Duplicate Detection in Symbolically Compressed Documents

Dar-Shyang Lee and Jonathan J. Hull
Ricoh Silicon Valley, Inc.
2882 Sand Hill Road, Suite 115
Menlo Park, CA 94025, U.S.A.
email: {dsl,hull}@rsv.ricoh.com

Abstract

A new family of symbolic compression algorithms, such as the ongoing JBIG2 standardization and commercial products, has recently been developed. These techniques are specifically targeted for binary document images. They cluster individual blobs in a document and store the sequence of occurrence of blobs and representative blob templates, hence the name symbolic compression. This paper describes a method for duplicate detection on symbolically compressed document images. It recognizes the text in an image by deciphering the sequence of occurrence of blobs in the compressed representation. We propose a Hidden Markov Model (HMM) method for solving such deciphering problems and suggest applications in multilingual document duplicate detection.

1. Introduction

Document matching is an important component of a document image storage system, allowing for removal of duplicates, copyright violation detection and other applications. Since document images are usually stored and transmitted in compressed formats, considerable advantages are realized by performing the matching process directly on compressed images [5][10]. One technical challenge is the extraction of meaningful information from relatively unstructured compressed data.

The dramatic increase in the use of document images in recent years has led to research in compression technology for textual images. *Symbolic* compression schemes preserve much of the structure in a document image, facilitating feature extraction. They cluster individual blobs in a document and store the sequence of occurrence of blobs and representative blob templates. This kind of compression scheme for digitized textual data has been proposed before [1]. Lossless compression algorithms can be designed around this idea by supplementing a coding scheme for the residuals that result from pattern matching [11]. It has been shown that such separate coding of patterns and residuals achieves better compression ratios than conventional methods. Numerous algorithms based on the pattern matching approach such as JBIG2 [4] and others [8][17] have since been proposed.

In this paper, a method of exploiting the structure contained in the sequence of identifiers, using the deciphering approach, to partially recover character interpretation for document matching is presented. The sequence of cluster identifiers is extracted from the compressed file and deciphered using Hidden Markov Models (HMM). A modified n-gram method then takes the deciphered interpretation and retrieves matching documents from the database. The rest of the paper is organized as follows. Section 2 briefly introduces the symbolic compression scheme and the connection to ciphers. Section 3 presents an HMM based deciphering process for recovering character interpretations directly from the symbol sequence. Experimental results for HMM deciphering and duplicate detection are presented in Section 4, followed by conclusions in Section 5.

2. Document Deciphering

In symbolic compression, images are first coded with respect to a library of templates, which can either be provided externally or constructed from representative patterns generated from clusters of similarly shaped blobs. Blobs, or connected components, in the original image are grouped together based on their shape similarities. A respective template pattern and a unique identifier is generated for each group. Subsequently, blobs in the image are stored as a sequence of cluster identifiers and their location offsets from the previous component. For example, if clusters '`c`', '`h`', '`a`', '`r`', '`t`', '`e`' are assigned identifiers 1 through 6, respectively, then components in the word image ‘character’ map onto the integer sequence ‘1 2 3 4 3 1 5 6 4’. In this way, a good approximation of the original document can be afforded without duplicating storage for similar patterns. Minor differences between individual components and their representative templates, as well as all other components which are not encoded in this manner, are optionally coded as residuals. Without duplicating storage for similar patterns, a symbolic method improves the compression ratio by 50% to 100% over the commonly used Group 4 standard. If small differences in the residuals can be tolerated, a lossy version can achieve a 4 to 10 times better compression ratio than Group 4 [18].

The connection between symbolically compressed document images and substitution ciphers is in the correspondence between cluster identifiers and characters. If all the blobs corresponding to each character are perfectly clustered, there is a one-to-one mapping between cluster identifiers and character interpretations. This is a simple substitution cipher that can be easily solved to recover the text on the document.

However, such idealized case usually does not apply to the compressed document images encountered in practice. Image segmentation errors might split single characters into several blobs. Also, multiple clusters might be generated for a single character. This can produce a non-monographic many-to-one substitution cipher for which there is no known general solution.

