Binary Image Compression Using Conditional Entropy-Based Dictionary Design and Indexing*

Yandong Guo[a], Dejan Depalov[b], Peter Bauer[b], Brent Bradburn[b], Jan P. Allebach[a], and Charles A. Bouman[a]

[a]School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907-2035, U.S.A.;
[b]Hewlett-Packard Co., Boise, ID 83714, U.S.A.

ABSTRACT

The JBIG2 standard is widely used for binary document image compression primarily because it achieves much higher compression ratios than conventional facsimile encoding standards, such as T.4, T.6, and T.82 (JBIG1). A typical JBIG2 encoder works by first separating the document into connected components, or symbols. Next it creates a dictionary by encoding a subset of symbols from the image, and finally it encodes all the remaining symbols using the dictionary entries as a reference.

In this paper, we propose a novel method for measuring the distance between symbols based on a conditional-entropy estimation (CEE) distance measure. The CEE distance measure is used to both index entries of the dictionary and construct the dictionary. The advantage of the CEE distance measure, as compared to conventional measures of symbol similarity, is that the CEE provides a much more accurate estimate of the number of bits required to encode a symbol. In experiments on a variety of documents, we demonstrate that the incorporation of the CEE distance measure results in approximately a 14% reduction in the overall bitrate of the JBIG2 encoded bitstream as compared to the best conventional dissimilarity measures.

Keywords: JBIG2, binary document image compression, conditional entropy estimation, optimal dictionary design, sparse image representation

1. INTRODUCTION

Binary image compression is an important problem, which is quite different than conventional continuous tone image compression implemented by widely used methods such as JPEG or JPEG2000. Perhaps the best existing standard for binary image compression is the JBIG2 compression standard developed by the Joint Bi-level Image Experts Group.[1] The JBIG2 standard is important because it can achieve much higher compression ratios than previous binary image compression methods, such as T.4, T.6, and T.82 (JBIG1),[2] through the use of symbol dictionaries. In addition, while the other binary image compression standards support only lossless encoding, JBIG2 supports both lossless and lossy modes.

A typical JBIG2 encoding is achieved by a combination of the following operations: image segmentation, symbol extraction, dictionary construction, and entropy encoding.[3,4] First, the original binary document image is separated into regions corresponding to text (i.e., repeated symbols) and non-text. In this work, we will assume the entire JBIG2 document is treated as text. Next, individual connected components are extracted that typically correspond to individual text symbols. Then, the extracted symbols are used to create a representative dictionary of typical symbols, and the dictionary is encoded using a traditional entropy encoding method. Finally,

*This work was supported by the Hewlett-Packard Company.

Further author information: (Send correspondence to Yandong Guo)
Yandong Guo: E-mail: guoy@purdue.edu, Telephone: +1 765 418 3801
Dejan Depalov: E-mail: dejan.depalov@hp.com, Telephone: +1 208 396 7902
Peter Bauer: E-mail: peter.bauer@hp.com, Telephone: +1 208 396 6981
Brent Bradburn: E-mail: brent.michael.bradburn@hp.com, Telephone: +1 208 396 4855
Jan P. Allebach: E-mail: allebach@purdue.edu, Telephone: +1 765 494 3535
Charles A. Bouman E-mail: bouman@purdue.edu, Telephone: +1 765 494 0340

Color Imaging XVIII: Displaying, Processing, Hardcopy, and Applications,
edited by Reiner Eschbach, Gabriel G. Marcu, Alessandro Rizzi, Proc. of SPIE Vol. 8652
865208 · © 2013 SPIE-IS&T · CCC code: 0277-786X/13/$18 · doi: 10.1117/12.2006141

the symbols for each page of the document are encoded by indexing them to individual entries of the dictionary, and these indexes are entropy encoded. With lossly JBIG2 encoding, each symbol is represented approximately with a corresponding dictionary entry; but with lossless JBIG2 encoding, the difference between the symbol and corresponding dictionary entry is also entropy encoded.

While all JBIG2 encoders perform these basic tasks, some encoders achieve better compression by constructing better dictionaries and using those dictionary entries more effectively. For example, a larger dictionary can reduce the number of bits required for lossless encoding of each symbol because each dictionary entry can more precisely match its corresponding symbol. However, a larger dictionary will also increase the number of bits required to encode the dictionary itself, so the best coder must balance these two objectives.

Even if the dictionary is fixed, an encoder can improve compression by more optimally selecting the dictionary entry for each new symbol. Ideally, each symbol should be encoded using the dictionary entry that produces the lowest number of bits in the encoded output. However, this approach, which is known as the operational rate-distortion approach,[5,6] is not practical since it requires too much computation. Therefore, any practical JBIG2 coders must use a more computationally efficient method to match each symbol to a dictionary entry.

