They cluster individual blobs and store the sequence of occurrence of clusters and representative blob templates. This kind of compression scheme was originally proposed for binary images of text (Ascher and Nagy, 1974). Numerous algorithms based on this kind of technique, such as JBIG2 (e.g., Howard et al., 1998) have been proposed recently.

Duplicate detection is particularly important for large document databases like that produced by the Infinite Memory Multifunction Machine (IM^3). The IM^3 system (Hull and Hart, 1998; Hull et al., 1999) captures a copy of every printed, copied, or faxed document generated in an office. This guarantees that almost any document a user might need will be available when they need it. This concept was developed after it was observed

[*] Corresponding author.
E-mail addresses: dsl@crc.ricoh.com (D.-S. Lee), hull@crc.ricoh.com (J.J. Hull).

0167-8655/01/$ - see front matter © 2001 Elsevier Science B.V. All rights reserved.
PII: S0167-8655(00)00115-X

that even though the obvious method for document capture, namely scanners, were commonly available, they were not commonly used.

The IM^3 makes document capture effortless by saving an electronic copy of *every* document that users copy, print, or fax. Furthermore, users are not asked whether any particular document should be captured – no conscious decision is required at capture time. Thus, every person in an office that copies, prints, or faxes a document automatically contributes data to the IM^3.

The design for the IM^3 prototype system that was implemented and tested at the authors' laboratory is shown in Fig. 1. It is based on a typical office environment in which PC's, Mac's, Unix workstations, digital copiers and printers are interconnected on a local area network. When users print a document, it is first sent to the print server. In addition to sending it to the appropriate printer, an electronic copy is transferred to the IM^3 server. OCR is automatically performed and the document is indexed for later retrieval. Copiers and fax machines work similarly. After over three years of use, more than 70,000 documents with greater than 300,000 pages have been saved. A substantial number of these documents are undoubtedly duplicates of one another or were created in an edit-print cycle that is commonly used when creating new documents.

Documents stored in the IM^3 are accessed with a web browser. Each user has a home page that provides a portal to their document collection. A number of techniques are provided for search and retrieval. These include full text search and various methods for browsing based on the dates when documents were captured. A method for duplicate detection would give users a means for retrieving exact copies and other versions of a given document. Version retrieval would be particularly useful when after retrieving a document by full text search with a certain set of keywords, the user would like to retrieve other versions of that document. Even though substantial amounts of text may be common between versions, they might not all share the keywords that were used for full text search. A more robust technique is called for.

2. Duplicate detection

We assume that all document images are compressed with a "symbolic" technique (e.g., JBIG2 (Howard et al., 1998)). Features are extracted directly from the compressed version of document images. A comparison procedure determines whether the feature descriptions of any two documents are similar enough for the original documents to be duplicates.

Symbolic compression for binary document images first clusters connected components, which often correspond to isolated characters. A unique identifier is then assigned to each cluster. The compressed document contains one image for each cluster and the sequence of identifiers for the connected components (also called blobs) in the original image. This sequence of identifiers corresponds to the sequence of occurrence of characters in the original document.

An idealized example of a symbolically compressed (similar to JBIG2) document image is

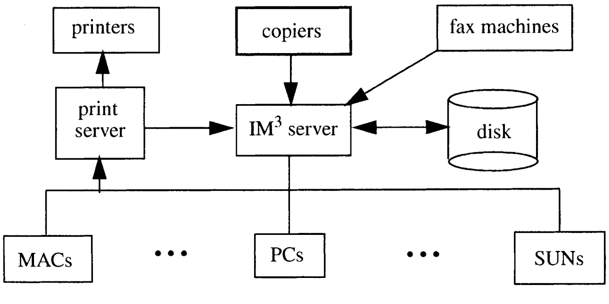

Fig. 1. IM^3 system design.

