Shannon's entropy,
information theory,
visualization

Marcin CHOLEWA[1], Malgorzata PALYS[1]

ESTIMATION OF INFORMATION ENTROPY BASED ON ITS VISUALIZATION

This paper describes the method which allows an estimation of information entropy in the meaning of Shannon. The method is suitable to an estimation which sample has a higher value of information entropy. Several algorithms have been used to estimate entropy, assuming that they do it faster. Each algorithm has calculated this value for several text samples. Then analysis has verified which comparisons of the two samples were correct. It has been found that the probabilistic algorithm is the fastest and most effective in returning the estimated value of entropy.

1. INTRODUCTION

Since the discovery, information entropy [10] has found many uses in science [1], [2], [4], [7], [13]. One of the field in which it has found application is digital image processing [5], [8]. It should be noted here that the JPEG image file format uses Shannon's entropy theorem [1] in coding image [12]. Medicine increasingly uses these techniques that help diagnose diseases in humans [3], [9]. The analysis of classical methods of entropy computing has emerged the new method for faster entropy computing. Ultimately, this can be moved into software development that will more quickly return the results from the analyzed sample being the image.

2. VISUALIZATION OF ENTROPY

The entropy information [1] estimation of binary sequence can be done via counting the all ones from this sequence. The visualization of Shannon's entropy can be shown with any n-element sequence composed of 0 and 1 symbols, however 4-bit vector presented as string will be used to facilitate understanding the visualization (Table 1).
It should be followed by a rule. From Table 1, it follows that sequence consisting only of the same bits (bit 0 or 1) have the lowest entropy equal to zero. When the sequence consists of equal number of bits 0 and 1, its entropy is the greatest. Symbols between $\langle\,\rangle$ means string sequence. The *Binary vector* is binary representation of number in *Decimal value* column represented as string. Another column *Number of different pairs* contains an integer number of estimated entropy.

[1]University of Silesia, Institute of Computer Science, Będzińska 39, 41-200 Sosnowiec, Poland
e-mail: {marcin.cholewa, malgorzata.palys}@us.edu.pl

Table 1. The visualization of information entropy for a 4-bit binary vector.

No.	Decimal value	Binary vector	Number of different pairs (integer entropy)	Shannon's entropy
1	0	⟨0000⟩	0	0.000
2	1	⟨0001⟩	3	0.811
3	2	⟨0010⟩	3	0.811
4	3	⟨0011⟩	4	1.000
5	4	⟨0100⟩	3	0.811
6	5	⟨0101⟩	4	1.000
7	6	⟨0110⟩	4	1.000
8	7	⟨0111⟩	3	0.811
9	8	⟨1000⟩	3	0.811
10	9	⟨1001⟩	4	1.000
11	10	⟨1010⟩	4	1.000
12	11	⟨1011⟩	3	0.811
13	12	⟨1100⟩	4	1.000
14	13	⟨1101⟩	3	0.811
15	14	⟨1110⟩	3	0.811
16	15	⟨1111⟩	0	0.000

Let $\langle a_0a_1a_2a_3...a_{n-1}\rangle$ be input sequence composed of n elements and P be set all possible pairs from all $\langle a_0a_1a_2a_3...a_{n-1}\rangle$ elements, then estimated entropy (hereinafter referred to as *integer entropy*) is defined as:

$$En(\langle a_0a_1a_2a_3...a_{n-1}\rangle) = \sum_P \gamma(\{a_i, a_j\}), \tag{1}$$

where the function γ is defined as:

$$\gamma(\{a_i, a_j\}) = \begin{cases} 1, & a_i \neq a_j \\ 0, & a_i = a_j \end{cases} \text{ for } i, j = 0, 1, ..., n-1. \tag{2}$$

The mentioned P set symbolizes set of all pairs from binary vector $\langle a_0a_1a_2a_3...a_{n-1}\rangle$ generated via algorithm:

function $En_I(\langle a_0a_1a_2a_3...a_{n-1}\rangle)$
for $i \leftarrow 0$ **to** $\texttt{LENGTH}(\langle a_0a_1a_2a_3...a_{n-1}\rangle) - 1$
  for $j \leftarrow i+1$ **to** $\texttt{LENGTH}(\langle a_0a_1a_2a_3...a_{n-1}\rangle) - 1$
    $P \leftarrow P \cup \{\texttt{GET\_ELEMENT}(\langle a_0a_1a_2a_3...a_{n-1}\rangle, i), \texttt{GET\_ELEMENT}(\langle a_0a_1a_2a_3...a_{n-1}\rangle, j)\}$
return P

The $\texttt{GET\_ELEMENT}(\langle a_0a_1a_2a_3...a_{n-1}\rangle, i)$ function returns only one element at i-position from $\langle a_0a_1a_2a_3...a_{n-1}\rangle$. The algorithm En_I is graphically presented on Figure 1.