Conventional methods select the dictionary entry by minimizing a measure of dissimilarity between the symbol and the dictionary entry. Dissimilarity measures widely used in JBIG2 include the Hamming distance, known as XOR,[7] and weighted Hamming distance, known as WXOR.[8,9] The weighted Hamming distance is calculated as the weighted summation of the difference between a symbol bitmap and a dictionary entry bitmap. Zhang, Danskin, and Yong have also proposed a dissimilarly measure based on cross-entropy which is implemented as WXOR with specific weights.[10,11] In addition, various methods have been proposed to solve the dictionary design problem. These methods typically cluster the symbols into groups, according to a dissimilarity measure, using K-means clustering or a minimum spanning tree.[12–14] Within each group, one dictionary entry is used to represent the all the symbols of that group.

In this paper, we propose a novel method for measuring the dissimilarity between symbols, which is based on conditional-entropy estimation (CEE). The CEE distance measure provides a fast and accurate method to estimate the number of bits required to encode a symbol using its associated dictionary entry. This method is based on the estimation of the entropy of a symbol conditioned on its associated dictionary entry. A fundamental difference between our proposed CEE distance measure and the cross-entropy measure of[10,11] is that our CEE algorithm estimates the conditional entropy from the statistics of the image being encoded, which leads to a more accurate estimate of the bitrate. In addition, we incorporate our CEE in a cluster algorithm to design the dictionary used for encoding the binary document.

Compared to the current dictionary design algorithm, our dictionary design and indexing substantially improves the lossless JBIG2 compression ratio. Finally, as a learning-based prior model, our dictionary learning method has a wide range of potential applications, such as image reconstruction,[15,16] compressed sensing,[17] and other kinds of dictionary-based image compression.[18]

The organization of the rest of the paper is as follows. In Sec. 2, we formulate the dictionary design as a mathematical optimization problem, and describe the overall approach of our CEE algorithm. In Sec. 3, we describe our CEE algorithm in detail, and compare our algorithm with the conventional algorithms. Section 4 presents the experimental results.

2. PROBLEM FORMULATION

The JBIG2 encoder extracts a sequence of symbols from the text region and then encodes them using a dictionary. More specifically, let $\{\boldsymbol{y}_i\}_{i=1}^N$ denote the N symbols that are extracted from the document. Each symbol $\boldsymbol{y}_i$ contains the bitmap of the i^{th} symbol on the page, and each symbol is encoded using a corresponding dictionary entry $\boldsymbol{d}_j$ selected from a complete dictionary $\boldsymbol{D} = \{\boldsymbol{d}_j\}_{j=1}^M$, where M is the number of entries in the dictionary. Furthermore, let $j = f(i)$ denote the function that maps each individual symbol, $\boldsymbol{y}_i$, to its corresponding dictionary entry, $\boldsymbol{d}_j$.

Let $R_J(\boldsymbol{D}, f)$ denote the number of bits required to encode the symbols using the dictionary, $\boldsymbol{D}$, and the mapping, $j = f(i)$. Then we can approximate $R_J(\boldsymbol{D}, f)$ by the formula

$$R_J(\boldsymbol{D}, f) \cong \sum_{i=1}^{N} \left[R_y(\boldsymbol{y}_i|\boldsymbol{d}_{f(i)}) + C_1 + \log_2(M)\right] + \sum_{j=1}^{M} \left[R_d(\boldsymbol{d}_j) + C_2\right] \ , \tag{1}$$

where the first summation represents the bits used to encode the symbols, and the second summation represents the bits used to encode the dictionary. In the first sum, the term $R_y(\boldsymbol{y}_i|\boldsymbol{d}_{f(i)})$ represents the bits required to encode the binary bitmap of the symbol $\boldsymbol{y}_i$, using the dictionary entry $\boldsymbol{d}_{f(i)}$. The term C_1 is a constant that denotes the overhead (in bits) required for encoding the symbol's width, height, and position; and the term $\log_2(M)$ accounts for the bits required to encode the index of the dictionary entry.

In the second sum, the term $R_d(\boldsymbol{d}_j)$ represents the bits required to encode the binary bitmap of the dictionary entry $\boldsymbol{d}_j$; and the term C_2 is a constant that denotes the overhead (in bits) required for encoding the dictionary entry's width and height.

