Biased Run-Length Coding of Bi-Level Classification Label Maps of Hyperspectral Images

Amir L. Liaghati, W. David Pan, *Senior Member, IEEE*, Zhuocheng Jiang

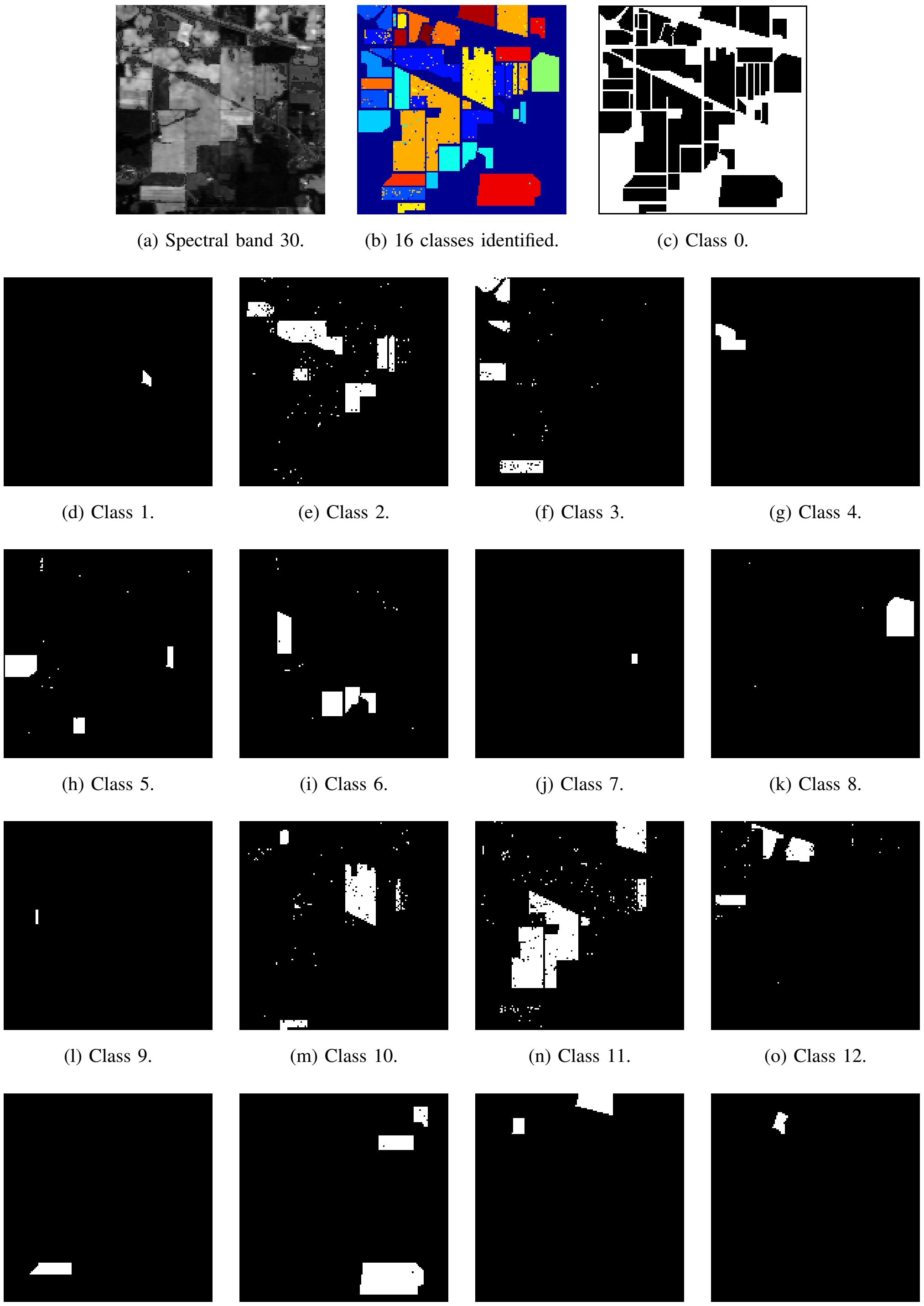
***Abstract*—For efficient coding of bi-level sources with some dominant symbols often found in classification label maps of hyperspectral images, we proposed a novel biased run-length (BRL) coding method, which codes the most probable symbols separately from other symbols. To determine the conditions in which the BRL coding method would be effective, we conducted an analysis of the method using statistical models. We first analyzed the effect of two-dimensional blocking of pixels which were assumed to have generalized Gaussian distributions. The analysis showed that the resulting symbol blocks tended to have lower entropies than the original source without symbol blocking. We then analyzed the BRL coding method applied on the sequence of block symbols characterized by a first-order Markov model. Information-theoretic analysis showed that the BRL coding method tended to generate codewords that have lower entropies than the conventional run-length coding method. Furthermore, numerical simulations on lossless compression of actual data showed improvement of the state-of-the-art. Specifically, end-to-end implementation integrating symbol blocking, BRL and Huffman coding achieved up to 4.3% higher compression than the JBIG2 standard method, and up to 3.2% higher compression than the conventional run-length coding method on classification label maps of the widely used "Indian Pines" dataset.**

I. Introduction

Hyperspectral imaging techniques have been used in a wide array of earth observing applications such as material quantification and target detection. A hyperspectral image is often organized as a three-dimensional dataset with two spatial dimensions and one spectral dimension. As an example, see Fig. 1(a), which is the 30th spectral band (out of a total of 220 bands) from NASA's Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) hyperspectral image dataset [1]. The high spectral resolution makes it possible to address various applications requiring very high discrimination capabilities in the spectral domain. However, the large data volume of hyperspectral images presents a challenge for both data transmission and storage [2]. Our recent work in [3] addressed lossless methods for efficiently compressing arbitrarily shaped "sub-images" belonging to a certain class of particular interest (see Fig. 1 for an example of the bi-level classification label maps for 16 classes resulting from pattern classification via the support vector machine [4]). Any pixel within an hyperspectral image either belongs to a certain class or belongs to any other classes. Therefore, the class label map is a bi-level image, with the same size as that of the original image in any spectral band. The map provides critical spatial information required to reconstruct the pixel values belong to a certain class. Therefore, efficient lossless compression of these bi-level maps would be useful or even critical in some remote sensing applications with severely limited bandwidths [5]. This is a separate problem not addressed in our prior work [3], [6].

