Information Fusion 41 (2018) 161–175

Contents lists available at ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus

A tutorial review on entropy-based handcrafted feature extraction for information fusion

Rodrigo Capobianco Guido

Instituto de Biociências, Letras e Ciências Exatas, Unesp - Univ Estadual Paulista São Paulo State University, Rua Cristóvão Colombo 2265, Jd Nazareth, São José do Rio Preto - SP, 15054-000, Brazil

ARTICLE INFO

Article history:
Received 19 June 2017
Revised 27 August 2017
Accepted 2 September 2017
Available online 12 September 2017

Keywords:
Entropy
Handcrafted feature extraction
Information fusion
Deep networks
Restricted-vocabulary speech recognition
Image synthesis

ABSTRACT

Entropy (H) is the main subject of this article, concisely written to serve as a tutorial introducing two feature extraction (FE) methods for usage in digital signal processing (DSP) and pattern recognition (PR). The theory, carefully exposed, is supplemented with numerical cases, augmented with C/C++ source-codes and enriched with example applications on restricted-vocabulary speech recognition and image synthesis. Complementarily and as innovatively shown, the ordinary calculation of H corresponds to the outcome of a partially pre-tuned deep neural network architecture which fuses important information, bringing a cutting-edge point-of-view for both DSP and PR communities.

1. Introduction

1.1. Objective and text structure

This is the third in a set of tutorials I have recently published with the same objective: innovative usage of humble and well-known concepts for the benefit of both digital signal processing (DSP) and pattern recognition (PR) communities. The preceding texts, [23] and [24], were respectively dedicated to the exploration of relevant aspects of *energy* by means of proposed methods A_1, A_2 and A_3, and *zero-crossing rates* (ZCRs), according to the techniques introduced as B_1, B_2 and B_3. Successfully, I employed those formulations for neurophysiological signal analysis, texture characterisation, text-dependent speaker verification, speech classification and segmentation, image border extraction and biomedical signal processing. Energy, that is used to express the potential to perform work, as well as ZCRs, which are commonly applied to elementary spectral content analysis, act disparately in correlation to *entropy* (H) [13,70], the feature explored in this article.

Despite the emerging deep learning (DL) technologies employed for automatic feature learning [22,34], handcrafted feature extraction (FE), i.e., the situation in which the system engineer chooses the appropriate features to be extracted from the signal under analysis, continues to play an important role in DSP and PR. Particularly, I demonstrate that H, by itself, obtained based on two proposed approaches for FE from both unidimensional (1D) and bidimensional (2D) data, has flagrant potential, as also evidenced in relevant scientific articles published last year [11,17,20,42,50,51,53,81,84,86] and a few years ago [6,15,21,37,56,57,62,63,73,75,83]. Similarly to the characterization of ZCRs as neurocomputing agents [24], H is shown to be the outcome of a specifically tuned deep neural network (DNN) that fuses important information, bringing an innovative point-of-view for both DSP and PR communities. Furthermore, experiments and applications on restricted-vocabulary speech recognition and image synthesis reassure the efficacy of the proposed techniques.

Compromised with a balance among *creativity, simplicity* and *accuracy*, exactly as in [23] and [24], this paper is organised as follows. A review on H accompanied by some of its recent applications is the theme of the next subsection. Section 2, oppositely, describes the proposed approaches in detail and my particular point-of-view about H for both 1D and 2D signals. Proceeding, Section 3 presents some numerical examples which complement the theoretical explanations, easing their comprehension. Illustrative experiments and applications involving 1D and 2D signals can be found in Section 4 and, lastly, I conclude the paper. Following my previous rationale, as indicated in [24]-pp.1 with respect to article [23], I emphasise the importance of those articles, suggesting their readings beforehand for a better understanding of the ideas discussed herein.

E-mail address: guido@ieee.org
URL: http://www.sjrp.unesp.br/~guido/

http://dx.doi.org/10.1016/j.inffus.2017.09.006
1566-2535/

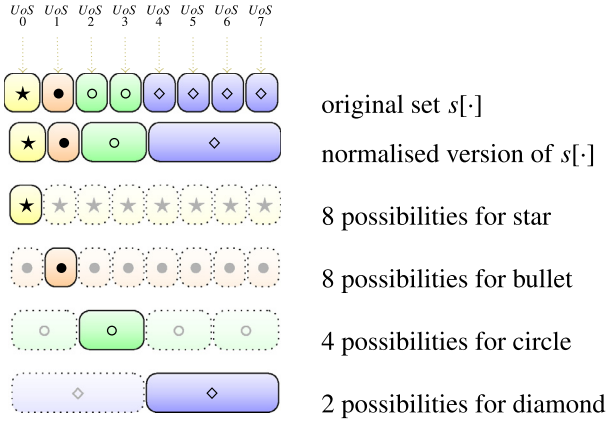


Fig. 1. The way normalisations should be interpreted for the example set $s[\cdot] = \{\star, \bullet, \circ, \circ, \diamond, \diamond, \diamond, \diamond\}$. *UoS* means "unit of space".

1.2. A review on entropy and its applications

The records I found on the *Web of Science* website show that the first published article about H is dated from 1900 [82]. As science advances, H has been at the forefront of research in a diversity of fields such as general physics [39], thermodynamics [10], astrophysics [8], statistical mechanics [67], genetics [65], economics [46] and arts [1]. In order to understand its essence, focusing particularly on the field of Information Theory, where the DSP and PR communities find utility [19,49], the readers are first requested to reflect on the meaning of *information* [39]-pp.117, as follows. The exposition hereafter is inspired by the traditional article [68] published in 1948 by Claude Shannon[1], who is considered the father of Information Theory, and by additional respected bibliographical materials.