Since we are only considering lossless JBIG2 encoding, our only objective will be to minimize the total number of bits in the encoding. As shown above in (1), the total number of bits is determined by the dictionary $\boldsymbol{D}$ and the mapping f. In other words, the bitrate minimization of (1) only depends on construction of the best dictionary, $\boldsymbol{D}^*$, and selection of the best indexing or mapping function, $f^*(i)$.

$$\{\boldsymbol{D}^*, f^*\} = \underset{\boldsymbol{D}, f}{\operatorname{argmin}}\, R_J(\boldsymbol{D}, f) \tag{2}$$

Assuming the dictionary $\boldsymbol{D}$ is given, selecting the best dictionary entry for each of the symbols needs to minimize the total number of bits required to encode the symbols using the dictionary as a reference. Formally, this can be expressed as

$$f^* = \underset{f}{\operatorname{argmin}} \sum_{i=1}^{N} \left[R_y(\boldsymbol{y}_i|\boldsymbol{d}_{f(i)})\right] \ . \tag{3}$$

In practice, the encoding of pervious symbols can have some effect on the bitrate achieved in the encoding of the current symbol due to the adaptation of the arithmetic encoder. However, we will approximate the bitrate for each pixel as being independent of the past encodings. With this approximation, each term $R_y(\boldsymbol{y}_i|\boldsymbol{d}_{f(i)})$ in (3) is only dependent on a single value of $f(i)$; and the best dictionary entry for the symbol $\boldsymbol{y}_i$ is given by

$$f^*(i) = \underset{j}{\operatorname{argmin}}\, R_y(\boldsymbol{y}_i|\boldsymbol{d}_j) \, . \tag{4}$$

However, computing the precise value of $R_y(\boldsymbol{y}_i|\boldsymbol{d}_j)$ for each symbol $\boldsymbol{y}_i$ is too computationally expensive to be practical. This is because in order to determine the precise value of $R_y(\boldsymbol{y}_i|\boldsymbol{d}_j)$ one would first need to run the encoder for all previous symbols, $\boldsymbol{y}_r$ for $r < i$, and then encode the symbol $\boldsymbol{y}_i$ (from the same initial state) for each of the M possible dictionary entries, $\boldsymbol{d}_j$. Therefore, we will replace $R_y(\boldsymbol{y}_i|\boldsymbol{d}_j)$ in (4) with an approximation, $\tilde{R}_y(\boldsymbol{y}_i|\boldsymbol{d}_j)$, which can be calculated efficiently. Using the substitution, we obtain the index of the optimal dictionary entry for the symbol $\boldsymbol{y}_i$ by

$$\tilde{f}^*(i) = \underset{j}{\operatorname{argmin}}\, \tilde{R}_y(\boldsymbol{y}_i|\boldsymbol{d}_j) \, . \tag{5}$$

3. DICTIONARY ENTRY SELECTION

In this section, we discuss how to select the best dictionary entry for a given symbol. In the first subsection, we briefly review the conventional methods for dictionary entry selection and show the drawbacks of these methods. Then in the second subsection, we propose a novel method for dictionary entry selection based on conditional entropy estimation.

3.1 Conventional methods

Conventional methods select the dictionary entry by minimizing a measure of dissimilarity between the symbol $\boldsymbol{y}_i$ and the dictionary entry $\boldsymbol{d}_j$. Dissimilarity measures widely used for binary signals include the Hamming distance, known as XOR,[7] weighted Hamming distance, known as WXOR,[8,9] and cross-entropy based on a low pass filter, known as CE-LPF.[10,11]

Let w_i and h_i denote the width and the height, respectively, of the i^{th} symbol. The XOR between the symbol $\boldsymbol{y}_i$ and the dictionary entry $\boldsymbol{d}_j$ is calculated using the following formula

$$d_{XOR}(\boldsymbol{y}_i, \boldsymbol{d}_j) = \frac{\sum_{s\in\{0,\ldots,w_i-1\}\times\{0,\ldots,h_i-1\}} [y_i(s) - d_j(s)]^2}{w_i \cdot h_i} , \tag{6}$$

where the parameter $s = (s_\alpha, s_\beta)$ is used to explicitly denote the 2-D coordinates. Here, s_β is the horizontal coordinate and s_α is the vertical coordinate. Note that if a dictionary entry has a size that is different from that of a given symbol, this dictionary entry will not be considered as this symbol's associated dictionary entry. In this case, d_{XOR} will not be calculated.

The WXOR between the symbol $\boldsymbol{y}_i$ and dictionary entry $\boldsymbol{d}_j$ is computed as

$$d_{WXOR}(\boldsymbol{y}_i, \boldsymbol{d}_j) = \frac{\sum_{s\in\{0,\ldots,w_i-1\}\times\{0,\ldots,h_i-1\}} a(s)(y_i(s) - d_j(s))^2}{w_i \cdot h_i} , \tag{7}$$

where

$$a(s) = \frac{\sum_{r\in\{-1,0,+1\}\times\{-1,0,+1\}} [y_i(s+r) - d_j(s+r)]^2}{9} . \tag{8}$$