Conventional bi-level image compression techniques include run-length coding [7], arithmetic coding [8], and geometric-based coding [9], [10]. In addition, International standards for binary image compression have been developed [11], including JBIG2 [12], and JPEG 2000 [13]. In order to exploit the pixel correlations in both horizontal and vertical directions, our previous work [14] adopted a symbol packing approach, where a binary image was first partitioned into blocks, with each block being scanned in a raster-scan order. The resulting sequence of bi-level symbols was then converted to a binary representation of that block. We observed that in many bi-level images, either the all-1 or the all-0 block symbol tends to be the most probable one among all possible symbols. To take advantage of the redundancy associated with the most probable symbols, we introduced a biased run-length encoding method, which run-length codes only the most probable block symbol. In the following, we first give a brief survey in Section II on the run-length coding methods. We then point out the novelty of the proposed biased run-length method. Section III presents the analysis for proposed biased run-length coding method, based on the mathematical model given in Section IV. Simulation results are given in Section V. The paper is concluded in Section VI.

A. L. Liaghati is with the Boeing Company, 1100 Redstone Gateway SW, Huntsville, AL 35824, USA, and W. D. Pan and Z. Jiang are with the Dept. of Electrical and Computer Engineering, University of Alabama in Huntsville, Huntsville, AL 35899, USA. The first two authors contributed equally to this work.



(a) Spectral band 30. (b) 16 classes identified. (c) Class 0.

(d) Class 1. (e) Class 2. (f) Class 3. (g) Class 4.

(h) Class 5. (i) Class 6. (j) Class 7. (k) Class 8.

(l) Class 9. (m) Class 10. (n) Class 11. (o) Class 12.

(p) Class 13. (q) Class 14. (r) Class 15. (s) Class 16.

Fig. 1: Sample dataset (“Indian Pines”) and the classification label maps. (a) A sample band. (b) 16 classes identified using the support vector machine (SVM) method [6], [14]. (c) Class “0” belongs to the pixels that are not actually classified (due to lack of ground truth for assessment). (d)-(s) Individual classification label maps (pixels belonging to the same class are shown in white, with the pixels of other classes shown in black).

II. Run-Length Coding Methods

A run is a sequence of pixels having an identical value, and the number of such pixels is length of the run. Run-length encoding (RLE) is a very simple form of lossless data compression on which run of data are stored as a single data value and count, rather than as the original run. This is most useful on data that contains many such runs [7], [15]. In some recent work, [16] introduces a so-called extended frequency-directed run length coding for test data compression, and [17] proposes a variable prefix dual run length coding for VLSI test data compression, among other similar run length coding methods proposed for this purpose [18]–[20]. Furthermore, [21] combined run-length coding with Huffman coding for lossless compression of fluoroscopy medical images. [22] presented a lossless audio coding method using Burrows-Wheeler Transform with the combination of run-length coding. Additional work on application of run-length coding includes papers on image and data compression [23]–[26], feature and region data extraction [27], [28], image quality index [29] and image hiding [30]. To improve the conventional RLE, [31] introduces an adaptive scheme that encodes run and level separately using adaptive binary arithmetic coding and context modeling. [32] proposed a method to parse the binary data sequences to make run-length coding more efficient, where the run-length code belongs to the family of variable-length constrained sequence codes [33].

In contrast to the exiting methods, we proposed to code only the most probable block symbol to avoid excessive number of short runs (a major source of coding inefficiency). Thus the main novelty of the proposed biased run length (BRL) coding methods lies in its separate special treatment of the most probable symbol from the rest of the block symbols. The run-lengths of the most probable block symbol are entropy coded using a variable-length code such as the Huffman code. On the other hand, we use a separate Huffman code to compress the modified sequence of block symbol values. Fig. 2 shows the block diagram for the biased run length coding method. The goal of this paper is to provide an in-depth analysis of the compression performance of the proposed biased run-length coding method based on statistical model, instead of relying on empirical data. The analysis will gain insight into why and when the proposed method would be effective.

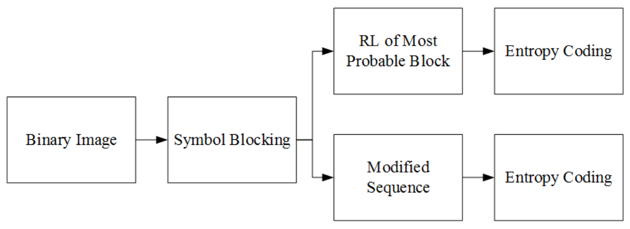


Fig. 2: The biased run length coding method.

III. Biased Run-length Coding Method

In order to exploit the pixel correlations in both horizontal and vertical directions, we proposed a symbol packing approach in order to pack more pixels in a block symbol, because the background appears to contain the majority of the pixels. In addition, objects usually have some "thickness", meaning that there might be more grouped "0" pixels in the image than those isolated "0" pixels. To show that this is indeed the case, we calculated the probability distribution of block symbols for all the bi-level maps considered. Fig. 3 shows the first nine most probable block symbols in descending order from left to right. Note that all 512 possible block symbols will be used in constructing the Huffman code table. Block symbol "511" is the most probable block symbol with probability of 0.9192. Block symbol "0" is the next most probable block with probability of 0.0408. We can see that the probability of the block symbols with same pixel values grouped together (group of white/black pixels) are higher than block symbols with random pattern (isolated black or white dots). Note that using larger blocks would allow us to better exploit the spatial correlations in the source image; however, a large alphabet of block symbols would make the actual implementation of entropy coding (e.g., Huffman coding) overly complicated. Given a block size of $N \times N$, the number of distinct block symbols is 2^{N^2}. In this work, we found the block size of 3×3 pixels (corresponding to $2^9 = 512$ possible block symbols) offered a good tradeoff. A further increase of the block size to, for instance, 4×4, will generate $65,536$ possible block symbols, making the Huffman code table too large to be manageable in practical implementations.