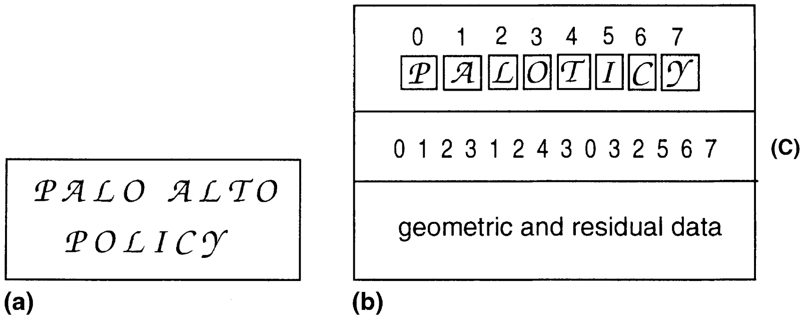

Fig. 2. Depiction of symbolic compression, showing an original image (a) and the compressed image (b). The sequence of indentifiers in (c) encodes the order of characters in the original document.

shown in Fig. 2. An original document image is shown (a) as well as its compressed form (b). The unique letters in the original image are represented as individual sub-images and numeric identifiers at the top of (b). The sequence of identifiers shown in Fig. 2(c) encodes the order in which the corresponding sub-images occurred in the original image (a). For example, “0 1 2 3” are the first four sub-images in this sequence. They correspond to the first four letters in the image, “$\mathscr{P}$ $\mathscr{A}$ $\mathscr{L}$ $\mathscr{O}$”. The *x–y* locations of the sub-images and image residual data are also encoded in the compressed format.

The characteristic of symbolic compression that we use for duplicate detection is the sequence of cluster identifiers (“0 1 2 3 1 2 4 3 0 3 2 5 6 7” in Fig. 2(c)). This sequence encodes a representation for the text in the original document. Since each cluster, for the most part, corresponds to a single character, we can treat the sequence of cluster identifiers as a *substitution cipher*.

A substitution cipher replaces one character with another to produce an enciphered message. The original plain text can be recovered by a deciphering algorithm that computes the pairwise correspondence between symbols in the enciphered message and plain text characters. For the example shown in Fig. 2, this correspondence is $\{(0,\mathscr{P}),(1,\mathscr{A}),(2,\mathscr{L}),(3,\mathscr{O}),(4,\mathscr{T}),(5,\mathscr{I}),(6,\mathscr{C}),(7,\mathscr{Y})\}$.

We apply a deciphering algorithm to the sequence of cluster identifiers. It computes a pairwise correspondence between cluster identifiers and letters. This correspondence is used to recover the text in the original document image. This is essentially OCR'ing the document without actually applying any OCR techniques. A similar idea was first proposed in (Casey and Nagy, 1968). Our method takes advantage of the image preprocessing done by the symbolic compression technique. Also, we developed a new algorithm for substitution cipher decoding that takes into account characteristics of symbolic clustering in document images (Lee and Hull, 1999a,b). Other techniques for substitution cipher decoding are described elsewhere (King and Bahler, 1992; Peleg and Rosenfeld, 1979).

The deciphering algorithm reads the sequence of cluster identifiers from a symbolically compressed document image and uses character transition probabilities and a hidden Markov model (HMM) to estimate the text that appeared in the original document. There might not be a decision for every character and all the decisions might not be correct. However, enough of the text is usually correctly recovered that accurate duplicate detection can be performed.

The text strings extracted from two documents are compared using the *conditional n*-grams they have in common. A conditional *n*-gram is a sequence of *n* characters where each character satisfies a predicate. This predicate converts the original document into a new string from which *n*-gram indexing terms are extracted. For example, we used a predicate that every character must follow a space. Therefore, the new string generated by this predicate contains the first character of every word. Conditional trigrams are formed from the first characters of three consecutive words. More details on the selection criteria for the predicate and the performance of conditional n-grams can be found in (Lee and Hull, 1999a,b). The similarity between two documents is measured as a weighted dot product of the trigram frequencies. If this score exceeds a threshold, the documents are considered to be duplicates of one another.

3. Document deciphering

The HMM deciphering algorithm described here is more robust to the commonly occurring problem of there being more than one cluster for a given character identity in a typical document image than other substitution deciphering solutions. The deciphering approach is particularly suitable for processing symbolically compressed documents because pattern clustering and sorting are part of the compression process. The sequence of cluster identifiers, accounting for only 20% of the total bits required for lossy compression, can be easily accessed and transmitted without decoding the image.