The CE-LPF between the symbol $\boldsymbol{y}_i$ and dictionary entry $\boldsymbol{d}_j$ is computed as

$$d_{CE-LPF}(\boldsymbol{y}_i, \boldsymbol{d}_j) = \sum_{s\in\{0,\ldots,w_i-1\}\times\{0,\ldots,h_i-1\}} -\log\left(1 - |y_i(s) - \tilde{d}_j(s)|\right) , \tag{9}$$

where $\tilde{\boldsymbol{d}}_j$ is obtained by applying a 3×3 low pass filter g to the dictionary entry $\boldsymbol{d}_j$. The filter g used in[10,11] is fixed as

$$g = \begin{bmatrix} 1/36 & 4/36 & 1/36 \\ 4/36 & 16/36 & 4/36 \\ 1/36 & 4/36 & 1/36 \end{bmatrix} . \tag{10}$$

However, to select the dictionary entry by minimizing one of the above dissimilarity measures (XOR, WXOR, and CE-LPF) often does not provide the optimal dictionary entry to minimize R_y. This is because none of them are good estimates of R_y. To illustrate this, we give one example here by compressing the binary document image *img01.pbm* using the JBIG2 encoder. The test image *img01.pbm*, shown in Fig. 1, was scanned at 300 dpi and has size 3275×2525 pixels. The experiment using this image shows that the sample correlation between the number of bits R_y and d_{XOR} is only 0.4215. The sample correlation between R_y and d_{WXOR} is 0.5410, and the sample correlation between R_y and d_{CE-LPF} is 0.5969. The results are illustrated in Figs. 2 (a), (b), and (c).

3.2 Conditional entropy estimation

In this section, we describe our conditional entropy estimation (CEE) algorithm, which provides a more accurate approximation of $R_y(\boldsymbol{y}_i|\boldsymbol{d}_j)$ than conventional methods. The fundamental idea of our CEE algorithm is to approximate $R_y(\boldsymbol{y}_i|\boldsymbol{d}_j)$ by an estimate of the conditional information (in bits) of the symbol given the dictionary entry. So we have that

$$\tilde{R}_y(\boldsymbol{y}_i|\boldsymbol{d}_j) = \hat{I}_s(\boldsymbol{y}_i|\boldsymbol{d}_j) , \tag{11}$$

where $\hat{I}_s(\boldsymbol{y}_i|\boldsymbol{d}_j)$ is an estimate of the conditional entropy of the symbol $\boldsymbol{y}_i$ given the dictionary entry $\boldsymbol{d}_j$. The value of $\hat{I}_s(\boldsymbol{y}_i|\boldsymbol{d}_j)$ is obtained by

$$\hat{I}_s(\boldsymbol{y}_i|\boldsymbol{d}_j) = -\log \hat{P}(\boldsymbol{y}_i|\boldsymbol{d}_j) , \tag{12}$$

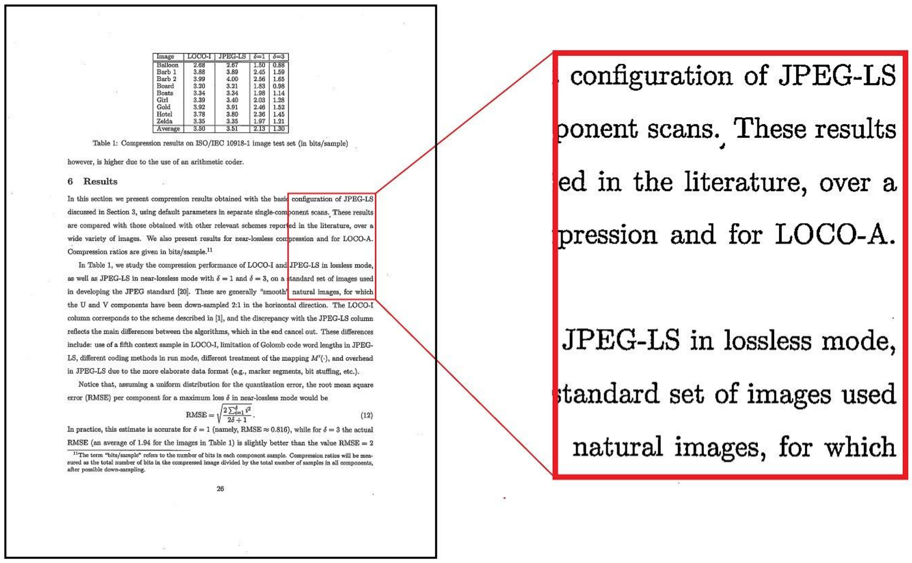


Figure 1. The test image *img01.pbm* with a region (marked by the red rectangle) zoomed in. We use this image for the comparison between XOR, WXOR, CE-LPF, and CEE.

