Ternary Data Categorization for Evaluating the Impact of Data Transforms on Image Compressibility

Andrii SAMOFALOV, Ladislav POLAK

Dept. of Radio Electronics, Brno University of Technology, Technicka 12, 616 00 Brno, Czech Republic

262759@vut.cz

Submitted November 28, 2025 / Accepted February 24, 2026 / Online first March 12, 2026

Abstract. *In our increasingly digital world, data compression remains an important research topic. One way to improve the compression ratio is to use data transform techniques to increase further compressibility of input data for specific compression algorithms. This paper introduces a novel ternary data categorization in order to evaluate the impact of data transform techniques on input data. The categorization is described and explained in detail. Then, three transforms used for testing are described: Burrows-Wheeler and Move-to-Front transforms (BWT and MTF), as well as shifting summation. These transforms are applied to the eight true color images from the* Kodak *and* Kaggle *image sets. The findings derived from the ternary graphs and statistical measures indicate that combining BWT and MTF transforms yields the best results for this test on the selected image data. The shifting summation approach opens up possibilities for further research, particularly in searching for data patterns.*

Keywords

Ternary categorization, ternary diagram, image compression, data transform, transform evaluation

1. Introduction

The world around us is rapidly becoming digital, and the importance of data compression is growing. As the number of electronic devices increases, so does the amount of data stored and transmitted through networks. The reduction in the space these data occupy is anticipated to provide businesses with numerous benefits, such as cutting infrastructure costs and gaining a competitive advantage by providing users with higher-quality digital products at lower costs.

While lossless data compression techniques may provide excellent results for many types of data, their use can often be limited by not sufficiently compressing the input data in comparison with lossy approaches. This is particularly important in image compression, as visual content is growing every day, especially due to social network users. For example, the size reduction benefits of applying lossy techniques to image data can greatly outnumber the loss in the visual quality to a certain extent. Paper [1] provides an overview of existing lossless and lossy compression methods alongside performance metrics for their evaluation. Alternatively, Deep Learning (DL) approaches for data compression provide an interesting substitute for conventional lossy compression techniques. Work [2] provides an overview and discussion of commonly used Machine Learning (ML) architectures utilized in lossy image compression. After that, pros and cons of each image compression frameworks are presented and several research gaps are discussed as well. While DL-based image compression approaches demonstrate a promising alternative to the conventional lossy compression methods, lossless compression techniques can still be applied to many cases (e.g., microscopy images). As demonstrated in [3], DL-based image compression techniques may outperform conventional algorithms. However, lossless compression approaches can still be beneficial for compressing latent representations obtained using novel ML models. Such an observation reinforces the idea that lossless compression algorithms will remain useful for various compression tasks despite the development of lossy DL approaches.

One way to increase the compression ratio of lossless techniques is the development of new lossless compression approaches. The other is to transform the initial input data by making more runs of similar symbols or creating specific patterns that can be effectively utilized later by compression methods. One of the most known data transforms is Burrows-Wheeler Transform (BWT) [4]. This work provides the initial description of this transform alongside the discussion of its implementation. Next, work [5] performs an experimental comparison of different transform techniques. The authors discovered that other transform algorithms are not as effective without BWT as they are when combined together. The application of BWT to improve the compression of information in deoxyribonucleic acid (DNA) sequences is studied in [6] and [7]. The first study employs BWT to search for tandem repeats in DNA sequences [6], explaining that the need for this research is based on the increases in the sizes in biological databases. The next research [7] focuses on the compression of repeating patterns in DNA sequences. The authors of [8]

DOI: 10.13164/re.2026.0164

apply Hilbert curves alongside selected lossless compression techniques to compress medical images. The results of this research also suggest that pre-processing operations on the input data could lead to the best compression results. These works underscore the importance of lossless compression in medical studies, given that the volume of research and data in this field only continues to grow.

While some researchers apply BWT to existing data to achieve better results in compression, other researchers seek ways to improve this technique. The authors of work [9] implement a weighted coding approach on input data after BWT. This procedure can increase savings for files with skewed data distributions. The author of the work [10] present the new tunneling technique, which can be applied to any BWT-based compression methods reducing the output sequences sizes. As BWT is often followed by Move-to-Front (MTF) transform, the work [11] discusses various subsequent stages of BWT such as inversion frequencies and others, which can be useful in selecting alternatives to the MTF transform in image data processing.

Paper [12] explores the use of MTF transform as a part of a pipeline for lossless image compression. It compares linearization methods (row-major and Hilbert curve) and shows that combining image locality and MTF improves compression performance. Many of compression techniques combine the use of BWT and MTF transforms. Authors of [13] introduce Move with Interleaving (MwI) transform, a variant of transforms, that includes MTF among baselines. It compares classical MTF, inversion frequencies, combining BWT and MTF, and the new MwI on raster images. It is shown that MwI with Hilbert ordering can reduce entropy similarly or more than BWT and MTF together. The study [14] compares classical MTF transform with a variant called Move-to-Front-or-Middle (MFM) transform, analyzing performance in compression rates on Calgary and Canterbury datasets.

Researchers in [15] investigate a combination of BWT, MTF and Huffman coding for text data. They show under what conditions the full pipeline becomes more efficient than simpler encoding methods. Authors of [16] perform theoretical analysis of MTF transform in list accessing problems, including performance on sequences without locality of reference. Work [17] compares BWT, Run-Length encoding (RLE), Lempel-Ziv-Welch and Huffman algorithms alongside other techniques, using a custom text dataset on compression ratio benchmarks.

To summarize, the field of data compression and more specifically data transformation remains of high interest for researchers. Contemporary literature presents many new looks and variations of such transforms. Lossy data compression techniques often outperform lossless approaches in terms of compression ratio, particularly for image compression. However, preserving original image quality is the biggest advantage of lossless data compression. This can be a deciding factor in certain situations (e.g., medical imagery). Many authors use conventional approaches to determine the impact of transforms, such as compression ratio [17] and entropy [13]. The other two papers evaluate the performance of different BWT variations on biological-based data. The author of the work [18] measures BWT construction and search performance using metrics such as average run length, average number of Central Processing Unit (CPU) threads, Random-Access Memory (RAM) usage, and kilobases processed per CPU second. Cenzato et al. [19] analyze dataset variability, Hamming distance, and average sequence length (similar to the previous paper). While these explored evaluation methodologies can be considered conventional, developing a novel approach to select the best variant or combination of transforms would be highly useful. Such a categorization would provide an alternative basis for the objective analysis beyond conventional approaches.

1.1 Contributions of the Paper

The main contributions of the paper are as follows:

1. The novel ternary categorization of data is presented. This approach is used for evaluation of the impact of data transform techniques on the input data to determine whether such transforms improve potential further data compressibility. The category patterns of input data are easily observed with the usage of ternary graphs by plotting the outputs of the described categorization.
2. The boundaries between categories and the process of selecting the optimal sequences from multiple variants, such as initial and numerous after-transform(s) sequences, are explored. The calculation of lowest prospective length, which is used in the paper for the purpose of selecting the optimal sequence, is presented.
3. Rigorous testing is performed on eight selected images from the open image sets, applying the principles presented in this paper. The results obtained are presented on ternary graphs and described in detail, followed by the combined table of conventional statistical evaluation methods.