Let $p_i = \frac{\alpha_i}{M} = \frac{1}{(M/\alpha_i)}$, $(0 \leq i < K)$ and $(0 \leq \alpha_i < M)$, be the *probability* [59] of the ith distinct *datum*, i.e., symbol, in a set $s[\cdot] = \{s_0, s_1, s_2, \ldots, s_{M-1}\}$ of size M with K distinct symbols. Consequently, there are α_i symbols within M matching the ith expectation or, equivalently, there is one among $\frac{1}{(M/\alpha_i)}$. Specifically, the denominator (M/α_i) represents the normalised number of possibilities for the ith symbol, with the normalisation interpreted in such a way that each subset of repeated elements is converted into an unique size representative of size equal to that of the entire subset. Furthermore, (M/α_i) written based on a certain alphabet β produces words for which **the length** corresponds to what we know as *information*.

Assume, as an example, the 8-sample long set $s[\cdot] = \{\star, \bullet, \circ, \circ, \diamond, \diamond, \diamond, \diamond\}$. The probabilities of stars, bullets, circles and diamonds are, respectively, $p_0 = \frac{1}{(M/\alpha_0)} = \frac{1}{(8/1)} = \frac{1}{8}$, $p_1 = \frac{1}{(M/\alpha_1)} = \frac{1}{(8/1)} = \frac{1}{8}$, $p_2 = \frac{1}{(M/\alpha_2)} = \frac{1}{(8/2)} = \frac{1}{4}$ and $p_3 = \frac{1}{(M/\alpha_3)} = \frac{1}{(8/4)} = \frac{1}{2}$. Thus, the corresponding normalised number of possibilities for each star, bullet, circle and diamond is $(8/1) = 8$, $(8/1) = 8$, $(8/2) = 4$ and $(8/4) = 2$. Fig. 1 illustrates the physical meaning of the normalisations. Regarding the star, only one unit of space among eight is required for its placement; thus, there are eight possibilities to place it. The same holds true for the bullet. In relation to both circles in the original set, the normalisation converts them in only one double-length circle and forces it to occupy two original units of space among eight; thus, there are four possible placements for the bigger circle. Lastly, the four original diamonds are converted, due to the normalisation, into only one larger diamond which occupies four original units of space, implying that there are only two possible placements for this enlarged symbol.

When choosing the binary basis [30], i.e., $= 2$, as being the alphabet, "0" and "1" are the only existing characters, known as bits, which compose the corresponding words. Particularly,

- "000", "001", "010", "011", "100", "101", "110" and "111" are the $2^3 = 8$ possibilities for placing stars, implying that **three** bits are needed to express such locations;
- "000", "001", "010", "011", "100", "101", "110" and "111" are also the $2^3 = 8$ possibilities for placing bullets, which consequently require **three** bits to express such locations;
- "00", "01", "10" and "11" are the $2^2 = 4$ possibilities for placing circles, which require **two** bits to express such locations;
- "0" and "1" are the $2^1 = 2$ possibilities for placing diamonds, which require only **one** bit to express such locations.

Instead of written and counted, the number of bits in each case may be easily calculated by means of the base $\beta = 2$ logarithms of the normalized possibilities ([30]-pp.8), i.e., $log_\beta(\frac{M}{\alpha_0}) = log_\beta(\frac{1}{p_0}) = log_2(8) = 3$, $log_\beta(\frac{M}{\alpha_1}) = log_\beta(\frac{1}{p_1}) = log_2(8) = 3$, $log_\beta(\frac{M}{\alpha_2}) = log_\beta(\frac{1}{p_2}) = log_2(4) = 2$ and $log_\beta(\frac{M}{\alpha_3}) = log_\beta(\frac{1}{p_3}) = log_2(2) = 1$. These values are the lengths, i.e., the number of bits, of the words in each case, which correspond to their **information**.

Generally, $log_\beta(\frac{1}{p_i})$ expresses information in terms of the number of elements required to write $\frac{1}{p_i}$ possibilities for placements based on alphabet β, i.e.,

the amount of information for the ith symbol is

$$\underbrace{log_\beta \overbrace{\left(\frac{1}{p_i}\right)}^{\substack{\text{normalised number of possibilities} \\ \text{for the } i^{th} \text{ symbol}}}}_{\substack{\text{information, i.e., number of elements required} \\ \text{to write } (1/p_i) \text{ using alphabet } \beta}}.$$

In order to obtain a **global quantification** for the information in the **entire** set $s[\cdot]$, the more natural procedure corresponds to the calculation of the weighted sum of the independent amounts of information, where the probabilities of occurrences are the respective weights. Thus,

$$H = \sum_{i=0}^{K-1} p_i \cdot log_\beta\left(\frac{1}{p_i}\right). \tag{1}$$

Alternatively and taking into account the property of logarithms which states that $log(\frac{1}{x}) = -log(x)$, $\forall x \in \Re^+$, Eq. (1) may be rewritten as

$$H = -\sum_{i=0}^{K-1} p_i \cdot log_\beta(p_i),$$

that is the most traditional way used to express entropy. In our previous example, $H = -\sum_{i=0}^{K-1} p_i \cdot log_\beta(p_i) = -\sum_{i=0}^{3} p_i \cdot log_2(p_i)$ $= -((p_0 \cdot log_2(p_0)) + (p_1 \cdot log_2(p_1)) + (p_2 \cdot log_2(p_2)) + (p_3 \cdot log_2(p_3)))$ $= -((\frac{1}{8} \cdot log_2(\frac{1}{8})) + (\frac{1}{8} \cdot log_2(\frac{1}{8})) + (\frac{1}{4} \cdot log_2(\frac{1}{4})) + (\frac{1}{2} \cdot log_2(\frac{1}{2})))$ $= \frac{3}{8} + \frac{3}{8} + \frac{1}{2} + \frac{1}{2} = \frac{7}{4}$ bits.

Deservedly also known as *Shannon's entropy*, H may alternatively be understood as a measure of unpredictability of information content, whereas it equals zero upon a concrete and fully predictable outcome, i.e., when $K = 1$ and $p_0 = 1$. Particularly concerning our example, the unpredictability of the normalised diamond is the lowest: there are only two possible placements for it, as seen in Fig. 1. In contrast, normalised circles have a higher unpredictability because there are four possible placements for them. Lastly, both normalised stars and bullets present the highest unpredictabilities amongst the symbols with eight possible placements.