where the term $\hat{P}\left(\boldsymbol{y}_i|\boldsymbol{d}_j\right)$ is the conditional probability estimate of $\boldsymbol{y}_i$ conditioned on $\boldsymbol{d}_j$. Unfortunately, it is not easy to obtain the value of $\hat{P}\left(\boldsymbol{y}_i|\boldsymbol{d}_j\right)$ directly, since both $\boldsymbol{y}_i$ and $\boldsymbol{d}_j$ are high dimensional random variables, and $\hat{P}\left(\boldsymbol{y}_i|\boldsymbol{d}_j\right)$ would have a complicated density function with multiple parameters. In order to solve this problem, we decompose $\hat{P}\left(\boldsymbol{y}_i|\boldsymbol{d}_j\right)$ into a product of several simpler probability density functions based on the following conditional independency assumption: We assume that conditioned on their reference context, the symbol pixels are independent of one another. The reference context for $y_i(s)$, denoted by $\boldsymbol{c}(s,i,j,\boldsymbol{D})$, is a 10-dimensional vector consisting of 4 causal neighborhood pixels of $y_i(s)$ in $\boldsymbol{y}_i$ and 6 non-causal neighborhood pixels of $d_j(s)$ in $\boldsymbol{d}_j$. With this conditional independency assumption, we have

$$\hat{P}\left(\boldsymbol{y}_i|\boldsymbol{d}_j\right) = \prod_{s\in\{0,\ldots w_i-1\}\times\{0,\ldots h_i-1\}} \hat{p}\left(y_i(s)|\boldsymbol{c}(s,i,j,\boldsymbol{D})\right) \,, \tag{13}$$

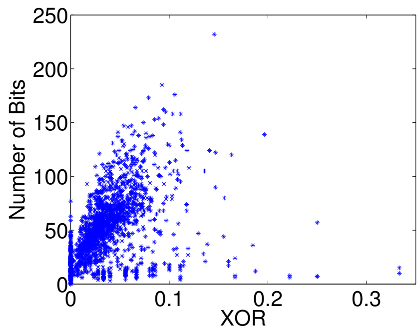
where the term $\hat{p}\left(y_i(s)|c(s,i,j,\boldsymbol{D})\right)$ is the conditional probability estimation for the symbol pixel $y_i(s)$ conditioned on its reference context. In our experiment, we set up a Bayesian framework to train the parameters of $\hat{p}\left(y_i(s)|c(s,i,j,\boldsymbol{D})\right)$. If training samples are limited, we constrain the function $\hat{p}\left(y_i(s)|c(s,i,j,\boldsymbol{D})\right)$ to be a linear function of $y_i(s)$ and $c(s,i,j,\boldsymbol{D})$, and use the SMT-based regression in[19] to estimate the parameters of the model.

In short, our CEE algorithm approximates $R_y(\boldsymbol{y}_i|\boldsymbol{d}_j)$ using the following equation

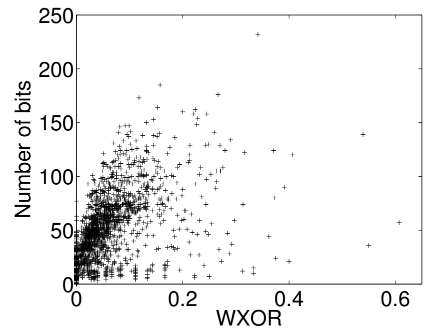
$$\tilde{R}_y(\boldsymbol{y}_i|\boldsymbol{d}_j) = \sum_{t\in\{0,\ldots w_i-1\}\times\{0,\ldots h_i-1\}} -\log\hat{p}\left(y_i(s)|\boldsymbol{c}(s,i,j,\boldsymbol{D})\right) \,. \tag{14}$$

Substituting (14) into (5) and minimizing (5) produces the index of the optimal dictionary entry selected for the symbol $\boldsymbol{y}_i$ using the CEE algorithm.

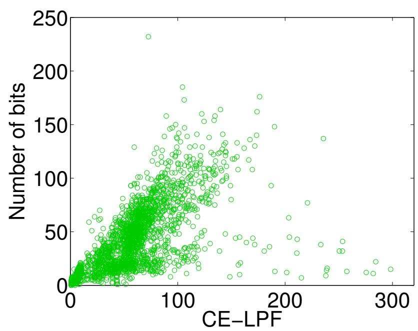
The experiment using the same image *img01.pbm* shows that the sample correlation between the CEE value and the number of bits R_y is 0.9832, as illustrated in Fig. 2 (c). In our study, more experiments were conducted with various kinds of binary document images to demonstrate the approximation accuracy of CEE. The detailed experiment description and more numerical results will be presented in Sec. 4.