1.2 Organization of the Paper

The rest of this paper is organized as follows: Section 2 explains the ternary categorization of sequence patterns in the input data. Data transform techniques that will be utilized during the tests are briefly described in Sec. 3. Section 4 evaluates and discusses the results obtained for various test images from the open image sets, combining the illustrated principles from previous sections. Finally, the paper is concluded in Sec. 5.

2. Data Analysis Tools

One thing is to perform a transform on the input data, the other thing is to determine whether this transform was actually helpful for the compression methods to increase the reduction of the input file size. To make such a novel cat-

egorization, the patterns of the data in the initial sequence are classified into three categories based on their occurrence in the input data and not taking into account their adjacency patterns (symbols are considered values that a one byte unit consisting of eight bits holds):

- **Plain** - this is the ideal situation when as many symbols in the initial sequence as possible are the same. This facilitates the usage of RLE, which achieves superior results among lossless data compression methods. In order for a value to be classified under the plain category, it must be the most commonly occurring symbol in the dataset, a concept known in statistics as the mode.
- **Stepped** - in this case the initial data have sequences of similar symbols, which also exhibit signs of the plain category. However, the main difference from the plain category is that there are multiple such groups of values with each according length exceeding the set threshold separating them from the chaotic category.
- **Chaotic** - this category encompasses all values that do not fall into the **Plain** and **Stepped** categories. In this case, all values in the input data are different or below a certain length threshold preventing them from turning into the **Stepped** category. This length threshold can be set either as some selected value or calculated dynamically based on the distribution of symbols in the input sequence. From the categorization point of view, these values do not form any patterns and require all eight bits to be encoded.

The presented categorization assigns certain number of bits needed to encode each symbol in the initial sequence. Specifically, the plain category uses zero bits per symbol, since RLE technically uses zero bits per symbol after the repeated symbol has been provided. The chaotic category requires eight bits per symbol because these symbols fall outside the defined groups and must be encoded as they are. Finally, stepped category values are encoded using from one to seven bits, allowing for up to 128 distinct stepped groups. It is important to ensure that the number of bits selected to encode symbols of the stepped category can cover all distinct values belonging to this category.

This classification does not take into account the number of bytes needed to encode the service information about each category positions, lengths, etc. Therefore, the described representation draws focus to the potential length that these symbols could occupy based on the length and type of the category. In case there is more than one symbol group with the highest length, then all of them are considered to be from the stepped category and the plain one is left empty. This approach allows to put the three described categories in opposite extreme corners, when the ideal input sequence consists of only one symbol type, the chaotic sequence contains all values that are different, and the stepped category has symbols with equally distributed lengths. Finally, the sum of the symbols from all the categories equals the length of the input data sequence.

Since the ternary categorization is task-agnostic, it can be applied to a broader range of data than presented in this paper. The categorization operates with provided byte values on an as-is basis, therefore future research could apply it to other data types. Since the ternary categorization is a statistical tool, it uses the provided input data directly. Therefore, any changes to the input data due to noise are processed as is. Outliers are handled similarly, meaning they will likely remain in the chaotic group rather than moving to the stepped group because there are not enough such similar symbols. Two problems may arise within the presented ternary categorization. The first worst-case scenario occurs when the selected bit length for the stepped category does not cover the full range of possible value ranges in the provided data. Therefore, ensuring the correct bit length for the stepped category symbols is necessary to prevent this error. Another option is to use prefix codes to encode the stepped symbols. In this case, the calculation of the lowest prospective length should be adjusted accordingly. The second issue is choosing threshold values that determine whether a symbol falls into the chaotic or stepped categories. If the threshold is set too high, many symbols will remain in the chaotic category. Conversely, if the threshold is too low, a very small number of distinct symbols (perhaps only two) may form a separate stepped category, even in data with thousands of values. Incorrect threshold selection skews the distribution of symbols among the ternary groups in the input data.

Following from this specification, the boundaries between the stepped and chaotic categories separated by equilibrium points could be calculated. In these points the length of the chaotic series equals the length of the stepped series of a certain bit number per value.

$$S_{\text{eq}} = \frac{C_l}{S_l + C_l} \quad \text{and} \quad C_{\text{eq}} = \frac{S_l}{S_l + C_l}. \tag{1}$$

The percent ratio of how much space is occupied by each category is calculated by dividing the number of bits needed to encode a symbol of the opposite category by the sum of bits from both the chaotic and stepped categories. As shown in (1), the stepped and chaotic equilibrium boundaries are represented by S_{eq} and C_{eq}, respectively, while the number of bits needed to encode one symbol of the corresponding category is represented by S_l and C_l.

	Numerical ratio		Percent ratio	
	Stepped	**Chaotic**	**Stepped**	**Chaotic**
1 bit	8	1	88.9%	11.1%
2 bits	4	1	80%	20%
3 bits	8	3	72.7%	27.3%
4 bits	2	1	66.7%	33.3%
5 bits	8	5	61.5%	38.5%
6 bits	4	3	57.1%	42.9%
7 bits	8	7	53.3%	46.7%

Tab. 1. Calculated equilibrium points.

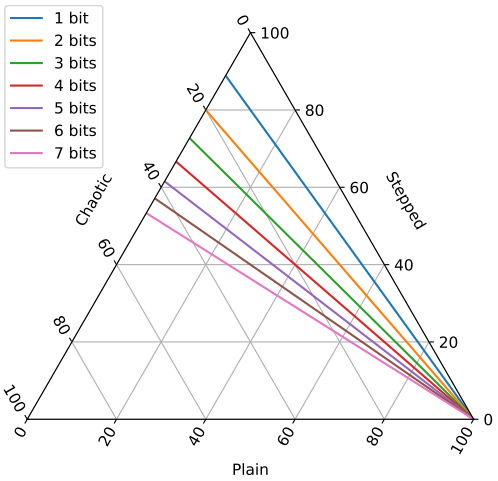


Fig. 1. Equilibrium points of the stepped and chaotic categories (axes values are in %).

The results are presented in Tab. 1 and Fig. 1. Categorizing data into three distinct groups, with the cumulative length sum of one hundred percent, makes ternary diagrams an ideal approach for visualizing the distribution of the input data values. Moreover, ternary diagrams provide an easy way to observe category trends and distributions in the data. All ternary graphs in this paper are made using the Mpltern[1] and Matplotlib [20] Python libraries. As we can see, for example, if four bits are used to encode values of the stepped category then two stepped symbols have the same length as one chaotic symbol. As we assume the length of each symbol in plain category to be zero, then this category has no impact on the ratio between the chaotic and stepped categories. Consequently, increase in numbers of the stepped symbols versus chaotic results in lower potential length needed to encode input data. Conversely, the larger number of chaotic symbols from the equilibrium line of a certain type indicate the loss in potential encoding savings. However, it is also needed to consider the impact of the plain sequence as it can soften the impact of going under the equilibrium line into the chaotic category, when considering the overall reduction in the obtained sequence size. In other words, going towards symbol increases in the stepped and, ultimately, the plain categories is perceived as a "win-win" situation, otherwise an additional analysis should be performed.