A. Distribution of Block Symbols

In this section, we will examine when the block symbol-based coding is beneficial using statistical model. It was shown that the distribution of integers can be modeled using the generalized Gaussian distribution (GGD) [34]. We assume the probability of block symbols can be shown in discrete Gaussian distribution. Since the block symbols are always positive integers, only the right side of the Gaussian function is chosen. Note that the normalization is required so that the sum of right-sided discrete Gaussian function is still unity. Note that we are not limited to this distribution. We have chosen this distribution because we can readily change its parameters to generate a large set of varying bi-level images. The probability G at given $k\beta$ can be calculated using the following formula, where k is an integer, β is the quantization level, and n is the maximum number of

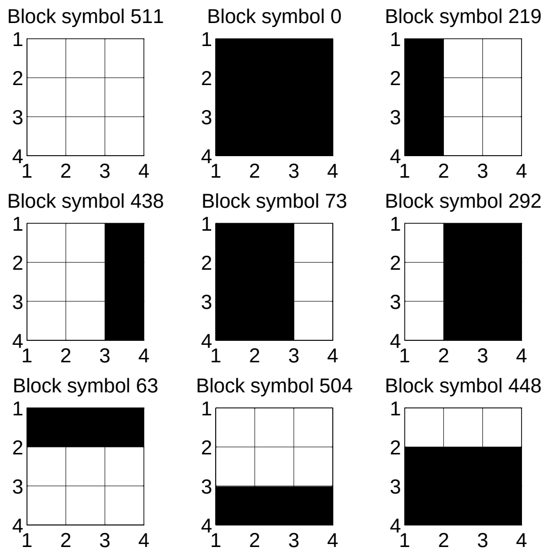


Fig. 3: Some symbol blocks in descending order of probability (from left to right).

block symbols.

$$G(x = k\beta) = \frac{\frac{1}{\sigma\sqrt{2\pi}}\exp(-\frac{(k\beta-\mu)^2}{2\sigma^2})}{\sum_{x=0}^{n\beta}\frac{1}{\sigma\sqrt{2\pi}}\exp(-\frac{(x-\mu)^2}{2\sigma^2})}. \tag{1}$$

Fig. 4 shows the discrete Gaussian Distribution probabilities for different β values. We can see that as β increases, the distribution changes as well, and as a result the entropy varies. The entropy using the block-symbol technique can be calculated using the following equation:

$$\text{Entropy of block symbols} = \frac{-\sum_{n=1}^{2^N} G_n \log G_n}{N}, \tag{2}$$

where N is the number of pixels per block, and 2^N is the number of block symbols. Note that the top part of the equation is the average amount of pixels per block, which needs to be divided by the number of pixels per block to calculate the average number of bits per pixel.

Given the statistical probabilities of block symbols G_n, we seek to extract the probability of “1” from the source probabilities of block symbols. First, we find the probability of “1” in each block symbol, then rearrange it from the most probable to the least probable symbol block and store it in an array “A” which contains 2^N elements. Fig. 3 shows an example of nine first most probable block symbols using the bi-level maps. For instance, the most probable block symbol is 511 which consists of nine 1‘s so the probability of 1 is unity. Next is block symbol 0, which has no 1‘s, and therefore the probability of “1” is 0. Next is 219 which contains six

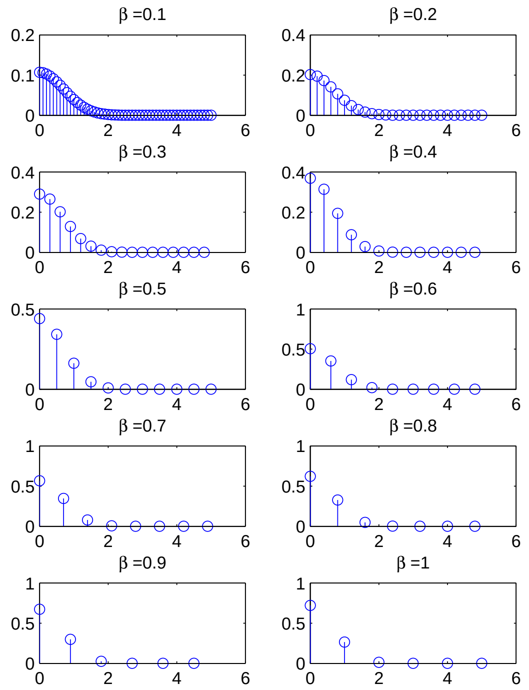


Fig. 4: Discrete Gaussian distribution for different β for $\sigma^2 = 0.5$.

1‘s, and as a result, the probability of “1” is 0.6667. As a result, the probability of “1” (p) can be calculated as:

$$p = \sum_{n=1}^{2^N} A_n G_n. \tag{3}$$

Once the probability of “1” is calculated from block symbols, the entropy of the newly extracted binary source can be calculated using the following equation:

$$H = -p\log_2 p - (1-p)\log_2(1-p). \tag{4}$$

Next, we compared the entropy values between the 2-D block-symbol source and the derived 1-D binary source. It can be seen in Fig. 5 that as σ^2 decreases, entropy decreases as well for both sources. For a given β value, the 1-D source has higher entropy than the 2-D source, which shows the advantage of using block symbols. As β increases, the entropy values of both sources get closer and closer. We also compared the empirical entropy values of actual binary images and the corresponding sequence of block symbol sources, as shown in Fig. 6, where we used 3×3 blocks. Here the probability of 1’s for each image was found based on the actual data instead of a model. We see again blocking helps reduce the source entropy, thereby improving compression efficiency. Note that when the 2-D block symbol sequence