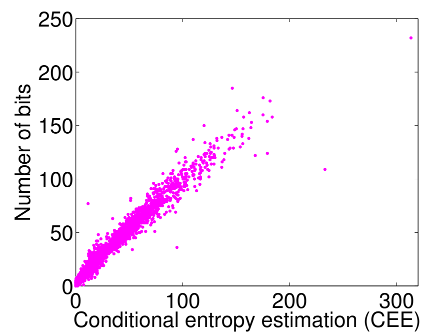
(a) XOR experimental result



(b) WXOR experimental result



(c) CE-LPF experimental result



(d) CEE experimental result

Figure 2. Comparison of the performance of different methods for estimating the number of bits needed to encode the symbols in the image *img01.pbm*. Each dot in the figures corresponds to one symbol in the test image shown in Fig. 1. For each of the subfigures, the vertical coordinate of each dot is the number of bits used to encode the symbol using the associated dictionary entry in the JBIG2 bitstream. In the subfigure (a), the horizontal coordinates are the XOR between the symbols and the associated dictionary entries. In the subfigure (b), the horizontal coordinates are the WXOR between the symbols and the associated dictionary entries. In the subfigure (c), the horizontal coordinates are the CE-LPF between the symbols and the associated dictionary entries. In the subfigure (d), the horizontal coordinates are our CEE approximation of the number of bits required to encode symbols using the associated dictionary entries. As can be seen in the figure, CEE has a much higher correlation with the number of bits required to encode symbols using the dictionary than XOR, WXOR, or CE-LPF.

4. EXPERIMENTAL RESULTS

In this section, we compare the bitrate approximation accuracy of our CEE algorithm with the accuracy of the conventional methods XOR,[7] WXOR,[8,9] and CE-LPF.[10,11] Among these conventional methods, XOR[7] and WXOR[8,9] are widely used. We also compare the compression ratio obtained by using the CEE-based dictionary design with the compression ratio obtained with other widely used dictionary design techniques. The test images we used are 41 scanned binary document images. All of them were scanned at 300 dpi, and have size 3275×2525 pixels. The test images contain mainly text, but some of them also contain line art, tables, and generic graphical elements, but no halftones. The text in these test images has various typefaces and font sizes. More details of the experimental procedure are provided in the following subsections.

4.1 Number of bits estimation results

In order to evaluate the performance of our CEE algorithm, we estimate the correlation (denoted by ρ_{CEE}) between the CEE estimation and R_y the number of bits used to encode a symbol using its associated dictionary entry. For comparison, we also estimate the correlation between Hamming distance d_{XOR} and R_y, the correlation between weighted Hamming distance d_{WXOR} and R_y, and the correlation between d_{CE-LPF} and R_y. These correlation coefficients are denoted by ρ_{XOR}, ρ_{WXOR}, and ρ_{CE-LPF} respectively.

The correlation ρ_{CEE} is estimated using the following procedure: First, we use the JBIG2 encoder to encode a page of a binary document image containing N symbols. Then we obtain N measurements for R_y denoted by $\{r_i | i = 1, \ldots, N\}$, and N corresponding measurements for the CEE estimation denoted by $\{\tilde{r}_i | i = 1, \ldots, N\}$. Last, we calculate the sample correlation using these N sample measurement pairs as the correlation estimation $\hat{\rho}_{CEE}$.

$$\hat{\rho}_{CEE} = \frac{\sum_{i=1}^{N}(r_i - \bar{r})(\tilde{r}_i - \bar{\tilde{r}})}{\sqrt{\sum_{i=1}^{N}(r_i - \bar{r})^2}\sqrt{\sum_{i=1}^{N}(\tilde{r}_i - \bar{\tilde{r}})^2}} \,, \tag{15}$$

$$\text{where } \bar{r} = \frac{1}{N}\sum_{i=1}^{N} r_i \,, \; \bar{\tilde{r}} = \frac{1}{N}\sum_{i=1}^{N} \tilde{r}_i \,.$$

We encoded all 41 images in the test image set, and obtained 41 estimations for ρ_{CEE}. Afterwards, we averaged these 41 estimations to obtain the averaged correlation estimation $\bar{\rho}_{CEE}$ shown in Table 1. We calculated and averaged the estimation of the correlation ρ_{XOR}, ρ_{WXOR}, and ρ_{CE-LPF} using the same scheme with the same test image set. The averaged correlation estimation of ρ_{XOR}, ρ_{WXOR}, and ρ_{CE-LPF} are also shown in Table 1.

Table 1. Performance of XOR, WXOR, CE-LPF, and CEE in estimating the number of bits needed to encode a page of symbols, as indicated by correlation.

	Averaged correlation estimation
XOR	0.561 ± 0.010
WXOR	0.455 ± 0.016
CE-LPF	0.611 ± 0.018
CEE	0.955 ± 0.003

4.2 Compression ratio

In this section, we investigate the compression ratio improvement obtained by using CEE. The widely used one-pass (OP) dictionary design algorithm proposed in[4] is listed for comparison. The results of the lossless TIFF compression algorithm are also listed for comparison. Since we use lossless compression for all the dictionary design algorithms, only the compression ratios using different algorithms are compared, and no distortion is considered.