We avoid the complex cipher problem by assuming that the majority of blobs correspond to a single character. The existence of more than one cluster in the compressed file for a given character identity (a many-to-one substitution cipher) is handled by the HMM deciphering algorithm proposed here. It will be shown empirically that the text recovered from a compressed file by the HMM, while often containing errors, is still sufficient for duplicate detection.

Compared to conventional OCR, machine reading of document images by deciphering has received little attention [2][12]. However, 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 template identifiers, accounting for only 20% of the total bits required for lossy compression, can be easily accessed and transmitted without decoding the image. Furthermore, the deciphering approach is quick to adapt to new languages, requiring only language statistics or an electronic corpus. The combination of typeface independence, language adaptability and accessibility from compressed documents offers great opportunities for the deciphering techniques in multilingual document database applications.

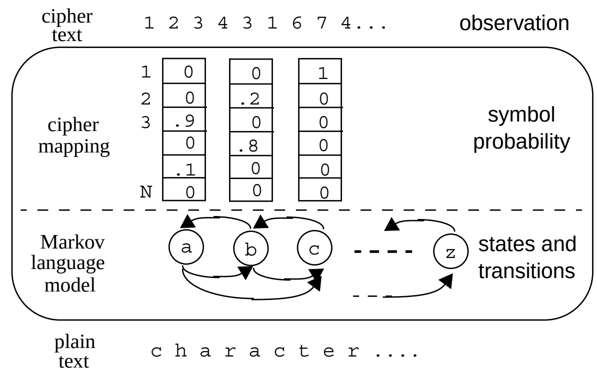

Figure 1 - In the hidden Markov approach, the deciphering problem is formulated as finding the enciphering mapping that most likely produced the observed cipher text for the underlying Markov language source.

3. Deciphering by Hidden Markov Models

Numerous solutions for the deciphering problem have been proposed, including relaxation algorithms [13][7], dictionary based pattern matching [12], and optimization techniques [3][16]. We propose a solution using the well developed theory of Hidden Markov Models [15]. The abstraction of state transitions and observable symbols in an HMM is analogous to the separation of a Markov language source and subsequent enciphering step.

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 hidden Markov model 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 Figure 1.

In a first order model, there are *n* states in the model, 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 [15]. The initial estimation $P_\alpha^{(0)}(c_i)$ is defined as

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

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

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

To determine the decipher mapping, 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.

$$D_{HMM}^{(t)}(c_i) = \underset{\alpha}{\operatorname{argmax}}\ P_\alpha^{(t)}(c_i)Prob(\alpha)$$

It should pointed out that we have explicitly constructed the deciphering function from the estimated symbol probabilities. However, it is unnecessary, even circuitous, to produce the underlying plain text this way. With fixed transition probabilities and estimated symbol probabilities, the Viterbi algorithm can be used to find the most likely sequence of states through which the observed symbols are produced. This state sequence, corresponding to the most likely plain text from which the cipher text is generated, given our parameter estimations, can be used directly for evaluation. Contrary to the deciphering solution where all occurrences of a cipher symbol in the cipher text must decode into the same letter in plain text, the most probable plain text generated from a state sequence may not have a consistent one-to-one mapping to the cipher text. Constructing the deciphering function incorporates the likelihood of all possible paths and has shown better results in our experiments than the direct method.

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. Each of these experiments will be described in this section.

For the simulated simple substitution cipher experiments, the University of Calgary corpus is 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 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 I.

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

Table I - Summary of final decoding rates for HMM bigram and trigram models on simple ciphers.

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. These numbers compared favorably against other non-lexicon based deciphering algorithms such as a relaxation method.

The HMM deciphering algorithm was then applied to blob sequences extracted from real images compressed with *mgtic* in the MG library [18]. Character interpretation rates varied between 80% to 95%, depending on the content, typesetting and image quality of the document. Based on these observations, duplicate detection experiments were simulated at 85% and 90% decode rates using the University of Washington database. The data set contained 979 documents and included 146 pairs of duplicates. Noise was injected independently for the 146 pairs of duplicates to produce different OCR results. Although conventional keyword and n-gram based methods performed poorly at these noise levels, a modified n-gram indexing with term weighting achieved better than 97% recall with high precision [9], as shown in Table II.