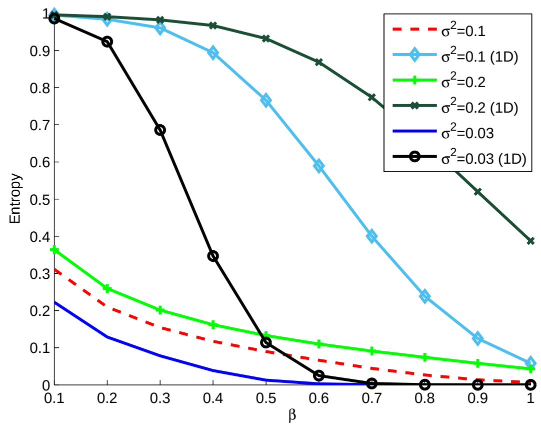


Fig. 5: Entropy comparison between the 1-D source and 3×3 block-based source for different β and σ^2.

has small entropies, the gaps between 1-D and 2-D sequence tend to be small (e.g., maps 1, 7, and 9). This behavior is similar to what is shown in Fig. 5 for a large β value. Overall, symbol blocking appears to be beneficial for compression due to its ability to capture two-dimensional correlations in the original source.

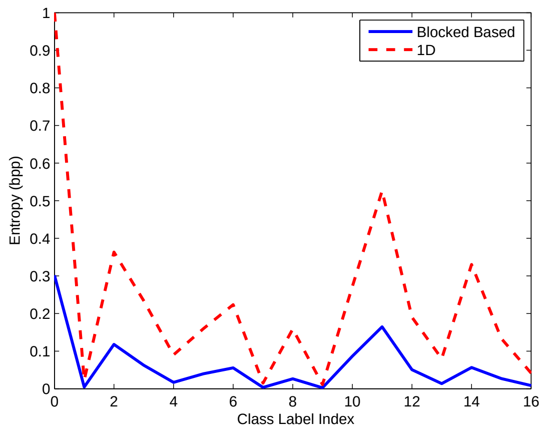
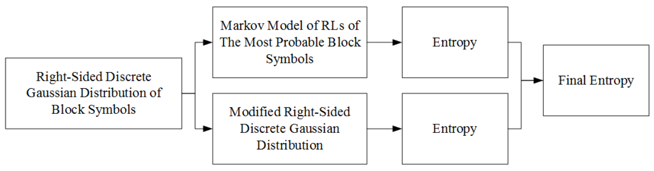


Fig. 6: Entropy comparison between the bi-level classification label maps of "Indian Pines" and the corresponding sequence of 3×3 block-based symbols.

IV. Statistical Model and Analysis

In the biased run-length coding method, the original bi-level data source is segmented into two sources with different distributions. The first source consists of the run-lengths of the most probable block symbol. It can be described by the Markov model using the probability of the most probable block symbol, which is extracted from the original distribution of block symbols. Fig. 7 shows the diagram for the biased run-length coding method using the statistical models.



Right-Sided Discrete Gaussian Distribution of Block Symbols
Markov Model of RLs of The Most Probable Block Symbols
Modified Right-Sided Discrete Gaussian Distribution
Entropy
Entropy
Final Entropy

Fig. 7: Analysis of the biased run-length coding method using the statistical model.

A. Geometric Distributions

Run lengths of binary source can normally be modeled using the geometric distribution as following

$$G(n) = p^n q = p^n(1 - p), \tag{5}$$

where p is the probability of "1", and n is the run length of "1". However, regarding the biased run-length coding method, since the most probable block symbol is only run-length encoded, p is now the probability of the most probable block symbol, and n is the run lengths of the most probable block symbol. However, the geometric distribution model cannot capture the transition probabilities between the most probable block symbol and other block symbols. To this end, we introduce a new Markov model based on the Capon model [35].

B. Markov Model for Distribution of the RLs

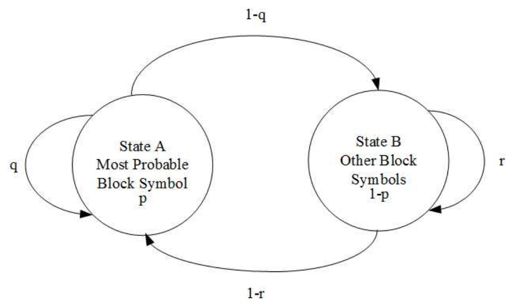


Fig. 8: Transition probabilities between block symbol "511" and other blocks.

In this analysis, we chose to use a first-order, two-state Markov model to characterize inter-symbol correlations in terms of different transition probabilities between block symbols. Fig. 8 shows the transition probabilities between the most probable block symbol and other symbol blocks. p is the probability of the most probable block symbol occurs, $1-p$ is the probability that other blocks symbols occur. Let A represents the state for most probable block symbol, and B represents the state for other block symbols. Given the current block symbol is in state A, the probability of next block symbol being also in A is q. As a result, the probability of next block is in state B (given the current state is in A) is $1-q$. Similarly, the probability of next state is B (given the current state is also B) is r, and the probability of next state is A (given the current state is B) is $1-r$. The following inequality (7) must be satisfied in order to maintain the stability for transitions between states (Eq. 6). Moreover, we want to emphasize that in Capon's original model, the probabilities of runs of white and black pixels are assumed to have the same distribution. However, our mathematical model is uniquely derived using the biased distribution (of the most probable block symbol and rest of the block symbols).

$$p(1-q) = (1-p)(1-r) \tag{6}$$

$$0 < \frac{p(1-q)}{1-p} < 1 \Rightarrow q > 2 - \frac{1}{p} \tag{7}$$

Table I shows the probability distribution of the run lengths of the most probable block symbol using the Markov model. Note that for demonstration purposes, the codewords are shown in binary format, where indeed 1 represents the most probable block symbol, and 0 represents other block symbols. As a result, the Markov model for run-lengths can be defined as the following equation:

$$M(n) = pq^{(n-1)}(1-q). \tag{8}$$

The Markov distribution will be used to calculate the reduced probability of the most probable block symbol using the original probability. Before calculating the entropy, the Markov probability distribution needs to be normalized so the sum of probabilities is equal to unity.