The OP algorithm can be based on XOR or WXOR. The OP algorithm based on XOR (OP-XOR) requires one to specify a threshold parameter T_{XOR}. The compression ratio of the encoder using OP-XOR is sensitive to T_{XOR}. The optimal T_{XOR} value which minimizes the JBIG2 bitstream file size for different images can range from 0.01 to 0.09. Therefore, we compressed each image with OP-XOR four times using $T_{XOR} = 0.01$, 0.03, 0.06, and 0.09, respectively. In our experiment, the bitstream file size obtained by using OP-XOR is the averaged bitstream file size obtained using these different threshold values.

The OP algorithm based on WXOR (OP-WXOR) also requires a threshold parameter T_{WXOR}. The compression ratio of the encoder using OP-WXOR is not sensitive to T_{WXOR}. We find $T_{WXOR} = 0.27$ works well for most of the images. In our experiment, the bitstream file size for OP-WXOR is obtained by setting $T_{WXOR} = 0.27$.

Our CEE algorithm is utilized to improve the compression ratio in the following two methods. In the first method, we construct the dictionary using OP-WXOR, and use CEE to select the optimal dictionary entry $\boldsymbol{d}_{f(i)}$ for each of the symbols (CEE-based dictionary indexing). In the second method, we incorporate our CEE algorithm into the agglomerative clustering algorithm to obtain the optimal dictionary and indexing (CEE-based

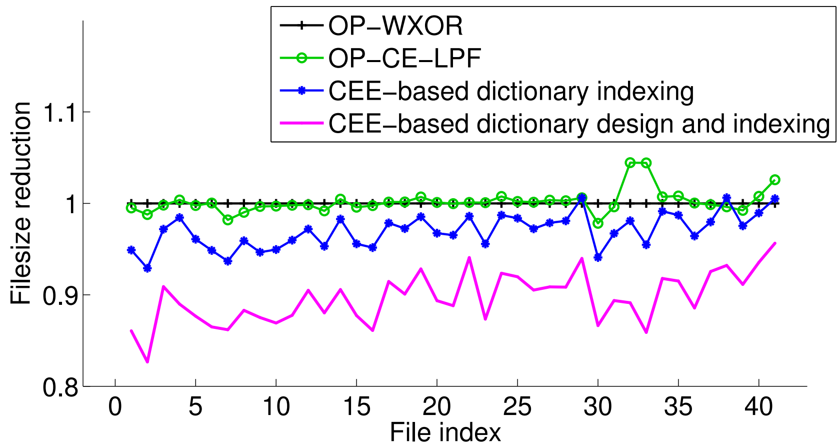


Figure 3. The filesize reduction (compared with the filesize obtained by using OP-WXOR) for each of the test images.

dictionary design and indexing). Note that our dictionary entry selection should also work with other dictionary construction methods, such as K-SVD[15] or minimum spanning tree.[12]

The compression ratio experimental results are list in Table 2. The overall bitstream file size in the table is the sum of the bitstream file sizes for all the 41 test images in the test set described in the beginning of this section. The compression ratio is calculated as the raw image file size of all the test images divided by the overall bitstream file size.

Table 2. Compression performance comparison between different dictionary construction algorithms. The overall bitstream file size listed in the table is the sum of the bitstream file sizes of all 41 test images. The compression ratio is calculated as the raw image file size of all the test images divided by the overall bitstream file size.

Dictionary design method	Overall bitstream file size	Compression ratio
Lossless-TIFF	2.20 MB	19.37
OP-XOR	1.45 MB	29.36
OP-WXOR	1.31 MB	32.54
OP-CE-LPF	1.31 MB	32.54
CEE-based dictionary indexing	1.26 MB	33.83
CEE-based dictionary design and indexing	1.15 MB	37.07

As shown in Table 2, our CEE-based dictionary design and indexing improves the compression ratio by 26.26% compared with the OP-XOR, and 14.45% compared with the OP-WXOR. Note that the bitstream we generate is still compatible with the standard JBIG2, and can be decoded by a public JBIG2 decoder. According to our experiment, our algorithm is efficient enough to be implemented in embedded systems, such as multi-functional printers.

4.3 Filesize reduction for each of the test images

In this section, we demonstrate more detailed experiment results by present the filesize reduction obtained by using our CEE for each of the test images. Since the OP-WXOR dictionary design is widely used and considered as one of the most cutting-edge technologies, we use the bitstream filesize for OP-WXOR as the baseline, and for any other given algorithm, we calculate its filesize reduction as the bitstream filesize using this algorithm over the bitstream filesize using OP-WXOR.