As one of the means of such an analysis a formula could be developed which assigns a certain score to the input data based on its symbol content and, possibly, patterns. After that, these formula outputs could be further compared to decide which transform or combination of transforms, if anything, to use. Meanwhile, an approach calculating the lowest prospective length will be employed in this paper. In this approach, the plain category symbols are ignored (as they require zero bits to be encoded following from this categorization, there is no need to multiply their length by zero), the length of the chaotic series is multiplied by eight (space that one chaotic symbol occupies) and then added to the length

[1]https://doi.org/10.5281/zenodo.3528354

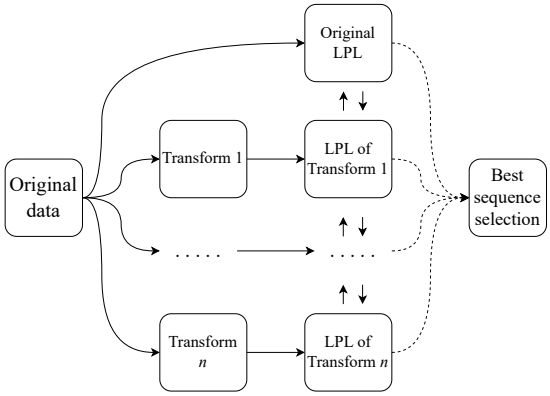


Fig. 2. Selection process of the best transform(s) based on LPL.

of the stepped series multiplied by the number of bits needed to encode one stepped symbol. After that, choosing the variant that provides the lowest value of the prospective length should be the goal within this categorization.

The process of selecting the best data representation based on the lowest prospective length (LPL) is shown in Fig. 2. As can be seen, all transformed sequences, including the initial, obtain their respective lowest prospective length scores. After that, the sequence with the minimal score is considered to provide the possibility of highest compression ratio for the further compression algorithms.

3. Description of Transforms

To demonstrate the transforms that will be used for the tests in this paper, a specific seven letter word *"Success"* was selected. It contains three letters of *S*, two letters of *C*, and two other distinct letters *U* and *E*. This distribution of letters provide a clear plain category, one run of the stepped category with a threshold of 2, and two distinct chaotic symbols.

The first transform to be discussed is the BWT. To perform this transform, it is needed to find all cyclic permutations of the input data, then sort them lexicographically, and take the sequence consisting of the last characters of the sorted input data permutations. This sequence is the transformation itself. If the input data contain many identical repeating characters, then with a high probability, the transformed string will contain these repeating characters close to each other.

In the BWT example, shown in Fig. 3, the test word does not contain neither the character representing the start of the sequence nor the character used for the sequence end. Because of this, the use of such characters and their selected position in the alphabetical sort could alter the output sequence of BWT, however, the underlying idea of this transform remains the same regardless.

The next transform to be discussed is the Move-to-Front transform [21], which is commonly used in conjunction with the BWT. It operates by maintaining a dynamic alphabetically ordered list of symbols and encoding each input symbol

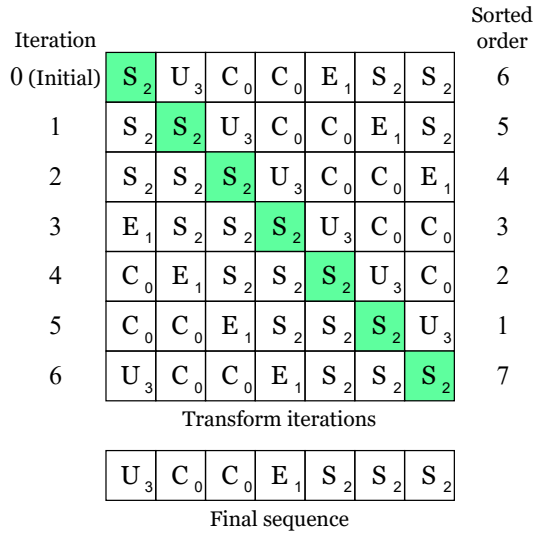


Fig. 3. An example of BWT on the test word *Success*.

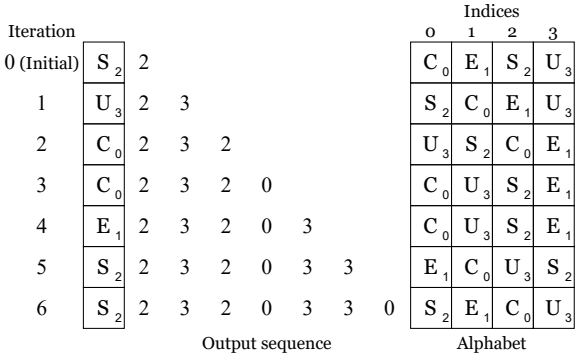


Fig. 4. An example of MTF transform on the test word *Success*.

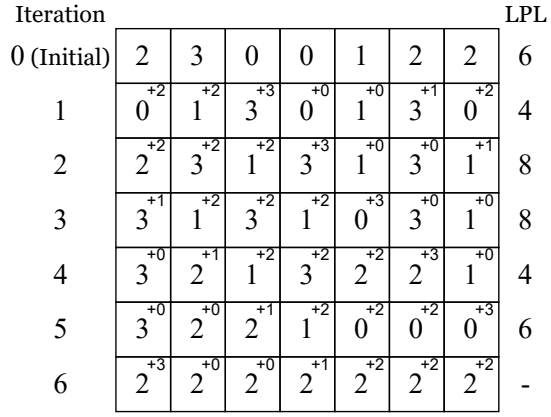


Fig. 5. An example of the shifting summation on the test sequence.

according to its position in this dynamic list. After encoding each subsequent symbol from the input data, the dynamic list is updated by pushing the used character to the front of the list. This transform is particularly effective when the input contains repeated patterns or when it is used after the BWT, which tends to cluster similar characters together.

The output sequence of numbers from the MTF transform on the test word is shown in Fig. 4. Runs of similar symbols produce sequences of zeros. Spikes in the number sequences appear when a less common symbol is encountered.

The final transform that will be used in the testing is the shifting summation. It is performed by shifting the initial sequence one position further in each iteration and adding its values to the previous ones. These resulting values are calculated in the modular arithmetic way, so that the remainders of this sum are given after the division by the alphabet size. An example of the shifting summation is shown in Fig. 5. In this example the letters of the word *"Success"* are encoded with their initial alphabet positions, the alphabet size equals four, the threshold of the stepped category is two, the indices in the top right corner of the cells represent the shift of the initial sequence, and "LPL" corresponds to the lowest prospective length. To calculate the lowest prospective length in each iteration, the symbol size of the stepped category is considered to be one bit and the chaotic category symbols take two bits. As it was mentioned earlier, if there is more than one symbol group with the highest length, then all of them are considered to be from the stepped category and the plain one is left empty. Even though the analysis in this paper is performed based on the lowest prospective length, we can see that the symbols in each iteration exhibit certain patterns, which could be more preferable by certain compression methods.