$$M_{normalized}(n) = \frac{M(n)}{\sum_{n=1}^{\infty} M(n)} \tag{9}$$

The entropy of the Markov distribution of the run-lengths can be calculated as

$$H_M = -\sum_{n=1}^{\infty} M_{normalized}(n) \log_2[M_{normalized}(n)]. \tag{10}$$

After the biased run-length coding, the number of merged most probable block will be smaller than the original number of the most probable blocks. For example, if the run-length of a "511" blocks is five, then five consecutive "511" blocks will be reduced to a single merged-block symbol, with its value being five. The average shrinking factor (SF) can be found as:

$$SF = \frac{1}{E[n]} = \frac{1}{\sum_{n=1}^{\infty} nq^{(n-1)}(1-q)} = 1-q. \tag{11}$$

In practical simulations with only finite run-lengths being observed, the average shrinking factor can be estimated as:

$$\overline{SF} = \frac{1}{\hat{E}[n]} = \frac{1}{\sum_{R} n \cdot \text{Prob}[n]}, \tag{12}$$

where $\hat{E}[n]$ is the sample mean of the set (R) of distinct run-lengths (n) ever encountered, and Prob$[n]$ represents the empirical distribution of the run-lengths. Consequently the length of the merged-block sequence can be found as

$$\text{Sequence Length} = N \cdot p \cdot \overline{SF}, \tag{13}$$

where N is the length of the original block symbol sequence.

Table II shows the original symbol blocks "511" versus the new merged blocks for different run-lengths of block symbol "511". Thus the total number of bits for the run-lengths can be calculated as

$$\text{Total \# of bits for RL} = H_M \cdot (N \cdot p \cdot \overline{SF}). \tag{14}$$

TABLE I: Generation of probability distribution of run-lengths using the Markov model. n is the run length of the most probable block symbol.

n	Codeword	Probability
1	10	$p(1-q)$
2	110	$pq(1-q)$
3	1110	$pq^2(1-q)$
4	11110	$pq^3(1-q)$
...	...	...
n	11...0	$pq^{(n-1)}(1-q)$

C. Construction of New Modified Sequence

To construct the modified sequence, we use the original right-sided discrete Gaussian distribution of block symbols in addition to the distribution of RLs. To obtain the original number of each of block symbols B, the original right-sided discrete Gaussian distribution in

Eq. (1) will be multiplied by N, the total number of blocks as shown below:

$$B = G \times N. \tag{15}$$

Then the new modified sequence will have the same distribution of blocks except for the most probable block symbol. That is, the original number of most probable block symbol will be replaced by the new number of block symbols, which is the same as the sequence length calculated in Eq. (13) as shown below:

$$\begin{aligned} &B(1) \text{ (New \# of most probable block symbol)} \\ &\qquad = N \cdot p \cdot \overline{SF}. \end{aligned}$$

Note that B is an array of probabilities, thus $B(1)$ is the first probability value in the array. Once the new distribution of block symbols is constructed, the new "Modified Sequence" (MS) can be calculated by normalizing the block distributions as follows:

$$MS = \frac{B}{\sum_{n=1}^{\infty} B}. \tag{16}$$

Next, the entropy of the modified sequence will be calculated. To calculate the total number of bits for the modified sequence, the entropy of the modified sequence should be multiplied by the number of symbols in the modified sequence. The total number of symbols in the modified distribution can be calculated using the following equation:

$$\begin{aligned} &\text{Total \# of symbols in the modified sequence} \\ &= N - Np + N \cdot p \cdot \overline{SF} = N[1 - p(1 - \overline{SF})] \end{aligned} \tag{17}$$

Next, the total number of bits required for the modified sequence can be found by multiplying the entropy of the modified sequence by the total number of symbols in the modified sequence, as given below:

$$\begin{aligned} &\text{Total \# of bits for MS} \\ &= -\sum_{n=1}^{\infty} MS \log_2(MS) \times N[1 - p(1 - \overline{SF})]. \end{aligned} \tag{18}$$

TABLE II: The "511" blocks vs. the new merged blocks.

n	Original Blocks	Merged Blocks
1	511	511
2	511 511	511
3	511 511 511	511
...	...	...
M	511 ... 511	511

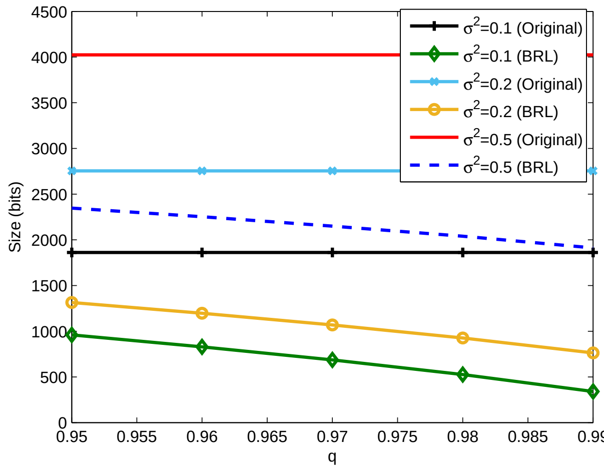


Fig. 9: Size comparison between the biased method and the entropy of the original distribution for $\beta = 0.5$, with different σ^2 and q.

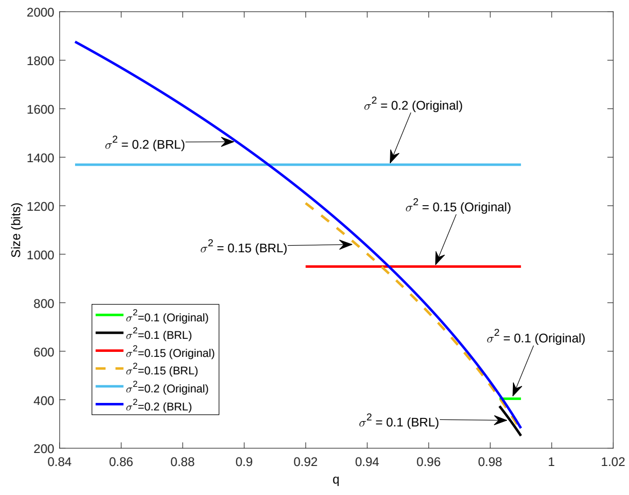


Fig. 10: Comparison of the bitrates between the biased run-length coding method and the entropy of the original distribution for $\beta = 0.8$, with different σ^2 and q values.