Markov models have been used for natural language modeling. If we accept the Markov process of state traversal as a language source from which a particular plain text message can be generated with some probability, then the added symbol production at the traversed states in a HMM perfectly describes the enciphering procedure of a monographic substitution cipher, where each letter in plain text is replaced with a cipher symbol one at a time. This analogy between source language modeling as a Markov process and representation of the enciphering function by symbol probabilities is the basis for our solution, as shown in Fig. 3.

In a first order model, there are n states, each representing a letter in the plain text alphabet. Associated with each state, α, is a state transition probability function, A_α, and a symbol probability function, P_α. The first state in a sequence is selected according to an initial probability, I_α. Subsequent states are generated according to the transition probabilities, outputting one of the cipher symbols $\{c_1, c_2, \ldots, c_m\}$ at each state with probability $P_\alpha(c_i)$. The transition probability from state α to state β can be calculated from the bigram frequencies that character α is followed by character β. The initial state probability I_α is simply the character frequency of α. Both the initial and transition probabilities are estimated from a corpus of the source language and remain fixed, providing a first order Markov modeling of the source language. Symbol probabilities P_α are estimated using the forward-backward algorithm (Rabiner and Juang, 1986). The initial estimation $P_\alpha^{(0)}(c_i)$ is defined as

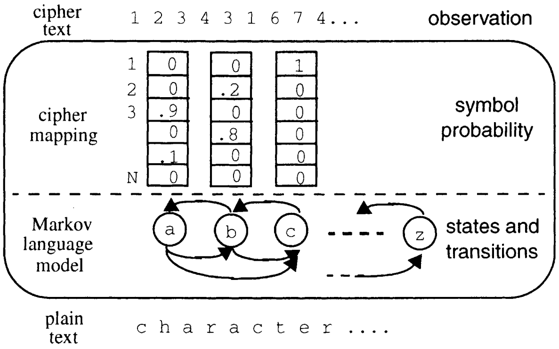

Fig. 3. In the hidden Markov approach, the deciphering problem is formulated as finding the enciphering mapping that most likely produced the observed cipher text for the underlying Markov language source.

$$P_\alpha^{(0)}(c_i) = \frac{B_i(\alpha)\mathrm{Prob}(c_i)}{\sum_{j=1}^{m} B_j(\alpha)\mathrm{Prob}(c_j)},$$

where $B_j(\alpha)$ is calculated from a cipher of length L with k_i occurrences of symbol c_i using a binomial distribution

$$B_i(\alpha) = \frac{\mathrm{Prob}(\alpha)^{k_i}[1 - \mathrm{Prob}(\alpha)]^{L-k_i}}{\sum_\beta \mathrm{Prob}(\beta)^{k_i}[1 - \mathrm{Prob}(\beta)]^{L-k_i}}.$$

To determine the decipher mapping, $\mathscr{D}_{\mathrm{HMM}}^{(t)}(c_i)$, we assign a plain symbol that most likely corresponds to each cipher symbol. Since P_α is conditioned on the cipher symbol, the following decision criterion is used

$$\mathscr{D}_{\mathrm{HMM}}^{(t)}(c_i) = \underset{\alpha}{\mathrm{argmax}} P_\alpha^{(t)}(c_i)\mathrm{Prob}(\alpha).$$

4. Experimental results

Several experiments were conducted. The HMM deciphering solution was first tested on simulated simple substitution ciphers to establish a baseline performance in the ideal case. The algorithm was then applied to a small number of symbolically compressed documents to measure its performance on real document images. This measured performance was then used in a simulated test to demonstrate the feasibility of detecting duplicates by deciphering images.

For the simulated simple substitution cipher experiments, the University of Calgary corpus was used as a language source. Our plain text alphabet is composed of 26 lower case letters and the space character. The identity matrix is used for enciphering: each lower case letter is mapped to its corresponding upper case letter, and the space character is mapped to itself. After removing typesetting commands, deleting punctuations and performing necessary preprocessing, test sets of varying length passages were generated. Bigram and trigram statistics estimated from a separate training file are used to initialize the HMMs. We ran each experiment for a maximum of 10 iterations or until the changes in the solution matrix become smaller than a threshold. The decode rates for the various trials are summarized in Table 1.

The results show that for ciphers of length greater than 1600 characters, both the bigram and trigram models can fully recover the original text. A trigram model can successfully decipher the majority of a cipher text as short as 400 characters, at a cost of increased running time. In most cases, a bigram model provides a good balance between efficiency and performance.