As can be seen, the total number of unique iterations equal the length of the initial sequence minus two because on the next iteration all symbols will have the same value of the sum of all elements mod the alphabet size. There is no need to calculate the lowest prospective length for the iteration when the sum is obtained because it is not possible to reconstruct the initial sequence from such data. The same process of summation can be continued after the iteration where the sum of all symbols was obtained, which will result in the gradual "elevation" of all values by the cumulative remainders of the sum. However, this process of summations after the sum might require further testing in accordance with the desired outcome based on the evaluation method or specific pattern seeking of selected compression approaches.

4. Results and Discussion

For the research purposes, five pictures from the *Kodak* image database[2] and three images from the the dataset available on Kaggle[3] were selected. The selected images have high variance of visual content, such as monotonous and vibrant color schemes as well as distinct depicted objects. Kodak images have dimensions of 768×512, with four of them having the bigger number as their widths apart from one (*Kodak-10*). Kaggle images have dimensions of 275×183, 225×225 and 177×284.

[2] https://r0k.us/graphics/kodak/

[3] https://www.kaggle.com/datasets/pavansanagapati/images-dataset

Such a number of selected images for the tests may limit the generalization to other image and data types. However, the discussed principles of the ternary categorization, along with the variety of images from the *Kodak* and *Kaggle* image databases provide a sound basis for further research. The program for ternary categorization of data was written in Python and run on a PC with 12th Gen Intel ® Core ™ i3-12100 CPU running at 3.30 GHz and 16 GB of installed RAM. Before the tests, all five selected Kodak images were resized to the size of 150×100 and 100×150 (*Kodak-10*) to speed up the calculations of transforms, maintaining the initial aspect ratio of 3 : 2 and the bit depth of 24. Kaggle images were also resized to the size of 150×100 (*horse-113*) and 100×150, maintaining the bit depth of 24. All the resized images were saved in the .png format. Next, the images were read and flattened into a single 45000-element array. The Numpy function *flatten* was called without any arguments provided, resulting in the default flattening of row-major (C-style) order. Then, the transforms were performed on the arrays in 1024-byte chunks. This produced 44 groups, which will be represented as points on the subsequent ternary graphs. Each axis on these graphs has a limit of 100, representing the percentage length of each category group in the data. The source code is available online[4].

The minimum length threshold for a symbol group to be considered the stepped category is 16. This means that in a segment of size 1024, there could be as many as 64 groups if the symbols are distributed evenly. Consequently, to incorporate all the values in this case, the size of each symbol in the stepped category will be set to six bits. These rules ensure the consistency of the testing and display of the results in the subsequent ternary graphs. The choice of the minimum length threshold and the size of the stepped group symbol could be calculated separately for each segment and based on the demands of the compression method of choice. However, in this paper the threshold of 16 is the middle ground between the threshold of eight (stepped category symbols need 128 values to cover all possible values in 1024-byte chunks and take seven bits to do so) and the threshold of 32 (many symbols may remain in the chaotic category without forming the stepped groups). If there is more than one chunk with the lowest prospective length, the first one is considered the best. The alphabet size for all segments in the shifting summation is set to 256. The range of values in a segment is calculated as the difference between the min and max values. In the BWT and MTF transforms combination, BWT was applied first, followed by MTF.

The results for the resized image *Kodak-2* are shown in Fig. 6. As can be seen, the initial image category distribution has a low percentage of the plain category and a certain number of chunks above the 6-bit equilibrium line. This could be explained by the fact that the initial image, *Kodak-2*, has one predominant color. The shifting summation demonstrates patterns similar to those of the initial data distribution. However, a significant number of chunks reach

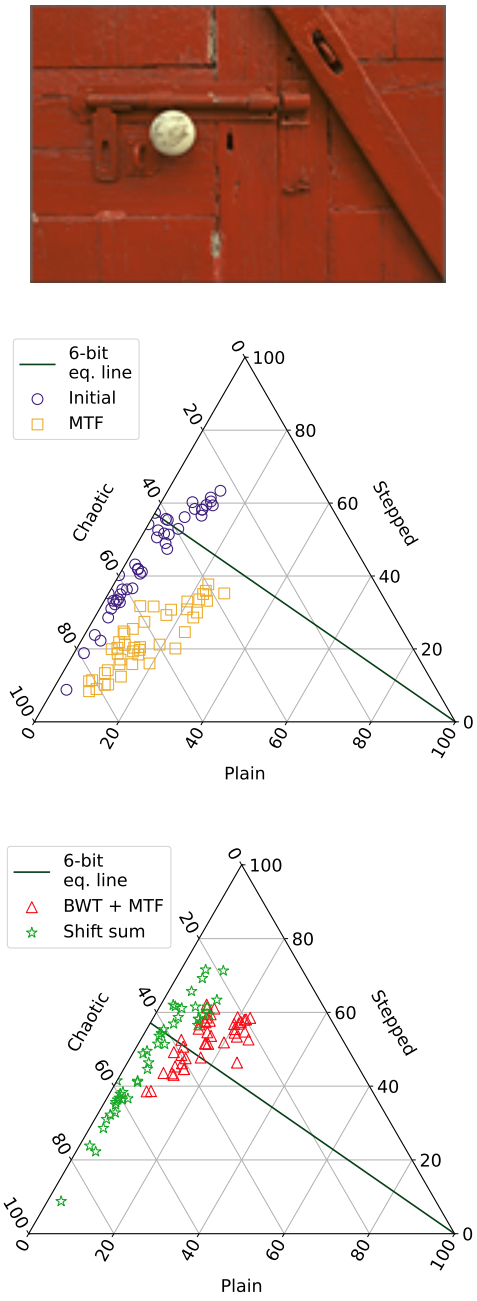
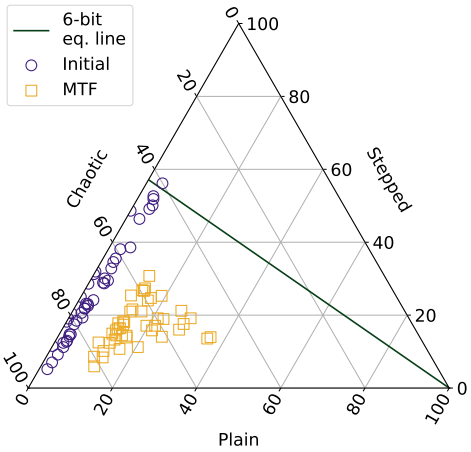


Fig. 6. Resized image *Kodak-2* and its ternary graphs: (top) Resized image *Kodak-2*; (middle) Initial and after MTF; (bottom) After BWT + MTF and shifting summation. All axes values are in %.

their lowest prospective length in the second iterations, indicating differences from the initial sequences. Next, the single MTF results in the clustering of the chunks around 40% and 80% of the chaotic category. Although all the chunks move below the equilibrium line, a higher percentage of the plain category symbols is obtained. Ultimately, the combination of BWT + MTF produces the best results, showing an overall trend of a greater presence of the stepped category around or above the equilibrium line while keeping the percentage of the plain category similar to that of the single MTF.