[1] Had he not passed away in 2001, Dr Shannon would have celebrated his 100th birthday on April 30, 2016.

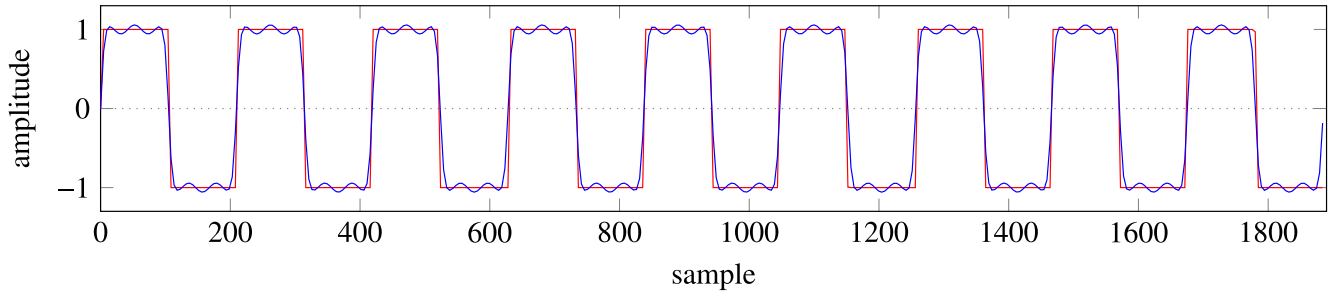


Fig. 2. Two square waves: clean and noisy, respectively drawn in red and blue. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Attention is required to the fact that neither the particular meaning of the symbols in $s[\cdot]$ nor the specific positions they occupy affect the value of H.

In order to provide an additional example, the reader is required to let $s[\cdot] = \{7, 7, 7, 5, 5, 5\}$ be the input vector under analysis. Among $M = 6$, there are only $K = 2$ different symbols, or events, in $s[\cdot]$, which correspond to "7" and "5". Thus, the respective probability vector derived from $s[\cdot]$ is $p[\cdot] = \{probability\ of\ \text{'7'}, probability\ of\ \text{'5'}\} = \{\frac{3}{6}, \frac{3}{6}\} = \{\frac{1}{2}, \frac{1}{2}\}$, implying that $H(s[\cdot]) = -\sum_{i=0}^{K-1} p_i \cdot log_\beta(p_i) = -\frac{1}{2} \cdot log_\beta(\frac{1}{2}) - \frac{1}{2} \cdot log_\beta(\frac{1}{2}) = -log_\beta(\frac{1}{2})$. Similar inputs, such as, for instance, $s[\cdot] = \{5, 5, 5, 7, 7, 7\}$, $s[\cdot] = \{5, 7, 5, 7, 5, 7\}$ or $s[\cdot] = \{5, 5, 7, 7, 5, 7\}$, take to the same value for H. Additionally, if other symbols replace "7" and "5", implying, for instance in $s[\cdot] = \{1, 1, 1, 9, 9, 9\}$, $s[\cdot] = \{6, 8, 8, 6, 6, 8\}$ or $s[\cdot] = \{-1, -1, 4, 4, 4, -1\}$, H is still the same. Letting $\beta = 2$, H corresponds to average number of bits required to represent the digital signal $s[\cdot]$. Optionally, bases $\beta = 10$, $\beta = e \approx 2.7182818$ and $\beta = \pi \approx 3.1415927$ may be adopted so that H represents, respectively, average number of "digits", number of "nats" and number of "slices".

Fundamentally and in comparison to the concepts of ene.g. [23] and ZCR [24], entropy can reveal characteristics from the signal under analysis which are unlikely found based on those features. The humble example shown in Fig. 2 illustrates this capacity. Assuming that a fixed-length window is adopted to analyse the signals, neither energy nor ZCRs can be successfully employed to distinguish between the clean and the noisy square waves, drawn respectively in red and blue. On one hand, the energy from both signals is practically the same because the positive spikes of the sinusoidal-like noise compensate the amplitude of the negative ones. Accordingly and regarding the ZCRs, both curves cross zero exactly the same number of times in a certain interval. On the other hand, the noisy signal requires more information to be characterised in comparison to its clean version because they differ in terms of the number of distinct amplitude values they contain. Thus, entropy can easily distinguish the signals whereas both energy and ZCRs fail to do so.

Overall, entropy has supported a considerable number of DSP and PR algorithms in the fields of speech and audio processing, image treatment and biomedical signal analysis. In paper [60], authors successfully use neural-scaled entropy (NSE) and cochlear-scaled entropy (CSE) in order to predict the effects of nonlinear frequency compression on speech perception due to sensorineural hearing loss (SNHL). In speech synthesis, the major limitations imposed with the use of tree-clustered context-dependent hidden semi-Markov models adopted are fruitfully circumvented by the authors of paper [40] based on entropy.

Speech emotion recognition algorithms, such as those developed by the authors of paper [72], have been frequently based on entropy, sometimes associated with time-frequency analysis (TFA). Similarly, TFA has been associated with entropy in order to evaluate speech dysfluencies and stuttering, as reported in paper [29]. Speech intelligibility in noisy environments, another relevant theme, can be predicted with CSE evaluation, as documented in [12] and [74]. Entropy has also played an important role in speech recognition, as shown by the authors of papers [48,54,78]. Particular types of speech signals may be accurately characterized with the use of entropy, as the authors of paper [85] demonstrate.

Relevant issues on the relationship between entropy and music are documented in article [14]. Music characterization and genre classification by means of entropy is explored in article [61]. Accordingly, Meragi's songs are evaluated with a combined entropy-fractal approach in paper [4]. Another interesting related approach can be found in article [44] in which authors characterise music styles based on entropy. Coupled to a hidden markov model (HMM) and a multilayer perceptron (MLP), music and speech signals are segmented on the basis of entropy according to article [2]. Maximum entropy learning is the fundamental concept for the development of a useful frequency component restoration algorithm, as shown in article [36]. Particular emotions expressed by pop music singers are found on the basis of entropy, being retrieved on the basis of the procedures presented in article [33]. Finally, online music identification is the theme of article [77], in which the authors successfully work with almost ten thousand songs and reach over 90% of accuracy.