As shown in Fig. 3, the CEE-based dictionary indexing reduces the filesize (compared to OP-WXOR) in 38 out of 41 cases. The CEE-based dictionary design and indexing can always produce smaller bitstream filesizes than OP-WXOR or OP-CE-LPF.

5. CONCLUSION

In this paper, we propose a novel dictionary design algorithm that can construct better dictionary entries and use those dictionary entries more effectively. The main novelty of our dictionary design is that it provides a more accurate and efficient approximation (CEE) of the number of bits required to encode a symbol using its associated dictionary entry. We applied our dictionary design in the JBIG2 compression and achieved promising results. The experimental results show that our CEE can provide much more accurate prediction with just a little more computational cost compared with conventional methods including XOR and WXOR. The averaged sample correlation between CEE and the number of bits required to encode each symbol in a sample document is larger than 90%, while the conventional methods can only provide an averaged sample correlation around 50%. The experimental results also show that the compression ratio of the JBIG2 encoder incorporating CEE is about 20% higher than the conventional JBIG2 encoders.

ACKNOWLEDGMENT

We would like to thank the Hewlett-Packard Company for their support of this research.

REFERENCES

[1] "JBIG2 final draft international standard," *ISO/IEC JTC1/SC29/WG1N1545* (Dec. 1999).

[2] Arps, R. B. and Truong, T. K., "Comparison of international standards for lossless still image compression," *Proc. of the IEEE* **82**, 889–899 (1994).

[3] Fumitaka, O., William, R., Ronald, A., and Corneliu, C., "JBIG2 - the ultimate bi-level image coding standard," in [*Proc. of IEEE Int'l Conf. on Image Proc.*], 140–143 (2000).

[4] Howard, P. G., Kossentini, F., Martins, B., Forchhammer, S., and Rucklidge, W. J., "The emerging JBIG2 standard," *IEEE Trans. on Circuits and Systems for Video Technology* **8**, 838–848 (1998).

[5] Ortego, A. and Ramchandran, K., "Rate-distortion methods for image and video compression," *IEEE Signal Proc. Magazine* **15**, 23–50 (1998).

[6] Ruf, M. J. and Modestino, J. W., "Operational rate-distortion performance for joint source and channel coding of images," *IEEE Trans. on Image Processing* **8**, 305–320 (1999).

[7] Holt, M. J. J., "A fast binary template matching algorithm for document image data cmpression," in [*Proc. of IEEE Int'l Conf. on Pattern Recognition*], 230–239 (1988).

[8] Pratt, W., Capitant, P., Chen, W.-H., Hamilton, E., and Wallis, R., "Combined symbol matching facsimile data compression system," *Proc. of the IEEE* **68**, 786–796 (1980).

[9] Figuera, M., Yi, J., and Bouman, C. A., "A new approach to JBIG2 binary image compression," in [*Proc. SPIE 6493, Color Imaging XII: Processing, Hardcopy, and Applications*], 649305 (2007).

[10] Zhang, Q. and Danskin, J. M., "Entropy-based pattern matching for document image compression," in [*Proc. of IEEE Int'l Conf. on Image Proc.*], 221–224 (1996).

[11] Zhang, Q., Danskin, J. M., and Young, N. E., "A codebook generation algorithm for document image compression," in [*1997 IEEE Data Compression Conf.(DCC)*], 300–309 (1997).

[12] Ye, Y. and Cosman, P. C., "Dictionary design for text image compression with JBIG2," *IEEE Trans. on Image Processing* **10**, 818–828 (2001).

[13] Ye, Y., Schilling, D., Cosman, P. C., and Ko, H. H., "Symbol dictionary design for the JBIG2 standard," in [*Data Compression Conference'00*], 33–42 (2000).

[14] Ye, Y. and Cosman, P. C., "Fast and memory efficient text image compression with JBIG2," *IEEE Trans. on Image Processing* **12**, 944–956 (2003).

[15] Elad, M. and Datsenko, D., "Example-based regularization deployed to super-resolution reconstruction of a single image," *Comput. J.* **52**, 15–30 (Jan. 2009).

[16] Jin, P., Haneda, E., and Bouman, C., "Implicit Gibbs prior models for tomographic reconstruction," in [*the 46th Asilomar Conference on Signals, Systems and Computers*], (Nov. 2012).

[17] Rubinstein, R., Bruckstein, A. M., and Elad, M., "Dictionaries for sparse representation modeling," *Proc. of the IEEE* **98**, 1045–1057 (2010).

[18] Elad, M., Goldenberg, R., and Kimmel, R., "Low bit-rate compression of facial images," *IEEE Trans. on Image Processing* **16**, 2379–2383 (2010).

[19] Cao, G., Guo, Y., and Bouman, C. A., "High dimensional regression using the sparse matrix transform (SMT)," in [*Proc. of IEEE Int'l Conf. on Acoust., Speech and Sig. Proc.*], 1870–1873 (2010).