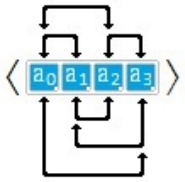

Fig. 1. The visualization of En_I algorithm.

The arrows that are shown on Figure 1 mean pairs. From n-elements sequence can be generated all possible pairs which its number is equal to $\binom{n}{2}$. In the case of 4-length binary vector, number of pairs equal to $\binom{n}{2} = 6$.

Positions of elements in pairs do not matter, because it follows directly from the function γ so that $\gamma(\{a_i, a_j\}) = \gamma(\{a_j, a_i\})$. Normally, arrangement elements in sequence $\langle a_0 a_1 a_2 a_3 ... a_{n-1} \rangle$ does not affect the integer entropy. However, in some cases such influence exists. This is shown in Figure 2. Furthermore, arrangement is compelling when using a pairs generator that produces less of it to decrease the speed of the algorithm.

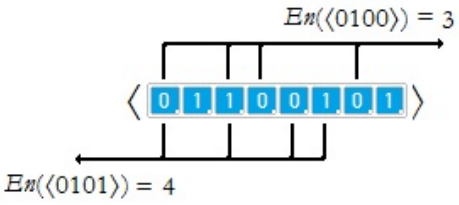

Fig. 2. An example of the influence of element selection on the integer entropy En value.

As mentioned earlier, the way of computing integer entropy for 4 elements can be expanded in the same way for n-elements. The entropy values obtained from the presented algorithm En_I are very close to Shannon's entropy [10], [11] (Fig. 3).

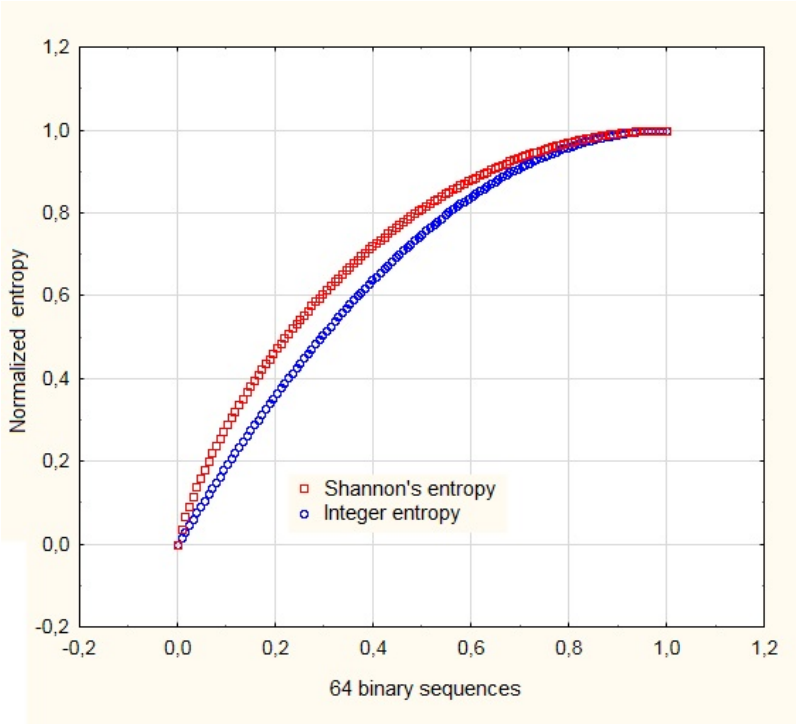

Fig. 3. The visualization of the integer entropy and Shannon's entropy for 128-elements binary sequence.