Finally, the total number of bits for the biased method can be calculated by adding the total number of bits for the run-lengths (Eq. (14)), and the modified sequence (Eq. (18)).

Fig. 9 shows comparison results on the synthesized images between the proposed biased RL model and the entropy of the original distribution. For this example "N" is 2304, β is 0.5, and σ^2 and q are variables. We can see that for higher q values, the biased run-length coding method is superior to the original distribution in terms of entropy.

Fig. 10 shows another comparison, where $N = 2,304$,

$\beta = 0.8$, σ^2 and q are variables, and the smallest possible values for q for each σ^2 were used. We can see that when σ^2 is very small, even in the worst case when q is the minimum possible value, our BRL coding method has lower entropy than the original distribution. As σ^2 increases, there is a margin for q where the BRL coding method is beneficial. In other words, when q is larger than some threshold, our method is superior, in terms of having a lower entropy than that of the original distribution of block symbols. The threshold values for each distribution vary. For example, when $\sigma^2 = 0.2$, the threshold is about 0.91. The analysis allows us to study different types of bi-level images with varying distributions of block symbols and inter-symbol correlations. For instance, when $\sigma^2 = 0.1$, the BRL method always compresses better than the conventional entropy coding, regardless of the symbol transition probabilities. As σ^2 increases, there are less all-1 block symbols. Thus as q decreases, the compression gain by using the BRL coding method gets reduced. Therefore, in general, we can use this method to select the best compression method (between the BRL coding and the conventional run-length coding methods).

V. Results on Maps of Actual Data

In addition to synthesized maps based on mathematical models, we also tested the proposed method on classification label maps obtained from actual hyperspectral data. As shown in Fig. 11, the BRL coding method achieved outstanding compression on 16 out of 17 bi-level classification label maps given in Fig. 1, with higher compression efficiency than various standards.

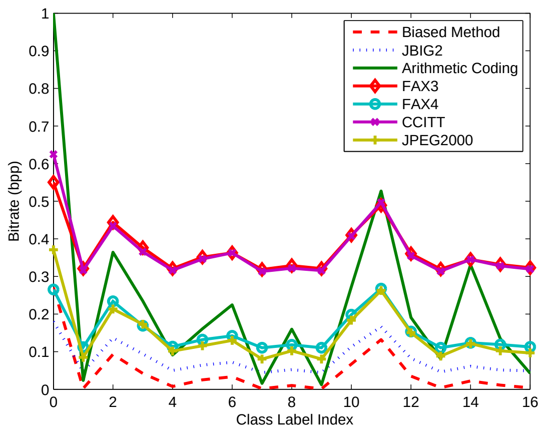


Fig. 11: Compression results for bi-level classification label maps of “Indian Pines”.

As a case study, we compared the empirical entropy values of the biased run-length (BRL) coding and those of the conventional run-length coding method. Fig. 12 shows how to calculate the entropies of the biased RL method as an estimate of the achievable amount of compression. Fig. 13 shows that the BRL coding method would allow for better compression than the conventional run-length coding method for all these maps.

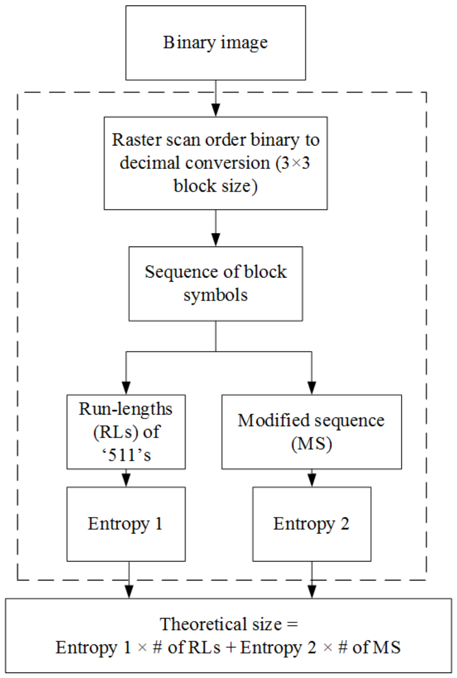

Binary image
Raster scan order binary to decimal conversion (3×3 block size)
Sequence of block symbols
Run-lengths (RLs) of ‘511’s
Modified sequence (MS)
Entropy 1
Entropy 2
Theoretical size = Entropy 1 × # of RLs + Entropy 2 × # of MS

Fig. 12: Empirical entropy calculation for the biased run-length coding method.