[4]https://drive.google.com/drive/folders/1a8-KXc7VbshGRxXd6TukEQ-2WCxWS5s9?usp=sharing



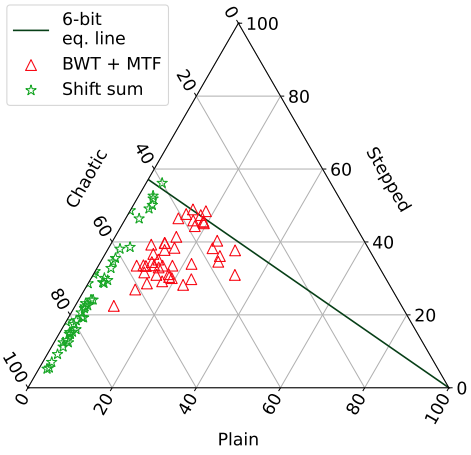
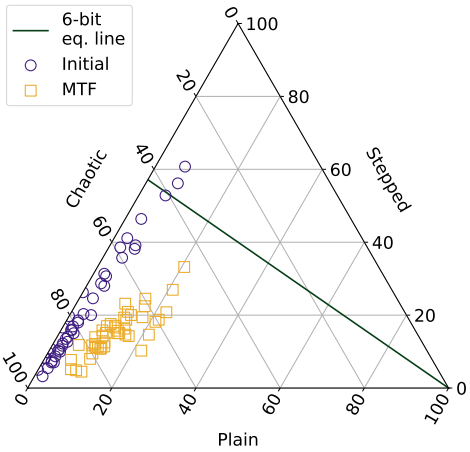


Fig. 7. Resized image *Kodak-3* and its ternary graphs: (top) Resized image *Kodak-3*; (middle) Initial and after MTF; (bottom) After BWT + MTF and shifting summation. All axes values are in %.



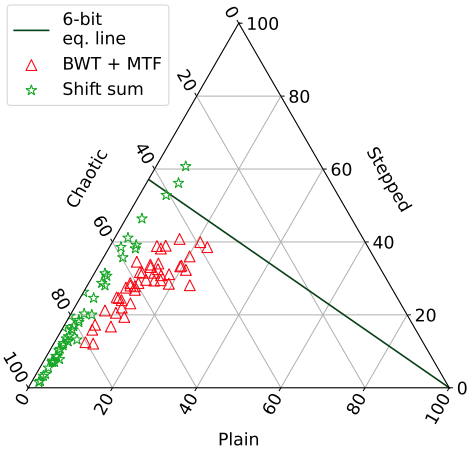


Fig. 8. Resized image *Kodak-7* and its ternary graphs: (top) Resized image *Kodak-7*; (middle) Initial and after MTF; (bottom) After BWT + MTF and shifting summation. All axes values are in %.

The next resized image, *Kodak-3*, also demonstrates the higher efficiency of the BWT and MTF transforms, whether used separately or together, as shown in Fig. 7. The initial and shifting summation distributions are once again very similar, with a stretched trend along approximately zero to five percent range of the plain category. The predominant part of the shifting summation iterations could not improve upon the initial data representation. The single MTF groups chunks close to each other, reducing the overall spread of the chaotic category and increasing the presence of the plain category. Subsequently, BWT + MTF shifts the general shape of this chunk group toward the higher percentages of the stepped category, while maintaining a somewhat similar percentage of the plain category.

The following image, *Kodak-7*, shows patterns similar to those in previous images, with the combination of BWT and MTF transforms being on top, as captured in Fig. 8. This combination once again provides the highest increases in the stepped and plain categories, in contrast to the single MTF, which only increases the plain category. Interestingly, two chunks of the initial data are above the equilibrium line. In the case of the shifting summation, only some iterations are better than the initial distributions named iteration 0, producing once again similar shapes to the initial data.

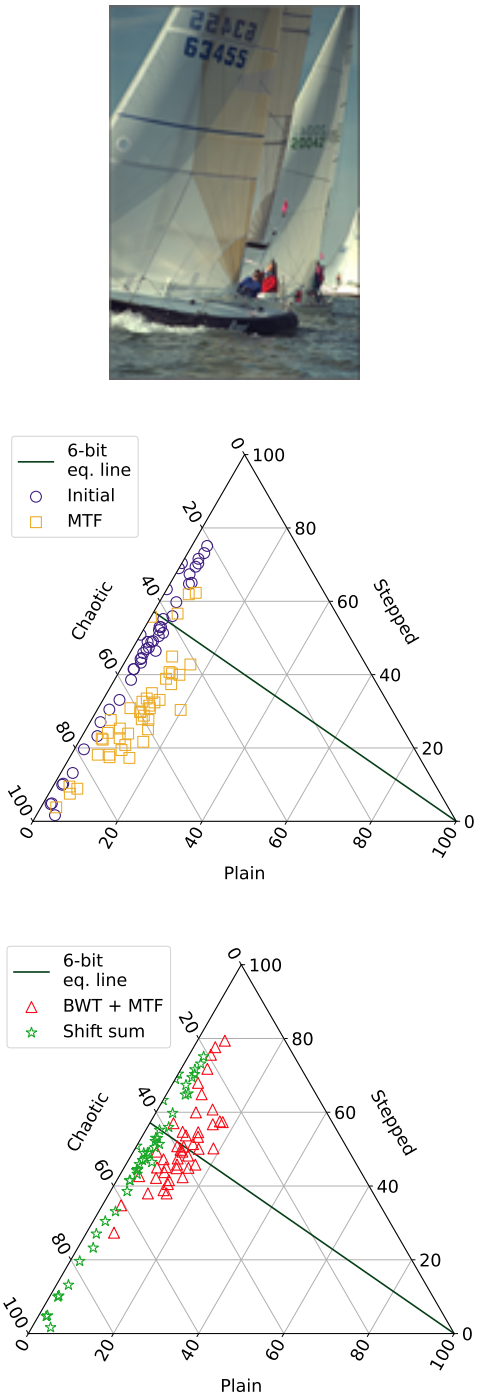


Fig. 9. Resized image *Kodak-10* and its ternary graphs: (top) Resized image *Kodak-10*; (middle) Initial and after MTF; (bottom) After BWT + MTF and shifting summation. All axes values are in %.

The results for the image *Kodak-10* in Fig. 9 indicate that none of the iterations of the shifting summation can find instances with a prospective length lower than that of the initial sequences. In this case, both data distributions are similar and distributed around five percent of the plain category and the range from zero to 78% of the stepped category. Additionally, less than a half of the chunks are above the 6-bit equilibrium line. Regarding the single MTF, it fails to increase the plain category for all chunks. However, this trend is visible for the higher percentage of them. Finally, the combination of BWT + MTF repeats its success from the previous images by moving many chunks close to or above the equilibrium line, with some scoring around 80% of the stepped category.