Image processing and computer vision are singular fields for which entropy plays a relevant role. Coloured images are segmented based on entropy according to the proposals found in article [7]. Similarly, entropy is associated with a combination of unidimensional fuzzy classifiers in order to segment images as shown in paper [79]. An entropy-based watermarking scheme is successfully implemented in article [47], robust against traditional signal processing attacks. Noisy spectral image fusion is the focus of article [87], in which the authors apply entropy to minimize the negative effects of noise on the selections of image features. Segmentation of river images on the basis of cross entropy is explored in paper [69], more accurate than the traditional Chan–Vese's model.

In paper [45], the authors propose the use of maximal entropy random walk on a graph for tampering localization in digital image forensics. Interestingly, the authors of paper [5] introduce an entropy-based method for segmenting the bone regions contained in X-ray images from their surrounding muscles and tissues. An alignment method for inverse synthetic aperture radar image formation, a hyper-spectral image segmentation using Renyi entropy and a fuzzy image texture analysis and classification approach are, respectively, the topics of papers [80], [64] and [58]. Importantly, a new uncertainty measure for image segmentation called average entropy is defined by the authors of paper [43].

Lastly, it is important to notice that, once they are used for FE, entropy-based feature vectors are usually associated with specific classifiers, such as those documented in [52] and [76] which describe a graph-based semi-supervised learning algorithm and a

fuzzy k-means clustering scheme, respectively. Overall, the reviews presented in this section cover the state-of-the-art in the field so that readers get involved with entropy.

2. The proposed approaches

2.1. Calculating H in practice

Prior to studying the proposed approaches, C_1 and C_2, the readers are provided with the pseudo-Algorithm 1, created for calculating the entropy of a digital signal $s[\cdot]$ of length M. Different types of signals, such as speech, music and image, may use the same method, there existing no need for specialities. First, $s[\cdot]$ is traversed using a bubble-sorting [38] procedure which holds a logarithmic order of time complexity [3] to arrange its elements in ascending order, consequently favouring the computation of the number of distinct symbols and their probabilities. Any other sorting strategy might be used as well.

Algorithm 1 Procedure to calculate H from the input signal $s[\cdot]$ of length M.

BEGINNING
INPUT: ensure that $s[\cdot]$, the original signal under analysis of length M and indexed from 0 to $M-1$, is available ;
STEP 1: rearrange $s[\cdot]$ in ascending order using your preferable method, such as Bubble Sort [38] ;
STEP 2: optionally, traverse $s[\cdot]$ adjusting its samples according to the quantisation defined (D) ;
STEP 3: calculate the probability p_i of each distinct element in $s[\cdot]$;
STEP 4: once the basis β is defined, calculate the entropy of $s[\cdot]$, i.e., $H(s[\cdot]) = -\sum_k p_i \cdot log_\beta(p_i)$;
OUTPUT: $H(s[\cdot])$ is provided ;
END.

The second step of the algorithm is peculiar, offering the possibility to re-quantise $s[\cdot]$. Recalling that the quantisation of a digital signal reflects the step-size adopted for dividing its amplitude axis during digitalisation [55], re-quantisation may be particularly interesting in practice whenever the highest possible precision is not required. Thus, this process avoids close values of amplitude, which might be distinct due to noise or artefacts, to inconveniently increase the number of different symbols, hindering the analysis. The sorted vector $s[\cdot] = \{-1000,\ 80,\ 1499.98,\ 1500.01,\ 1500.002,\ 1500.03,\ 12000,\ 32000\}$ exemplifies such an issue. Considering the overall signal amplitude, i.e., $32,000 - (-1000) = 33,000$, for most of the practical applications the four values close to 1500 could be adjusted to become an unique one. Duly, if $s[\cdot]$ is not re-quantized, those values will be interpreted as distinct symbols, disturbing the expected value for H in practice.

The third and fourth steps, respectively referring to the calculation of the required probabilities and the entropy, take place as soon as the sorting and optional re-quantisation are complete. Algorithm 2 contains the complete source-code in C/C++ programming language [71]. It implements bubble-sort with a *for* loop inside a *do-while* repetition which continues until a certain traverse of $s[\cdot]$ does not require any of its neighbouring elements to be exchanged, implying that the vector is sorted as intended. The differences between variables c and p are strategically used to account how many identical, or close in case of re-quantisation, symbols exist; all of which surely grouped together due to the previous sorting.

Although the preceding description adopted to calculate H concerns only 1D inputs for which the M-sample long signal $s[\cdot]$ is taken into account, the procedures are easily extended to 2D inputs. In order to do so, the matrix $m[\cdot][\cdot]$ with N rows and M columns is considered as being the input signal. Coherently with the 1D case, the positions of the symbols in $m[\cdot][\cdot]$ do not matter, and so, $H(m[\cdot][\cdot])$ is obtained by simply converting the matrix into a $(N \cdot M)$-sample long vector that contains all the original elements of $m[\cdot][\cdot]$ and then calculating the entropy of that vector. Algorithm 3 shows the complete source-code in C/C++ that

Algorithm 2 C/C++ source-code to calculate H from the input signal $s[\cdot]$ of length M.

```
double entropy_1D(double* a,int M)
{
double* s=new double[M]; //auxilia array to avoid changes in the original signal
for(int i = 0; i < M; i++)
    s[i]=a[i];
//bubble sort algorithm begins
double xch; //exchange auxilia variable
int there_were_an_exchange;
do
    {
    there_were_an_exchange=0; //no exchange in the current traverse
    for(int i = 0; i < M - 1; i++)
        if(s[i] > s[i + 1])
            {
            xch=s[i];
            s[i]=s[i + 1];
            s[i + 1]=xch;
            there_were_an_exchange=1; //flag to signalize an exchange
            }
    }while(there_were_an_exchange); //no exchange in the current traverse implies that s[·] is arranged in ascending order
//bubble sort is over. Now, re-quantisation takes place
int D = ; //Define here the quantisation, just in case. Comment this line for no re-quantisation.
//D corresponds to the number of divisions chosen for the re-quantised amplitude axis.
for(int i = 0; i < M; i++) // Comment this line for no re-quantisation.
    s[i]=((int)(s[i]/((double)(D))))*D; // Comment this line for no re-quantisation.
//re-quantisation is over. Now, H is calculated.
double H = 0;
int p = 0;
int c = 1;
do
    {
    while((s[p] == s[c])&&(c < M))
        c++;
    H-= ((c - p)/((double)(M))) * log2((c - p)/((double)(M)));
    p = c;
    c++;
    }while(c <= M);
delete[] s;
return(H);
}
```