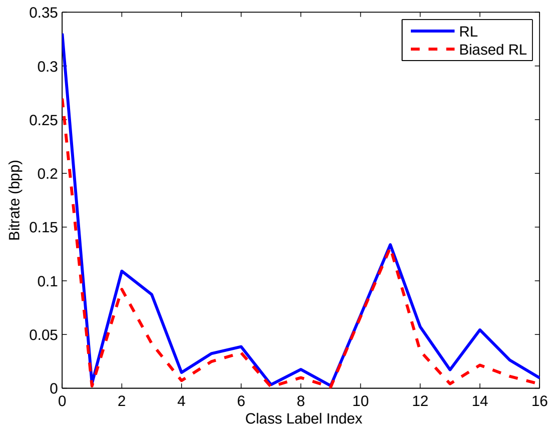
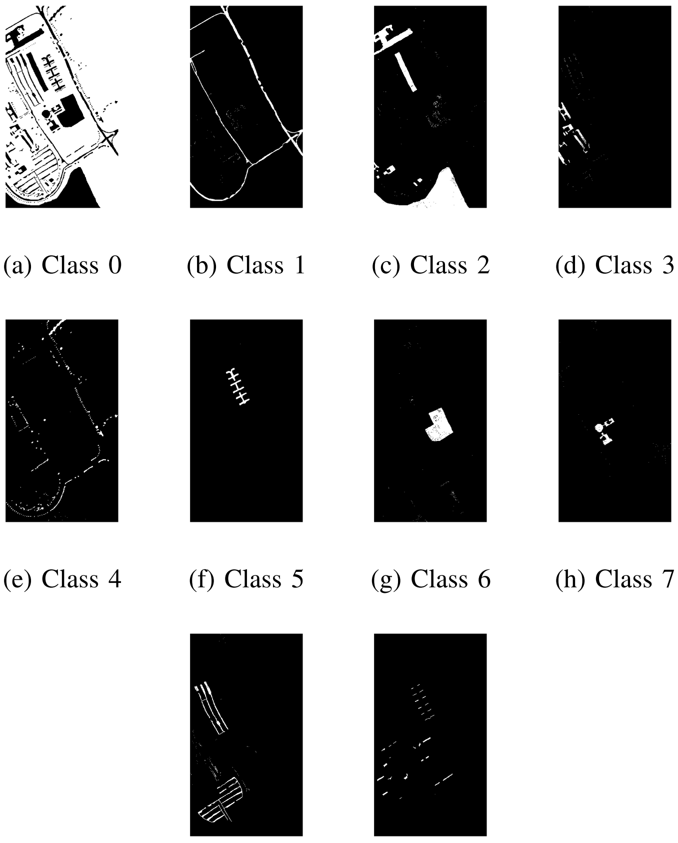


Fig. 13: Bitrates based on entropy calculated for the biased run-length coding method and the conventional run-length coding method.

To more comprehensively evaluate the proposed method, we also provided the results (Fig. 15) of compressing the classification label maps for another hyperspectral image dataset – “Pavia University” (PU) [1], which has much larger (610×340 pixels) classification label maps than the “Indian Pines” dataset. To apply the proposed symbol blocking method (with block size being 3×3), we resized the ten bi-level classification maps (see Fig. 14) to 609×339 pixels. Fig. 15 shows the proposed BRL



(a) Class 0 (b) Class 1 (c) Class 2 (d) Class 3

(e) Class 4 (f) Class 5 (g) Class 6 (h) Class 7

(i) Class 8 (j) Class 9

Fig. 14: Classification label maps of the PU dataset obtained using support vector machine based on the ground truth provided in [1].

method clearly outperforms the arithmetic coding method and the JPEG 2000 standard. The BRL method compresses slightly better than JBIG2 standard for maps 3, 5, 6 and 7. For the remaining maps where the foreground patterns tend to be less “clustered”, JBIG2 performs slightly better as expected due to the most probable block symbols becoming less dominant. Note that the Huffman code table used was trained based on the statistics of the maps for the “Indian Pines” dataset, thereby leading to lost coding efficiency if applied another dataset with varying statistics. This can explain why the advantage of BRL method is less pronounced than the results for the “Indian Pines” dataset shown in Fig. 11.

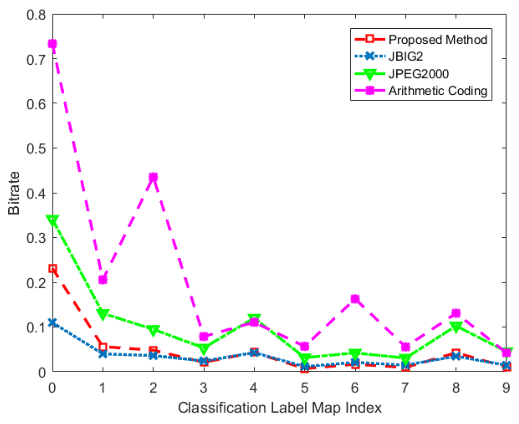


Fig. 15: comparison of compressed bit rates on the PU dataset.

VI. Conclusions

The bi-level classification label maps for hyperspectral images are the overhead information that needs to be compressed efficiently. We proposed to run-length coding the most probable symbol block separately to achieve more efficient compression. We provided model-based analysis on the conditions in which the BRL coding method could provide better compression on binary images than the conventional run-length coding method. End-to-end implementation integrating symbol blocking, BRL and Huffman coding achieved significantly higher compression than arithmetic coding method and the JPEG 2000 standard method. When the Huffman coding table was trained based on maps to be compressed, we can achieved up to 4.3% better compression than the highly optimized JBIG2 standard method for lossless compression of binary images. We expect the coding efficiency can be further improved by training the Huffman code using more data or using adaptive coding methods.

VII. Acknowledgment

We would like to thank Dr. Hongda Shen for providing the classification label maps. We also thank the anonymous reviewers for the help with improving the paper.

References

[1] “Hyperspectral remote sensing scenes data,” http://www.ehu.es/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes.

[2] J. Gonzalez-Conejero, J. Bartrina-Rapesta, and J. Serra-Sagrista, “JPEG 2000 encoding of remote sensing multispectral images with no-data regions,” *IEEE Geosci. Remote Sens. Lett.*, vol. 7, no. 2, pp. 251–255, April 2010.

[3] H. Shen, W. D. Pan, and Y. Wang, "A novel method for lossless compression of arbitrarily shaped regions of interest in hyperspectral imagery," in *Proc. 2015 IEEE SoutheastCon*, April 2015.

[4] F. Melgani and L. Bruzzone, "Classification of hyperspectral remote sensing images with support vector machines," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 8, pp. 1778–1790, Aug. 2004.

[5] W. D. Pan *et al.*, "A new method for compressing quality-controlled weather radar data by exploiting blankout markers due to thresholding operations," in *Proc. of AMS 27th Conference on International Interactive Information and Processing Systems (IIPS11) for Meteorology, Oceanography, and Hydrology*, January 2011.

[6] H. Shen, W. D. Pan, and D. Wu, "Predictive lossless compression of regions of interest in hyperspectral images with no-data regions," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 1, pp. 173–182, 2016.