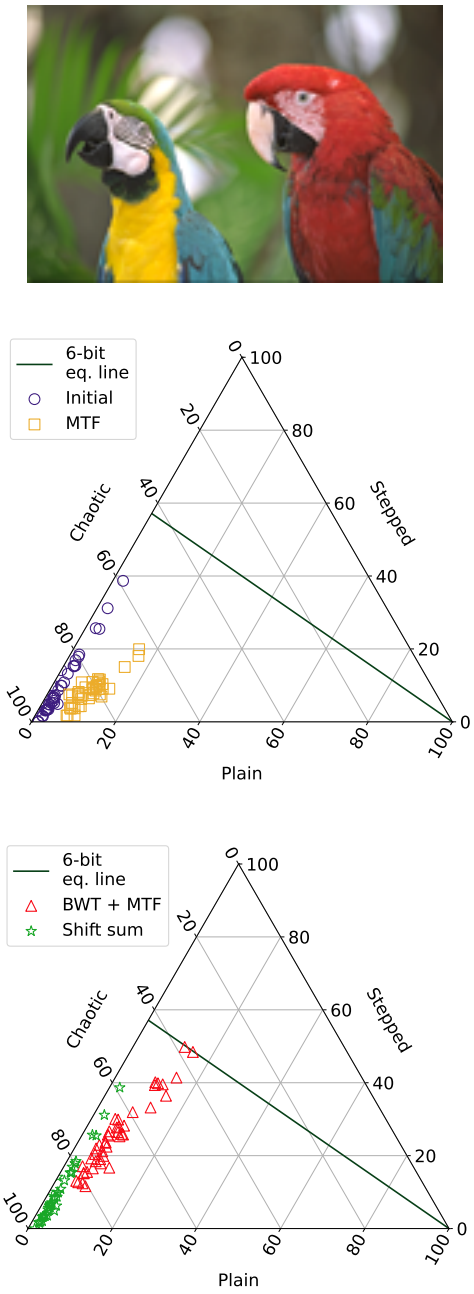


Fig. 10. Resized image *Kodak-23* and its ternary graphs: (top) Resized image *Kodak-23*; (middle) Initial and after MTF; (bottom) After BWT + MTF and shifting summation. All axes values are in %.

Finally, the results for the image *Kodak-23* are shown in Fig. 10. Once again, many of the lowest prospective lengths of the chunks in the shifting summation are the same as in the initial sequences. The initial sequence distributions maintain the linear trend of the 3% of the plain category with most of

		Mean	Std. dev.	Range
Kodak-2	Initial	76.82	52.62	255
	MTF	40.82	42.26	198
	BWT + MTF	23.52	39.12	198
	Shift. sum.	134.87	88.47	255
Kodak-3	Initial	96.60	45.68	255
	MTF	51.29	53.67	217
	BWT + MTF	34.98	48.92	217
	Shift. sum.	103.25	50.70	255
Kodak-7	Initial	104.61	40.32	255
	MTF	57.91	55.70	210
	BWT + MTF	39.70	47.00	210
	Shift. sum.	110.87	47.20	255
Kodak-10	Initial	119.65	37.19	249
	MTF	41.06	44.84	228
	BWT + MTF	24.95	31.73	228
	Shift. sum.	119.65	37.19	249
Kodak-23	Initial	102.38	56.98	255
	MTF	74.70	65.79	223
	BWT + MTF	55.21	58.92	223
	Shift. sum.	116.37	65.83	255

Tab. 2. Statistics for the tested *Kodak* images.

the chunks being above the 80% of the chaotic category. The single MTF yet again groups the chunks around 10% of the plain category and the combination of BWT + MTF brings them closer to the equilibrium line with some of them almost reaching it.

The combined statistical results for the selected images are presented in Tab. 2. As the data show, the shifting summation has the highest mean value of all transforms, or a similar value in one case. The standard deviation of the shifting summation is also higher throughout all images. The BWT + MTF combination produces the lowest mean and standard deviation among the three transforms, which supports the clustering seen in the ternary graphs. Interestingly, the standard deviation of the shifting summation is similar to the one of BWT and MTF on the image *Kodak-7*. However, it is around 18% higher than the initial data. The standard deviation of the single MTF for four images, excluding *Kodak-2*, is higher than that of the initial data.

The ternary category distributions for the resized image *horse-113* are shown in Fig. 11. Most chunks from the initial image and after the MTF transform are clustered around 90% of the chaotic and below 20% of the stepped categories. Interestingly, five chunks from the initial image are placed above the equilibrium line, while the MTF transform moves them towards the higher percentages of the plain category. The BWT + MTF combination alongside the shifting summation shows similar trends to the previously discussed pair. Four chunks of the shifting summation achieved a higher proportion of the plain category than in the initial distribution. Three chunks from the BWT + MTF combination reach about 80% of the plain category.

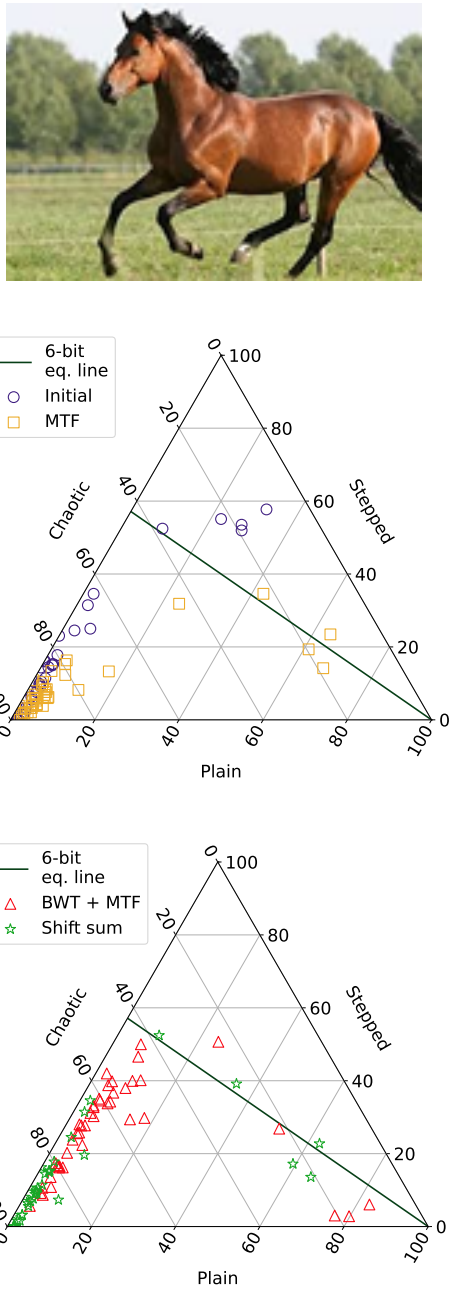
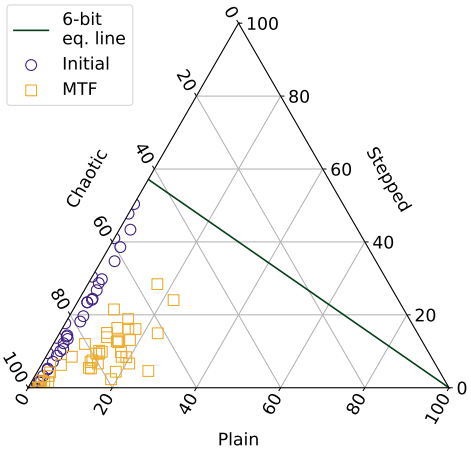


Fig. 11. Resized image *horse-113* and its ternary graphs: (top) Resized image *horse-113*; (middle) Initial and after MTF; (bottom) After BWT + MTF and shifting summation. All axes values are in %.