In the Figure 3 the x-axis shows only binary sequences that can be decimal as $0, 1, 3, 7, 15$, 31,..., $2^{128} - 1$ which gives only 64 sequences of all 2^{128} possible binary sequences. To improve readability the graph in Figure 3, values on the x-axis have been normalized to range $[0, 1]$ according to the rule $\frac{0}{\omega}, \frac{1}{\omega}, \frac{3}{\omega}, \frac{7}{\omega}, \frac{15}{\omega}, \frac{31}{\omega}$, ..., $\frac{2^{128}-1}{\omega}$, where $\omega = 2^{128} - 1$. The choice of such binary sequences is justified by the fact that these sequences determine all possible values of entropy for both Shannon [10] and integer entropies. To better illustrate this difference, see Figure 4. Normalized integer entropy was calculated using $En_{\text{NORMALIZED}}(\langle a_0 a_1 a_2 a_3 ... a_{63} \rangle) = \frac{En(\langle a_0 a_1 a_2 a_3 ... a_{63} \rangle)}{En_{\text{MAX}}(\langle a_0 a_1 a_2 a_3 ... a_{63} \rangle)}$ formula, where $En_{\text{MAX}}(\langle a_0 a_1 a_2 a_3 ... a_{63} \rangle) = En(\langle \underbrace{111...1}_{32} \underbrace{000...0}_{32} \rangle)$.

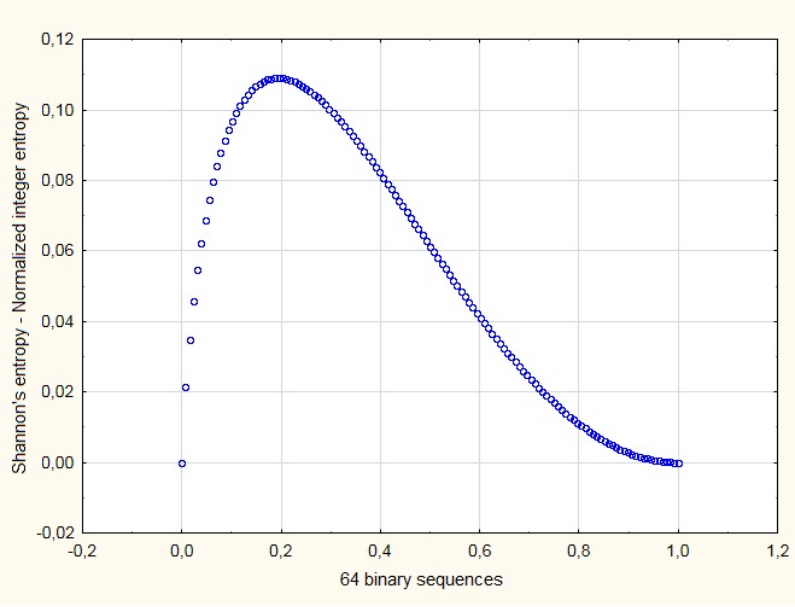

Fig. 4. The visualization of differences between the integer and Shannon's entropy.

It should be noted that when the different pairs increase, the difference between Shannon's entropy and integer entropy is decreased.

3. EXPERIMENTS AND EXPECTATIONS

The purpose of the analysis will be to select an algorithm that generates pairs the fastest. Such an algorithm must also return the correct value of the integer entropy. Validation of the correct integer entropy for a sequence can be made when it will compare with the integer entropy of another a sequence. However, the entropy of the information according to Shannon's rule for those sequences must be calculated first. The algorithms under investigation do not examine the distribution of symbols in a sequence, as this would lead to a significant increase in executing time. It can be said that the presented algorithms are variations of the construction of two *for* loops. The following algorithms of generation pairs will be checked:

I. The naive algorithm $En_I(\langle a_0a_1a_2a_3...a_{n-1}\rangle)$ - this algorithm was presented in the second chapter.

II. The random algorithm $En_{II}(\langle a_0a_1a_2a_3...a_{n-1}\rangle)$. In this method, the pairs are randomly selected. A random number generator is used which minimizes the probability of selecting a number that has already been selected. The $\alpha \in \mathbb{N}$ coefficient determines how many pairs will be drawn together and it satisfies the equation $\binom{n}{2} > (n-1)\alpha$ from a certain n. In this research the value of $\alpha = 4$ was assumed.

function $En_{II}(\langle a_0a_1a_2a_3...a_{n-1}\rangle)$
for $i \leftarrow 0$ **to** $(\texttt{LENGTH}(\langle a_0a_1a_2a_3...a_{n-1}\rangle) - 1) \cdot \alpha$
$a_r = \texttt{GET\_ELEMENT}(\langle a_0a_1a_2a_3...a_{n-1}\rangle, [\texttt{RAND}(0, n-1)])$
$a_l = \texttt{GET\_ELEMENT}(\langle a_0a_1a_2a_3...a_{n-1}\rangle, [\texttt{RAND}(0, n-1)])$
$P \leftarrow P \cup \{a_r, a_l\}$
return P

III. The algorithm of k-nearest adjacent changes $En_{III}(\langle a_0a_1a_2a_3...a_{n-1}\rangle)$. It can be selected

the following number of neighbors $k = 2, 4, 6, 8, ..., k_{max}$, where k_{max} should not exceed $\frac{2}{3}n$ to the algorithm returns the correct values for integer entropy.