[7] H. Tanaka and A. Leon-Garcia, "Efficient run-length encodings," *IEEE Transactions on Information Theory*, vol. 28, no. 6, pp. 880–890, Nov 1982.

[8] G. G. Langdon, Jr. and J. Rissanen, "Compression of black-white images with arithmetic coding," *IEEE Transactions on Communications*, vol. 29, no. 6, pp. 858–867, June 1981.

[9] S. Zahir and M. Naqvi, "A new rectangular partitioning based lossless binary image compression scheme," in *Canadian Conference on Electrical and Computer Engineering (CCECE'05)*, May 2005, pp. 281–285.

[10] L. Zhou and S. Zahir, "A new efficient algorithm for lossless binary image compression," in *Canadian Conference on Electrical and Computer Engineering (CCECE'06)*, May 2006, pp. 1427–1431.

[11] K. Sayood, *Lossless Compression Handbook*. Academic Press, 2002.

[12] "ISO/IEC 14492, Information technology - lossy/lossless coding of bilevel images," 2001.

[13] R. Raguram, M. Marcellin, and A. Bilgin, "Improved resolution scalability for bilevel image data in JPEG 2000," *IEEE Transactions on Image Processing*, vol. 18, no. 4, pp. 774–782, April 2009.

[14] A. L. Liaghati, H. Shen, and W. D. Pan, "An efficient method for lossless compression of bi-level ROI maps of hyperspectral images," in *Proc. 2016 IEEE Aerospace Conference*, March 2016.

[15] M. Y. Jaisimha, H. Potlapalli, H. Barad, A. B. Martinez, M. C. Lohrenz, J. Ryan, and J. Pollard, "Data compression techniques for maps," in *Proceedings of IEEE Energy and Information Technologies in the Southeast*, April 1989, pp. 878–883.

[16] M. Wan, Y. Ding, and Y. Pan, "Test data compression using extended frequency-directed run length code based on compatibility," *ELECTRONICS LETTERS*, vol. 46, no. 6, pp. 404–406, 2010.

[17] Y. Yu, Z. Yang, and X. Peng, "Test data compression based on variable prefix dual-run-length code," *IEEE International Instrumentation and Measurement Technology*, pp. 2537–2542, 2012.

[18] W. Zhan and A. El-Maleh, "A new scheme of test data compression based on equal-run-length coding (ERLC)," *Integration, the VLSI Journal*, vol. 45, no. 1, pp. 91 – 98, 2012.

[19] C. Kalamani and K. Paramasivam, "A combined compatible block coding and run length coding techniques for test data compression," *World Applied Sciences Journal*, vol. 32, no. 11, pp. 2229–2233, 2014.

[20] B. Ye, Q. Zhao, D. Zhou, X. Wang, and M. Luo, "Test data compression using alternating variable run-length code," *Integration, the VLSI Journal*, vol. 44, no. 2, pp. 103 – 110, 2011.

[21] A. S. Arif, S. Mansor, H. A. Karim, and R. Logeswaran, "Lossless compression of fluoroscopy medical images using correlation and the combination of run-length and Huffman coding," *IEEE-EMBS Conference on Biomedical Engineering and Sciences*, pp. 759–762, 2012.

[22] H. Elsayed, "Burrows-wheeler transform and combination of move-to-front coding and run length encoding for lossless audio coding," *International Conference on Computer Engineering and Systems*, pp. 354–359, 2014.

[23] K. Han, Z. Deng, and H. Gong, "Double-run-length compression of test vectors scheme for variable-length to variable-length," *International Conference on Computing and Convergence Technology*, pp. 835–839, 2012.

[24] M. J. P. Nagabhushan and B. B. Chaudhuri, "Entropy computations of document images in run-length compressed domain," *International Conference on Signal and Image Processing*, pp. 287–291, 2014.

[25] R. Sahoo, S. Roy, and S. S. Chaudhuri, "Haar wavelet transform image compression using run length encoding," *International Conference on Communication and Signal Processing*, pp. 71–75, 2014.

[26] M. Arif and R. S. Anand, "Run length encoding for speech data compression," *International Conference on Computational Intelligence and Computing Research*, pp. 1–5, 2012.

[27] H. Li, H. Zheng, and C. Han, "Adaptive run-length encoding circuit based on cascaded structure for target region data extraction of remote sensing image," *International Conference on Integrated Circuits and Microsystems*, pp. 20–27, 2016.

[28] K. Zhang, W. Hao, and Z. Xu, "A run-length based algorithm for feature extraction from multi-target image," *International Congress on Image and Signal Processing*, pp. 397–400, 2012.

[29] M. Kouchi and O. Watanabe, "An image quality index using zero-run-length of dct coefficients for jpeg xt images," *IEEE 5th Global Conference on Consumer Electronics*, pp. 1–5, 2016.

[30] W.-L. Tai, C.-S. Chan, and C.-A. Chu, "Apply run-length encoding on pixel differences to do image hiding," *International Conference on Information, Communications and Signal Processings*, pp. 1–5, 2013.

[31] C. Tu, J. Liang, and T. D. Tran, "Adaptive runlength coding," *IEEE Signal Processing Letters*, vol. 10, no. 3, pp. 61–64, 2003.

[32] S. Aviran, P. H. Siegel, and J. K. Wolf, "Optimal parsing trees for run-length coding of biased data," *IEEE Transactions on Information Theory*, vol. 54, no. 2, pp. 841–849, Feb 2008.

[33] A. Steadman and I. Fair, "Variable-length constrained sequence codes," *IEEE Communications Letters*, vol. 17, no. 1, pp. 139–142, January 2013.

[34] H. Malvar, "Adaptive run-length/Golomb-Rice encoding of quantized generalized gaussian sources with unknown statistics," in *Proc. of Data Compression Conference (DCC)*, March 2006, pp. 23–32.

[35] J. Capon, "A probabilistic model for run-length coding of pictures," *IRE Transactions on Information Theory*, vol. 5, no. 4, pp. 157–163, December 1959.