Figure 12 shows category distributions for the image *rider-107*. In this case, chunk distributions for the initial image and the shifting summation show minimal variations between them. The MTF transform groups most chunks around 10-20% of the plain category. The BWT + MTF combination then moves these chunks closer to the equilibrium line, with some crossing it. A small number of chunks, however, still remain within five percent of the plain category.



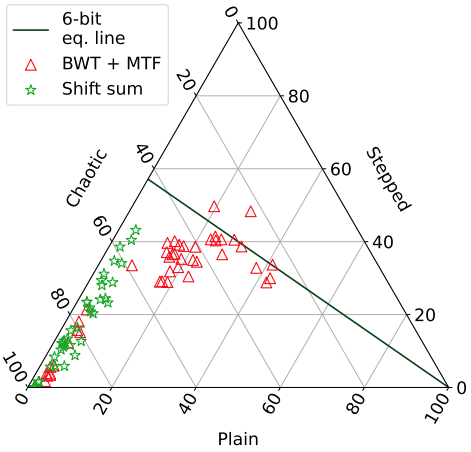
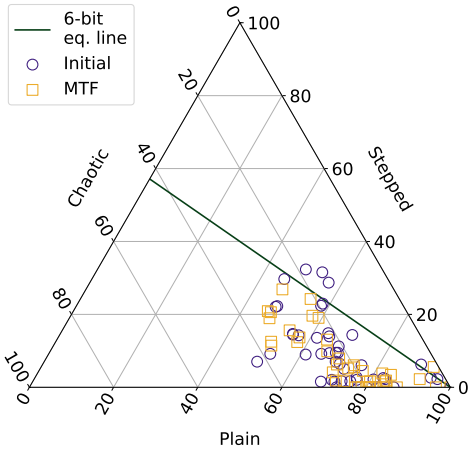


Fig. 12. Resized image *rider-107* and its ternary graphs: (top) Resized image *rider-107*; (middle) Initial and after MTF; (bottom) After BWT + MTF and shifting summation. All axes values are in %.



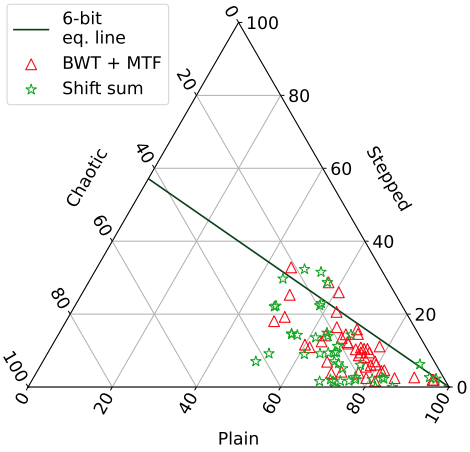


Fig. 13. Resized image *rider-111* and its ternary graphs: (top) Resized image *rider-111*; (middle) Initial and after MTF; (bottom) After BWT + MTF and shifting summation. All axes values are in %.

The last image, namely *rider-111*, shows the highest presence of the plain and stepped categories (see Fig. 13). Some chunks achieve a perfect 100% of the plain category, with others nearly reaching that level. The initial image and the shifting summation method share the same category distributions. The MTF transform produces a narrower spread of chunks than the initial distribution. The combination of BWT + MTF groups many chunks around 10% of the stepped and 80% of the plain categories. This can be explained by a single predominant background color and a small number of distinct colors, which pushes the chunks towards the higher percentages of the plain and stepped categories.

		Mean	Std. dev.	Range
horse-113	Initial	130.89	67.32	255
	MTF	71.22	62.05	233
	BWT + MTF	48.05	54.38	233
	Shift. sum.	128.30	66.93	255
rider-107	Initial	158.88	65.97	255
	MTF	71.15	61.73	219
	BWT + MTF	40.63	51.97	219
	Shift. sum.	149.44	65.88	255
rider-111	Initial	221.43	75.48	255
	MTF	10.02	26.06	174
	BWT + MTF	6.90	21.00	174
	Shift. sum.	221.43	75.48	255

Tab. 3. Statistics for the tested *Kaggle* images.

Table 3 presents statistics for the three images from the dataset available on *Kaggle*. The highest mean values occur for the initial images, while the lowest occur for the BWT + MTF combination. The mean for the shifting summation method is close to the initial distribution, and for *rider-111* it is identical. Likewise, the shifting summation standard deviations are similar to those of the initial images. Unsurprisingly, the lowest standard deviations are produced by the BWT + MTF combination. The mean values for the single MTF are closer to those for BWT + MTF than to the initial images, while its standard deviations are generally closer to the shifting summation values, except for image *rider-111* The image *rider-111* has the lowest range value among all eight tested images.

In summary, images with a lower number of distinct colors in general score higher in the lengths of the stepped and plain categories. The image with a single predominant color (see image *rider-111*) has the best category distribution results, meaning many chunks have a high presence of the plain and stepped categories. The combination of BTW and MTF transforms is the best transform in terms of the category distributions, as supported by the ternary graph trends and statistical measures. The shifting summation approach yields results similar to the initial sequence but is unable to surpass it on two test images, namely *Kodak-10* and *rider-111*. This can be explained by the fact that the alphabet size for the shifting summation approach is set to 256, while the number of different values in the chunks is lower, which can be seen from the range values in both variations of MTF transforms. However, as discussed in Sec. 3, many iterations of the shifting summation approach produce distinct patterns that could be useful for certain pattern-seeking data compression methods rather than the comparison of the calculated lowest prospective length. Therefore, further research in this direction may be necessary.

5. Conclusion

This paper introduces a new approach to categorizing the impact of data transformation techniques on input data. In contrast to the existing evaluation methodologies of data transforms, the presented ternary categorization provides a new forecasting look into possible encoding space savings, should any transform(s) be selected. This allows us to estimate which transform(s) should be applied to achieve the highest encoding savings through further compression algorithms, eliminating the need to apply these algorithms beforehand. Furthermore, a more flexible configuration of transform evaluation can be achieved by setting different threshold values and developing other formulas for optimal sequence selection. Ternary graphs are used to determine if and to what extent selected transforms improve the potential for further compression by comparing the category distribution trends of the initial and transformed data. The properties of the three distinct categories were thoroughly described, and the category boundaries were calculated alongside the selection of the optimal sequence. Next, the transforms used for testing were described: BWT, MTF, and the shifting summation. These techniques were tested on eight resized images from the *Kodak* and *Kaggle* image sets. The discussed ternary categorization principles, that were applied to the variety of images from the Kodak and Kaggle image datasets, provide a sound basis for further research. The results from the ternary graphs and statistical measures indicate that the combination of BWT and MTF transforms yielded the best results in this test setting. On the other hand, the shifting summation can be used to obtain certain patterns in the data, which may be more beneficial for certain compression approaches.