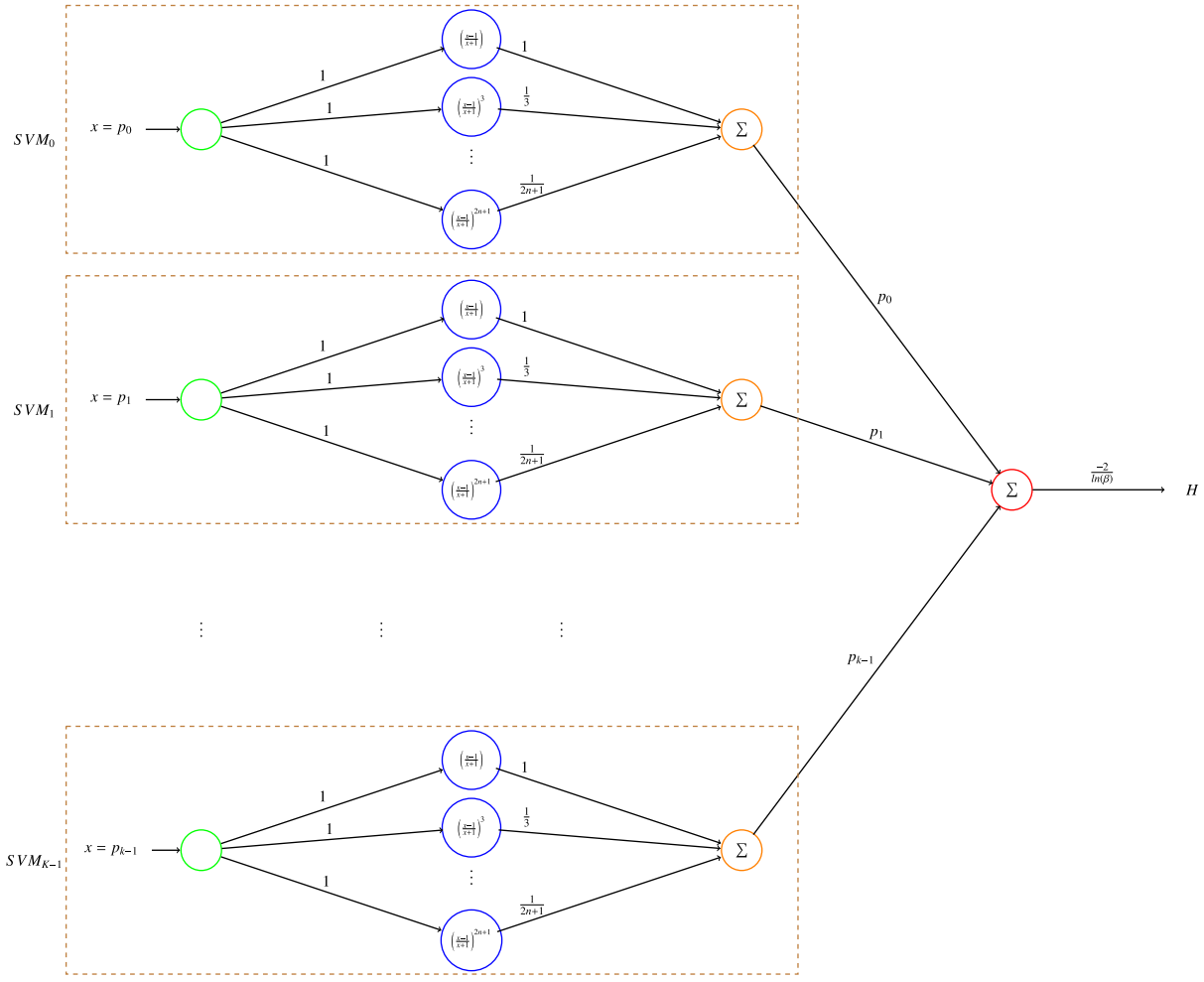


Fig. 3. H is the weighted sum of K isolated partially pre-tuned SVMs, forming a deep structure. Theoretically, the number of hidden elements in each SVM is infinity.

Algorithm 3 C/C++ source-code to calculate H from the input matrix $m[\cdot]$ with N rows and M columns.

```
double entropy_2D(double** m,int in_row,int in_col,int
N,int M)
{
double* s=new double[N * M];
for(int i = in_row; i < N; i++)
    for(int j = in_col; j < M; j++)
        s[(j − in_col) + ((i − in_row) * (M))] = m[i][j];
return(entropy_1D(s, N * M));
}
```

implements the conversion and, then, taking advantage of its 1D version, performs the calculation.