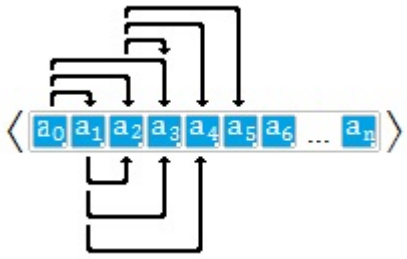

Fig. 5. An example of 3-nearest adjacent changes for a_0, a_1, a_2 element in sequence $\langle a_0a_1a_2a_3...a_{n-1}\rangle$.

IV. The algorithm selecting the pairs according to a sequence $En_{IV}(\langle a_0a_1a_2a_3...a_{n-1}\rangle)$. In this research, the sequence is $\langle 101010...10\rangle$. When encountered symbol 0 at i-position, algorithm omits appropriate element a_i.

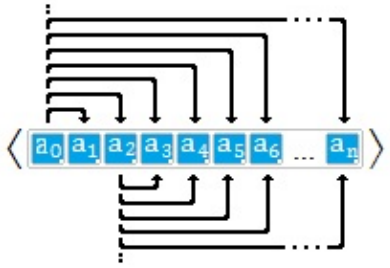

Fig. 6. The visualization of algorithm generating pairs from every second element of binary vector $\langle a_0a_1a_2a_3...a_{n-1}\rangle$ for three elements a_0, a_1 and a_2.

V. The algorithm with *step* instructions $En_V(\langle a_0a_1a_2a_3...a_{n-1}\rangle)$. Applying a *step* instruction to the *for* loop greatly speeds up the operation of this algorithm. This value for the *step* instruction was set to generate enough pairs to compute the integer entropy value.

function $En_V(\langle a_0a_1a_2a_3...a_{n-1}\rangle)$
for $i \leftarrow 0$ **to** $\texttt{LENGTH}(\langle a_0a_1a_2a_3...a_{n-1}\rangle) - 1$ **step 3**
  for $j \leftarrow i + 1$ **to** $\texttt{LENGTH}(\langle a_0a_1a_2a_3...a_{n-1}\rangle) - 1$ **step 3**
    $P \leftarrow P \cup \{\texttt{GET\_ELEMENT}(\langle a_0a_1a_2a_3...a_{n-1}\rangle, i), \texttt{GET\_ELEMENT}(\langle a_0a_1a_2a_3...a_{n-1}\rangle, j)\}$
return P

VI. The algorithm calculating the Shannon entropy $H(\langle a_0a_1a_2a_3...a_{n-1}\rangle)$. For n-elements sequence $\langle a_0a_1a_2a_3...a_{n-1}\rangle$ consists with values from 256-elements set and statistics set $\{S_0, S_1, ..., S_{255}\}, S_0, S_1, ..., S_{255} \in \mathbb{N}$, exists algorithm calculating the entropy H:

function $H(\langle a_0a_1a_2a_3...a_{n-1}\rangle)$
for $i \leftarrow 0$ **to** $\texttt{LENGTH}(\langle a_0a_1a_2a_3...a_{n-1}\rangle) - 1$
  $S_{\texttt{GET\_ELEMENT}(\langle a_0a_1a_2a_3...a_{n-1}\rangle, i)} \leftarrow S_{\texttt{GET\_ELEMENT}(\langle a_0a_1a_2a_3...a_{n-1}\rangle, i)} \cup$
  $\cup \{\texttt{GET\_ELEMENT}(\langle a_0a_1a_2a_3...a_{n-1}\rangle, i)\}$
for $i \leftarrow 0$ **to** 255
  if $S_i \neq \varnothing$
    $p \leftarrow \frac{card(S_i)}{\texttt{LENGHT}(\langle a_0a_1a_2a_3...a_{n-1}\rangle)}$
    $h \leftarrow p \cdot \log p$
    $H \leftarrow H + h$

return $-H$

Each algorithm will be tested on two groups of strings. The first group contains strings that form the text of a natural language. The second group contains randomly generated character strings. In each group there are 10 strings of different lengths, from which 5 comparisons will be done. The analysis deals with strings consist of 8, 16, 32, 64 and 128 characters.