The HMM deciphering algorithm was then applied to blob sequences extracted from real images compressed with *mgtic* in the MG library (Witten et al., 1994). Character interpretation rates varied between 80% and 95%, depending on the content, typesetting and image quality.

Table 1
Summary of final decoding rates for HMM bigram and trigram models on simple ciphers

Length (char)	100	400	800	1600
Bigram %decode	57.55	93.19	96.74	99.13
Trigram %decode	66.47	98.80	99.01	99.54

The performance of the conditional *n*-gram method for text string comparison was tested on the 979 documents in the University of Washington (UW) database (Phillips et al., 1993). It contains 146 pairs of duplicate documents. Each member of a pair had been scanned from a different generation photocopy of the same document. Test data was constructed by adding noise to the ASCII truth files to simulate a 90% correct decode rate (about the decode rate achieved on real images).

Each document was compared to the other 978 documents by calculating a similarity score using a weighted sum of the frequencies of the conditional *n*-grams they have in common. A sorted list of the 10 documents with the highest similarity scores was output. The most similar document is at the top of the list. Ideally, this is a duplicate of the original document, if it exists in the database.

The results showed that conditional trigrams provide a 100% correct rate in duplicate detection. This compared favorably to the 81.85% correct rate achieved by non-conditional trigrams, in the first choice, and 97.95% in the top 10 choices.

5. Conclusions

A method was presented for performing document duplicate detection directly on images in a symbolic compression format. Since the language statistics inherent in document content are largely preserved in the sequence of cluster identifiers, the original character interpretations can be recovered with a deciphering algorithm. We proposed an HMM solution for the deciphering problem. While the overall character interpretation rates are not perfect, we demonstrated that sufficient information is recovered for document duplicate detection. This offers an efficient and versatile solution to detecting full and partial duplicates. It also provides a useful method for indexing large document databases. Future work will consider implementation of this technique in the IM^3 system.

References

Ascher, R.N., Nagy, G., 1974. A means for achieving a high degree of compaction on scan-digitized printed text. IEEE Trans. Comput. C-23 (11), 1174–1179.

Casey, R., Nagy, G., 1968. Autonomous reading machine. IEEE Trans. Comput. C-7.

Howard, P., Kossentini, F., Martins, B., Forchhammer, S., Rucklidge, W.J., 1998. The emerging JBIG2 standard. IEEE Trans. Circuits Systems Video Technol. 8 (7), 838–848.

Hull, J.J., Hart, P., 1998. The infinite memory multifunction machine. In: Pre-proceedings 3rd IAPR Workshop on Document Analysis Systems, Nagano, Japan, 4–6 November, pp. 49–58.

Hull, J.J., Lee, D.-S., Cullen, J., Hart, P., 1999. Document analysis techniques for the infinite memory multifunction machine. In: Proc. 10th Internat. Workshop on Database and Expert System Applications, Florence, Italy, 1–3 September, pp. 561–565.

King, J., Bahler, D., 1992. An implementation of probabilistic relaxation in the cryptanalysis of simple substitution ciphers. Cryptologia 16 (3), 215–225.

Lee, D.-S., Hull, J.J., 1999. Information extraction from symbolically compressed document images. In: Proc. 1999 Symposium on Document Image Understanding Technology, Annapolis, MD, 14–16 April, pp. 176–182.

Lee, D.-S., Hull, J.J., 1999. Duplicate detection for symbolically compressed documents. In: Proc. 5th Internat. Conf. Document Analysis and Recognition, Bangalore, India, 20–22 September, pp. 305–308.

Peleg, S., Rosenfeld, A., 1979. Breaking substitution ciphers using a relaxation algorithm. Commun. ACM 22 (11), 598–605.

Phillips, I.T., Chen, S., Haralick, R.M., 1993. CD-ROM document database standard. In: Proc. 2nd ICDAR, pp. 478–483.

Rabiner, L.R., Juang, B.H., 1986. An introduction to hidden Markov models. IEEE ASSP Magazine, 4–16.

Witten, I., Moffat, A., Bell, T., 1994. Managing Gigabytes: Compressing and Indexing Documents and Images. Van Nostrand Reinhold, New York.