Acknowledgments

This work was supported by the BUT project no. FEKT-S-23-8191.

References

[1] ELAKKIYA, S., THIVYA, K. S. Comprehensive review on lossy and lossless compression techniques. *Journal of the Institution of Engineers (India) Series B*, 2021, vol. 103, no. 3, p. 1003–1012. DOI: 10.1007/s40031-021-00686-3

[2] JAMIL, S., PIRAN, J., RAHMAN, M., et al. Learning-driven lossy image compression: A comprehensive survey. *Engineering Applications of Artificial Intelligence*, 2023, vol. 123, p. 1–17. DOI: 10.1016/j.engappai.2023.106361

[3] ZHOU, Y., SOLLMAN, J., CHEN, J. Deep learning based image compression for microscopy images: An empirical study. *arXiv*, 2023, p. 1–15. DOI: 10.48550/arxiv.2311.01352

[4] BURROWS, M., WHEELER, D. J. *A block-sorting lossless data compression algorithm*. Technical Report 124. Palo Alto (CA, USA): Digital Equipment Corporation, 1994. [Online]. Available at: https://www.cs.jhu.edu/\~langmea/resources/burrows_wheeler.pdf

[5] DORRIGIV, R., LÓPEZ-ORTIZ, A., MUNRO, J. I. An application of self-organizing data structures to compression. *Lecture Notes in Computer Science*, 2009, vol. 5526, p. 137–148. DOI: 10.1007/978-3-642-02011-7_14

[6] POKRZYWA, R. Application of the Burrows-Wheeler transform for searching for tandem repeats in DNA sequences. *International Journal of Bioinformatics Research and Applications*, 2009, vol. 5, no. 4, p. 432-446. DOI: 10.1504/IJBRA.2009.027517

[7] ADJEROH, D., ZHANG, Y., MUKHERJEE, A., et al. DNA sequence compression using the Burrows-Wheeler transform. In *Proceedings of the IEEE Computer Society Bioinformatics Conference*. Stanford (CA, USA), 2003, p. 303–313. DOI: 10.1109/CSB.2002.1039352

[8] LIANG, J.-Y., CHEN, C.-S., HUANG, C.-H., et al. Lossless compression of medical images using Hilbert space-filling curves. *Computerized Medical Imaging and Graphics*, 2008, vol. 32, no. 3, p. 174–182. DOI: 10.1016/j.compmedimag.2007.11.002

[9] FRUCHTMAN, A., GROSS, Y., KLEIN, S. T., et al. Weighted Burrows-Wheeler compression. *SN Computer Science*, 2023, vol. 4, no. 3, p. 1–12. DOI: 10.1007/s42979-022-01629-5

[10] BAIER, U. On undetected redundancy in the Burrows-Wheeler transform. In *Annual Symposium on Combinatorial Pattern Matching (CPM)*. Qingdao (China), 2018, p. 1–15. DOI: 10.4230/LIPIcs.CPM.2018.3

[11] ABEL, J. Post BWT stages of the Burrows-Wheeler compression algorithm. *Software: Practice and Experience*, 2010, vol. 40, no. 9, p. 751–777. DOI: 10.1002/spe.982

[12] ABDOLLAHI, A., BRUCE, N., KAMALI, S., et al. Lossless image compression using list update algorithms. *Lecture Notes in Computer Science*, 2019, vol. 11811, p. 16–34. DOI: 10.1007/978-3-030-32686-9_2

[13] ŽALIK, B., STRNAD, D., PODGORELEC, D., et al. A new transformation technique for reducing information entropy: A case study on greyscale raster images. *Entropy*, 2023, vol. 25, no. 12, p. 1–13. DOI: 10.3390/e25121591

[14] BAISAKH, N., MOHAPATRA, H., GURU, A. Behavioral insights into compression: A study of move-to-front-or-middle deterministic online algorithm through sequence classification and characterization. *International Journal of Information Technology*, 2024, vol. 17, no. 1, p. 189–203. DOI: 10.1007/s41870-024-02218-w

[15] APRILIANTO, M., ABDUROHMAN, M. Improvement text compression performance using combination of Burrows-Wheeler transform, move-to-front, and Huffman coding methods. *Journal of Physics: Conference Series*, 2014, vol. 495, p. 1–6. DOI: 10.1088/1742-6596/495/1/012042

[16] MOHANTY, R., PATEL, S., DASH, S. P., et al. Some novel results from analysis of move-to-front (MTF) list accessing algorithm. *arXiv*, 2022, p. 1–5. DOI: 10.48550/arxiv.1206.6187

[17] SILUÉ, N., OUATTARA, S., DOSSO, M., et al. Quantitative comparative study of the performance of lossless compression methods based on a text data model. *Open Journal of Applied Sciences*, 2024, vol. 14, no. 7, p. 1944–1962. DOI: 10.4236/ojapps.2024.147127

[18] LI, H. BWT construction and search at the terabase scale. *Bioinformatics*, 2024, vol. 40, no. 12, p. 1–10. DOI: 10.1093/bioinformatics/btae717

[19] CENZATO, D., LIPTÁK, Z. A survey of BWT variants for string collections. *Bioinformatics*, 2024, vol. 40, no. 7, p. 1–10. DOI: 10.1093/bioinformatics/btae333

[20] HUNTER, J. D. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 2007, vol. 9, no. 3, p. 90–95. DOI: 10.1109/MCSE.2007.55

[21] GAGIE, T., MANZINI, G. Move-to-front, distance coding, and inversion frequencies revisited. *Lecture Notes in Computer Science*, 2007, vol. 4580, p. 71–82. DOI: 10.1007/978-3-540-73437-6_10

About the Authors . . .

Andrii SAMOFALOV was born in Severodonetsk, Ukraine, in 1998. He received his M.Sc. in Computer Engineering from Kyiv Polytechnic Institute (KPI) in 2022. He is currently pursuing the Ph.D. degree with the Department of Radio Electronics, Faculty of Electrical Engineering and Communication, Brno University of Technology (BUT). His research interests include data analysis, data compression, image processing and artificial intelligence.

Ladislav POLAK was born in Štúrovo, Slovakia, in 1984. He received the M.Sc. degree in 2009 and the Ph.D. degree in 2013, both in Electronics and Communication from the Brno University of Technology (BUT), Czech Republic. He is currently an Associate Professor at the Department of Radio Electronic (DREL), BUT. His research interests include digital TV broadcasting, mobile and wireless communication systems, RF measurements, signal and data processing.