For two s_1 and s_2 strings of the same length, the integer entropies $En_A(s_1), En_A(s_2)$ are calculated by means of algorithms $A = \{I, II, III, IV, V\}$ and by the standard algorithm for entropy H for $H(s_1), H(s_2)$ according to the Shannon's formula. First, we compare the entropy of the Shannon's formula to determine the correct result of the entropy comparison. The possible result r_H of the comparison is from the set $r_H \in \{H(s_1) = H(s_2), H(s_1) < H(s_2), H(s_1) > H(s_2)\}$ and then r_A for the integer entropies will be done by algorithms En_A indexed by A. If the result $r_H = r_A$, then the comparison is correct for the given algorithm A. The expectation is that as many comparisons are valid for the given algorithm in the shortest possible time of its execution.

4. RESULT

In this study, the choice of programming language is irrelevant, because the relationship between the positions of the instruction remains constant. In every procedural language the presented algorithms $En_I, En_{II}, En_{III}, En_{IV}, En_V, H$ can be expressed in the same way. In Tables 2 and 3, the unit of comparison time is *tu*, which is the average execution time of all the most important instructions in the algorithms involved in the research. The following tables show the results of the analysis of character strings by pair's generation algorithms.

Table 2. Times and integer entropies from the real data.

Real data sequence												
Comparison no.			1		2		3		4		5	
Sequence length			8		16		32		64		128	
Algorithms	En_I	Entr. int.	28	25	117	117	467	466	1898	1890	7573	7538
		Time [tu]	206	203	855	855	3477	3476	14060	14052	56471	56436
	En_{II}	Entr. int.	32	21	58	58	119	117	243	239	476	462
		Time [tu]	290	279	572	572	1154	1143	2293	2289	4574	4560
	En_{III}	Entr. int.	13	13	102	102	452	451	1884	1875	7560	7523
		Time [tu]	116	116	765	765	3387	3386	13971	13962	56383	56346
	En_{IV}	Entr. int.	21	20	84	84	325	317	1281	1266	5093	5077
		Time [tu]	165	164	628	628	2437	2429	9601	9586	38117	38101
	En_V	Entr. int.	6	6	15	15	65	61	216	203	888	890
		Time [tu]	51	51	120	120	486	482	1649	1636	6653	6655
	H	Entropy	3.0000	2.4056	3.6250	3.6250	3.6914	3.6639	4.0080	3.9516	3.9174	3.9152
		Time [tu]	2219	1864	3133	3133	3537	3537	4523	4346	4537	4893

The presented algorithms generating pairs have been verified practically. Many attempts have been made which indicates that in a large number of algorithms return the correct values of integer entropy. The correct values have been verified during comparisons. A total of 50 comparisons were made and 44 were indicates as valid. For real and random sequences using the En_V algorithm, 2 incorrect comparisons were made. Regarding the return of integer entropy, the En_V algorithm turned out to be the worst. Repeatedly repeated experiments for the En_V algorithm did not improve. Similarly, for the En_{III} algorithm with 8-elements sequences, the result could not be improved too.

Table 3. Times and integer entropies from the random data.

Random data sequence												
Comparison no.			1		2		3		4		5	
Sequence length			8		16		32		64		128	
Algorithms	En_I	Entr. int.	25	27	117	116	482	479	1958	1943	7854	7881
		Time $[tu]$	203	205	855	854	3492	3489	14120	14105	56752	56779
	En_{II}	Entr. int.	27	30	60	58	125	122	249	240	494	495
		Time $[tu]$	285	288	574	572	1151	1148	2299	2290	4592	4593
	En_{III}	Entr. int.	11	12	102	102	468	465	1944	1928	7839	7866
		Time $[tu]$	114	115	765	765	3403	3400	14031	14015	56662	56689
	En_{IV}	Entr. int.	18	20	84	82	329	324	1331	1316	5290	5297
		Time $[tu]$	162	164	628	626	2441	2436	9651	9636	38314	38321
	En_V	Entr. int.	5	6	15	14	62	62	225	229	906	916
		Time $[tu]$	50	51	120	119	483	483	1658	1662	6671	6681
	H	Entropy	2.4056	2.7500	3.6250	3.5000	4.2417	4.1403	4.6186	4.4726	4.7315	4.8078
		Time $[tu]$	1864	2042	3133	2956	4606	4427	5770	5594	6497	6497

In the study also checked the execution times of algorithms generating pairs of symbols from strings. It would be easy to define these times without performing algorithms; however, the value of the integer entropy returned by these algorithms is important. It allows checking whether integer entropies are calculated well.