2.2. Entropy calculation: a deep neural network outcome

In this subsection, an interesting point-of-view is offered to the readers. The use of logarithm property $log_\beta(x) = \frac{ln(x)}{ln(\beta)}$ allows the equivalence $H = -\sum_{i=0}^{K-1} p_i \cdot log_\beta(p_i) = -\sum_{i=0}^{K-1} p_i \cdot \frac{1}{ln(\beta)} \cdot ln(p_i) = \sum_{i=0}^{K-1} p_i \cdot ln(p_i)(-\frac{1}{ln(\beta)})$. Additionally, the expansion of $ln(x)$ using Taylor's series [35] allows to rewrite H so that

$$H = \sum_{i=0}^{K-1} p_i \cdot ln(p_i) \cdot \left(-\frac{1}{ln(\beta)}\right)$$

$$= \sum_{i=0}^{K-1} p_i \cdot \left(2\sum_{n=1}^{\infty} \frac{1}{2n+1}\left(\frac{p_i-1}{p_i+1}\right)^{2n+1}\right) \cdot \left(-\frac{1}{ln(\beta)}\right)$$

$$= \left(-\frac{2}{ln(\beta)}\right) \cdot \sum_{i=0}^{K-1} p_i \cdot \left(\sum_{n=1}^{\infty} \frac{1}{2n+1}\left(\frac{p_i-1}{p_i+1}\right)^{2n+1}\right)$$

$$= \underbrace{\left(-\frac{2}{ln(\beta)}\right)}_{\text{constant value}} \cdot \sum_{i=0}^{K-1} p_i \cdot \left(\left(\frac{p_i-1}{p_i+1}\right) + \frac{1}{3}\left(\frac{p_i-1}{p_i+1}\right)^3 + \frac{1}{5}\left(\frac{p_i-1}{p_i+1}\right)^5 + \cdots\right).$$

Fig. 3, which graphically represents the proposed formulation, corroborates the fact that, independently of the multiplication by the constant $\frac{-2}{ln(\beta)}$, H is the weighted sum of K scalars, with p_0, p_1, ..., p_{K-1} as the corresponding and pre-defined weights. Particularly, each scalar corresponds to the output of an isolated support vector machine (SVM) [18,32]. For each SVM, the initial element, the hidden elements and the output element are, respectively, passive, active non-linear polynomial-like kernels [32] and active linear. On one hand, passive elements just forward their inputs to the corresponding outputs. On the other, active elements process their inputs, modifying them accordingly.

The inspection of Fig. 3 clearly favours the stating that the complete structure is composed of four layers, from which only the first is passive. Notably and according to recent results [22], three

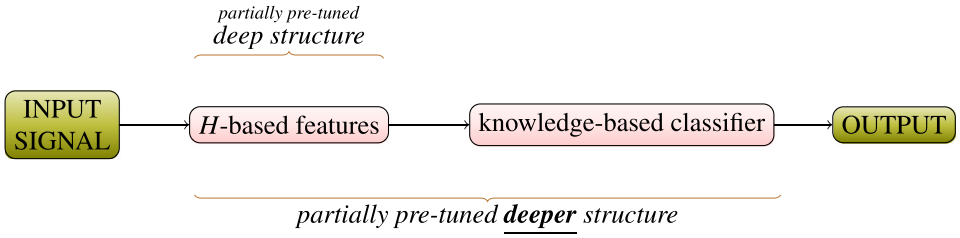


Fig. 4. Combining features based on H with a knowledge-based classifier.

active layers are enough to characterise the proposed architecture as being that of a deep network, i.e., H by itself consists of the outcome of an information fusion procedure from K distinct and particularly tuned SVMs.

Lastly, the readers know that features such as H are usually employed as input to pattern-matching or knowledge-based classifiers. Specifically when the latter type of classifier is adopted, as shown in Fig. 4, the entire algorithm matches that of a depper network and, therefore, has a considerable potential to solve classification tasks. Hopefully, this particular point-of-view motivates both the DSP and PR communities to use H. Once both the algorithms designed to calculate H are presented and complemented with my particular point-of-view which characterises H as the outcome of a partially pre-tuned deep neural structure, the readers are ready to understand both methods I propose, i.e., C_1 and C_2, as detailed in the next subsections. On one hand, C_1 is inspired both by A_1 and B_1, respectively from [23] and [24]. Accordingly, C_2 draws inputs from both A_2 and B_2, which are described in the same references. On the other hand, the procedure employed to define A_3 [23] and B_3 [24], which is based on the analysis of cumulative amounts of energy and ZCRs, does **not** admit any extension to be used with H. Particularly, considering a signal frame of length L_1 and another of length L_2, both starting at the same point and with $L_2 > L_1$, then $H(L_2)$ is not necessarily higher than $H(L_1)$, preventing the use of that strategy. This is the reason why such a method, that would possibly be called C_3 in analogy with A_3 and B_3, is not defined in this article.

2.3. The proposed method C_1

Similarly to A_1 [23] and B_1 [24], C_1 is the simplest method proposed in this work. For 1D input signal $s[\cdot]$ of length M, it consists of a sliding rectangular window, w, of length L traversing the signal so that, for each placement, H is calculated over that position. Each subsequent positioning overlaps in V% the previous one, being the over-content at the end, that is insufficient to cover a window, disposed. The restrictions $(2 \leq L \leq M)$ and $(0 \leq V < 100)$ necessarily apply.

As I explained in [23]-pp.2 for energy, entropy-based handcrafted FE also requires $s[\cdot]$ to be converted into a *feature vector*, $f[\cdot]$, of length $T = \lfloor \frac{(100 \cdot M) - (L \cdot V)}{(100 - V) \cdot L} \rfloor$, being $\lfloor \cdot \rfloor$ the floor operator. In this case, each f_k, $(0 \leqslant k \leqslant T-1)$, corresponds to H computed over the kth position of the window w. As it is also documented in [23]-pp.2, $f[\cdot]$ requires normalisation prior to its use as an input for a classifier, with the relative entropy [66] as the proper method. It allows H-based evaluations by comparisons, particularly forcing the highest H in $f[\cdot]$ to be 1 and adjusting the remaining, proportionally, within the range $(0 \sim 1)$. The procedure consists of dividing each individual value in $f[\cdot]$ by the highest unnormalised H contained in it, which is the one computed over the window placement named w_h, i.e.,

$$f_r \leftarrow \frac{f_r}{H(w_h)}, \qquad (0 \leqslant r \leqslant T-1).$$

The proposed approach can be easily extended to a 2D signal, $m[\cdot][\cdot]$, with N rows and M columns, which represent, respectively, the height and width of the corresponding image, so that the feature vector, $f[\cdot]$, contains not only T, but $T \cdot P$ elements, where $P = \lfloor \frac{(100 \cdot N) - (L \cdot V)}{(100 - V) \cdot L} \rfloor$. During the analysis, $m[\cdot][\cdot]$ is traversed along the horizontal orientation based on T placements of the square window w of side L, being $L < M$ and $L < N$. Then, the process is repeated for each one of the P shifts along the vertical orientation.