Finally, we implemented a multilingual duplicate detection system. A database consisting of 150 docu-

Decode rate	90% decode	85% decode
Top 1 recall	100.0%	97.3%
Top 10 recall	100.0%	99.7%
SNR (dB)	28.9	22.1

Table II - Duplicate detection performance at 85% and 90% document deciphering rate.

ments in English, German and Russian was constructed. A scanned document was first symbolically compressed, then simultaneously deciphered for all three languages. The likelihood calculated by each HMM was used to determine the language and the appropriate indexing table for subsequent duplicate detection. The system correctly identified the language and document in 30 test images. It also consistently recognized second and third generation copies.

5. Conclusions

A method was presented for performing document duplicate detection directly on images in the 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. Although various imaging issues differentiate the problem from the ideal one-to-one mapping, the HMM solution is robust in recovering the correct mapping from surrounding noise. While the overall character interpretation rates are far from perfect, we demonstrated that sufficient information is recovered for document duplicate detection. This offers an efficient and versatile solution to applications in multilingual document image database systems.

6. References

[1] R. N. Ascher and G. Nagy, "A means for achieving a high degree of compaction on scan-digitized printed text," IEEE Transactions on Computers, Vol. C-23, No. 11, pp. 1174-1179, Nov. 1974.

[2] R. Casey and G. Nagy, "Autonomous reading machine," IEEE Transactions on Computers, vol. C-7, May 1968.

[3] W. S. Forsyth and R. Safavi-Naini, "Automated cryptanalysis of substitution ciphers," Cryptologia, vol. 17, no. 4, pp. 407-418, 1993.

[4] P. Howard, F. Kossentini, B. Martins, S. Forchhammer, and W. J. Rucklidge, "The Emerging JBIG2 Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 7, pp. 838-848, November 1998.

[5] J. J. Hull, "Document matching on CCITT Group 4 compressed images," SPIE Conference on Document Recognition IV, pp. 82-87, 1997.

[6] J.J. Hull and P. Hart, "The Infinite Memory Multifunction Machine," Proceedings the Third IAPR Workshop on Document Analysis Systems, pp. 49-58, Nagano, Japan, Nov. 4-6, 1998.

[7] D. G. N. Hunter and A. R. McKenzie, "Experiments with relaxation algorithms for breaking simple substitution ciphers," The Computer Journal, vol. 26, no. 1, pp. 68-71, 1983.

[8] O. E. Kia, "Document image compression and analysis," Ph.D. dissertation, Depart of Electrical Engineering, University of Maryland, 1997.

[9] D-S. Lee, Jonathan J. Hull, "Information extraction from symbolically compressed document images," Proceedings of SDIUT, pp. 176-182, Annapolis, April 14-16, 1999.

[10] D-S. Lee, "Group 4 compressed document matching," Proceedings of Third IAPR Workshop on Document Analysis Systems, pp. 29-38, 1998.

[11] K. Mohiuddin, J. Rissanen and R. Arps, "Lossless binary image compression based on pattern matching," Proceedings of International Conference on Computers, Systems & Signal Processing, December, 1984.

[12] G. Nagy, S. Seth and K. Einspahr, "Decoding substitution ciphers by means of word matching with application to OCR," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-9, no. 5, pp. 710-715, 1987.

[13] S. Peleg and A. Rosenfeld, "Breaking substitution ciphers using a relaxation algorithm," Communications of the ACM, vol.22, no.11, pp. 598-605, November 1979.

[14] I. T. Phillips, S. Chen, R. M. Haralick, "CD-ROM document database standard," Proceedings of the 2nd ICDAR, pp. 478-483, 1993.

[15] L. R. Rabiner and B. H. Juang, "An introduction to hidden markov models," IEEE ASSP Magazine, pp. 4-16, January 1986.

[16] R. Spillman, M. Janssen, B. Nelson and M. Kepner, "Use of a genetic algorithm in the cryptanalysis of simple substitution ciphers," Cryptologia, vol. 17, no. 1, pp. 31-44, 1993.

[17] I. Witten, T. Bell, H. Emberson, S. Inglis, and A. Moffat, "Textual Image Compression: two stage lossy/lossless encoding of textual images," *Proceedings of the IEEE*, vol. 82, no. 6, pp. 878-888, June 1994.

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