The results show how important a random factor is, therefore, the best times have been obtained by the En_{II} algorithm. Algorithm En_{II} is so fast that it can overtake the algorithm H. The En_V algorithm also has good timing, but it does not always return the correct value of integer entropy. It can be admitted that the presented random algorithm is essentially a Monte Carlo method [6], which accelerates the calculation of the entropy value of the information.

The naive algorithm En_I is not worthy of use because the time of computing integer entropy far exceeds the entropy calculation time by the H algorithm. For real strings, time is 11 times longer than for random strings. This algorithm is also not worth developing and is suitable to return the correct value of integer entropy in further analyzes.

In real data, 16-character strings were chosen so that the entropy was equal. It was the trial whether it is possible detects entropy equation between two strings using integer entropy methods. These results provide confirmatory evidence that all algorithms deal with this situation.

It is also worth noting that all algorithms for short strings (8, 16, 32-characters) generate faster integer entropy values. Hence, here comes the conclusion that the algorithms are suitable for short samples.

5. FURTHER WORKS

The study focused on samples up to 128 elements. Further studies will include larger samples. To do this, the new research will use some of the existing $En_I, En_{II}, En_{III}, En_{IV}, En_V$ functions and match them to the modified input samples. Sample modification will consist in dividing them in such a way as to preserve the entropy information. In extended studies, we will also look for the dependence of element selection on the integer entropy value calculated from the selected elements. The dependency must strive for the principle of less elements and more accurate value of integer entropy. The purpose of such searches will be to find patterns of selection and to generalize them to a larger class of samples.

BIBLIOGRAPHY

[1] CHOLEWA M. Shannon information entropy as complexity metric of source code. 2017 MIXDES - 24th International Conference Mixed Design of Integrated Circuits and Systems, Bydgoszcz, 2017. pp. 468–471.

[2] CZYŻ T., HAUKE J. Zastosowanie miary entropii do badania zmian w zróżnicowaniu regionalnym Polski. Rozwój Regionalny i Polityka Regionalna, 2015, Vol. 35. pp. 17–30.

[3] DE ARRUDA P. F., GATTI M., FACIO JR. F. N., DE ARRUDA J. G., MOREIRA R. D., MURTA JR. L. O., DE ARRUDA L. F., DE GODOY M. F. Quantification of fractal dimension and Shannon's entropy in histological diagnosis of prostate cancer. BMC Clinical Pathology, 2013, Vol. 13, No. 1.

[4] EIMANN R. Network event detection with entropy measures. Ph. D. Thesis, University of Auckland, Auckland, New Zealand, 2008.

[5] GULL S. F., SKILLING J. Maximum entropy method in image processing. IEE Proceedings, 1984, Vol. 131, No. 6. pp. 646–659.

[6] KALOS M. H., WHITLOCK P. A. Monte Carlo methods. Wiley - VCH Verlag GmbH & Co. KGaA, 2008.

[7] LIN J. Divergence measures based on the Shannon entropy. IEEE Transactions on information Theory, 1991, Vol. 37, No. 1. pp. 145–151.

[8] RAHMAN S., RAHMAN M., ABDULLAH-AL-WADUD M., SCHOYAIB M. An adaptive gamma correction for image enhancement. EURASIP Journal on Image and Video Processing, 2016, Vol. 35, No. 1.

[9] RAJALAXMI S., NIRMALA S. Entropy-based straight kernel filter for echocardiography image denoising. Journal of Digital Imaging, 2014, Vol. 27, No. 5. pp. 610–624.

[10] SHANNON C. E. A mathematical theory of communication. Bell System Technical Journal, 1948, Vol. 27, No. 3. pp. 379–423.

[11] SHLENS J. A light discussion and derivation of entropy. CoRR, 2014, Vol. abs/1401.1998.

[12] WALLACE G. K. The JPEG still picture compression standard. IEEE Transactions on Consumer Electronics, 1992, Vol. 38, No. 1.

[13] WĘDROWSKA E. Wykorzystanie entropii Shannona i jej uogólnień do badania rozkładu prawdopodobieństwa zmiennej losowej dyskretnej. Przegląd statystyczny, 2010, Vol. 57, No. 4.