Identically as in A_1 [23] and B_1 [24], for both 1D and 2D signals, respectively, C_1 is only capable of generating a T, or a $T \cdot P$, sample-long vector $f[\cdot]$ if the value of L is subjected to the value of M, or M and N. Thus, the value of L intrinsically depends on the length of the input 1D signal $s[\cdot]$, or the dimensions of the input 2D matrix $m[\cdot][\cdot]$, creating a disadvantage: irregular, temporal or spatial analysis. Oppositely, the advantage is that a few sequential elements of $f[\cdot]$, obtained by predefining L, T and P, allow the detection of some particular event in the 1D or 2D signal under analysis.

Algorithms 4 and 5 contain the C/C++ implementations for C_1 in 1D and 2D, respectively. Additionally, Figs. 5 and 6 illustrate the way C_1 works for both 1D and 2D inputs, respectively.

Algorithm 4 C/C++ source-code for C_1 in 1D.

```
// . . .
// ensure that s[·], with length M, is available as input
int L = /* the desired positive value, not higher than M */;
int V = /* the desired positive value, lower than 100 */;
int T = (int)((100 * M - L * V)/((100 - V)*L));
double highest_H = 0;
double *f = new double[T]; // dynamic vector declaration
for(int k = 0; k < T; k++)
    {
     f[k] = entropy_1D(&s[k * ((int)(((100 - V)/100.0) *
L))], L);
    if (f[k] > highest_H)
        highest_H = f[k];
    }
for(int k = 0; k < T; k++)
    f[k]/ = highest_H;
// at this point, the feature vector, f[·], is ready
// . . .
```

2.4. The proposed method C_2

C_2, analogously to C_1, focuses on windowing $s[\cdot]$, or $m[\cdot][\cdot]$, albeit with differences. In the former method, no overlaps take place and the window length for 1D, or the rectangle sizes for 2D, changes according to the granularity adopted. Thus, as in A_2 [23] and B_2 [24], C_2 provides different levels of resolution to analyse the input signal, being this the main incentive to choose it. Particularly, the feature vector generated from C_2 is defined as being the concatenation of Q sub-vectors of different dimensions, i.e., $f[\cdot] = \{\xi_1[\cdot]\} \cup \{\xi_2[\cdot]\} \cup \{\xi_3[\cdot]\} \cup \ldots \cup \{\xi_Q[\cdot]\}$.

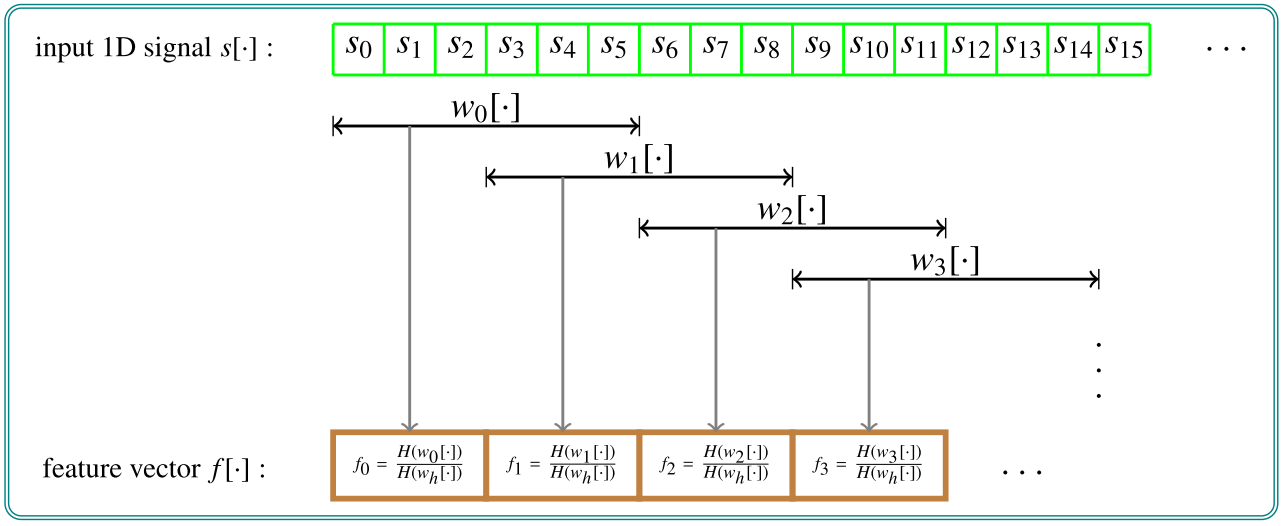


Fig. 5. C_1 applied to an 1D input signal $s[\cdot]$. Since A_1, described in [23], inspires C_1, this diagram is inherited from that article, in which the only differences are the components of the feature vector $f[\cdot]$.

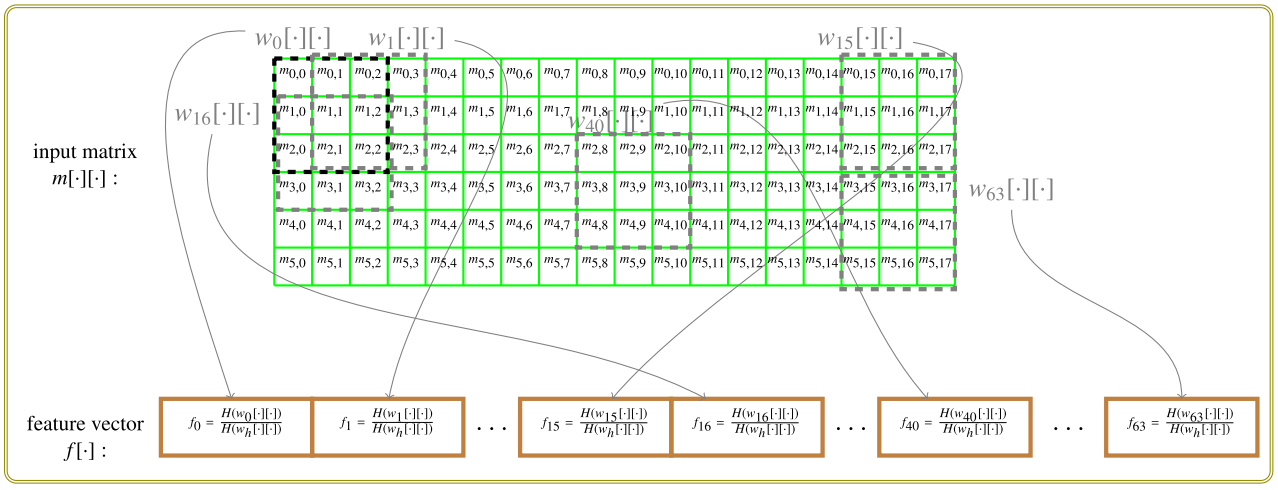


Fig. 6. C_1 applied to a 2D input signal $m[\cdot][\cdot]$. Once again, since A_1, described in [23], inspires C_1, this diagram is inherited from that article, in which the only differences are the components of the feature vector $f[\cdot]$.

For 1D, each sub-vector $\xi_i[\cdot]$, $(1 \le i \le Q)$, results from dividing $s[\cdot]$ in T non-overlapping sequential windows, being T a prime number in order to avoid one sub-vector to be a linear combination of another, and then calculating the corresponding normalised entropies, i.e.,

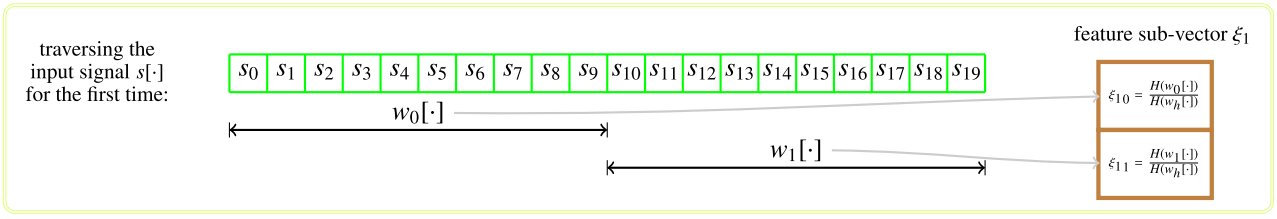
- subvector $\xi_1[\cdot]$ is obtained by letting $L = \lfloor \frac{M}{2} \rfloor$, which that its size is $T = 2$;
- subvector $\xi_2[\cdot]$ is obtained by letting $L = \lfloor \frac{M}{3} \rfloor$, which that its size is $T = 3$;
- subvector $\xi_3[\cdot]$ is obtained by letting $L = \lfloor \frac{M}{5} \rfloor$, which that its size is $T = 5$;
- . . .
- subvector $\xi_Q[\cdot]$ is obtained by letting $L = \lfloor \frac{M}{X} \rfloor$, which that its size is $T = X$.

In order to match the system engineer's objectives, Q may vary.

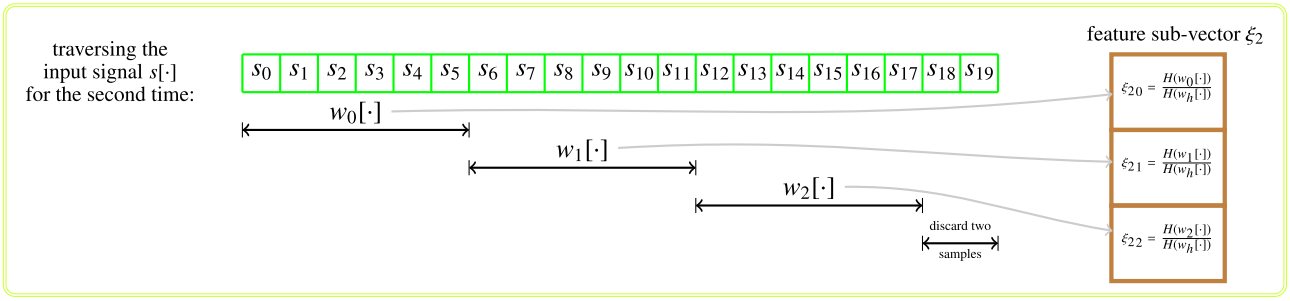
In regard to the 2D case, each sub-vector is produced by framing $m[\cdot][\cdot]$ using $T \cdot P$ non-overlapping rectangles, being $T = P$ prime numbers, and then calculating the corresponding normalised entropies, i.e.,

- subvector $\xi_1[\cdot]$ is determined by letting $L = \lfloor \frac{M}{2} \rfloor$ to obtain $T = 2$, and then by letting $L = \lfloor \frac{N}{2} \rfloor$ to obtain $P = 2$ so that $m[\cdot][\cdot]$ is divided in $T \cdot P = 2 \cdot 2 = 4$ non-overlapping rectangles;
- idem to subvector $\xi_2[\cdot]$, obtained by letting $L = \lfloor \frac{M}{3} \rfloor$ and then $L = \lfloor \frac{N}{3} \rfloor$, which that $T = 3$ and $P = 3$, respectively, originating $T \cdot P = 3 \cdot 3 = 9$ non-overlapping rectangles;
- idem to subvector $\xi_3[\cdot]$, obtained by letting $L = \lfloor \frac{M}{5} \rfloor$ and then $L = \lfloor \frac{N}{5} \rfloor$, which that $T = 5$ and $P = 5$, respectively, originating $T \cdot P = 5 \cdot 5 = 25$ non-overlapping rectangles;
- . . .
- idem to subvector $\xi_Q[\cdot]$, obtained by letting $L = \lfloor \frac{M}{X} \rfloor$ and then $L = \lfloor \frac{N}{X} \rfloor$, which that $T = X$ and $P = X$, respectively, originating $T \cdot P = X \cdot X = X^2$ non-overlapping rectangles.

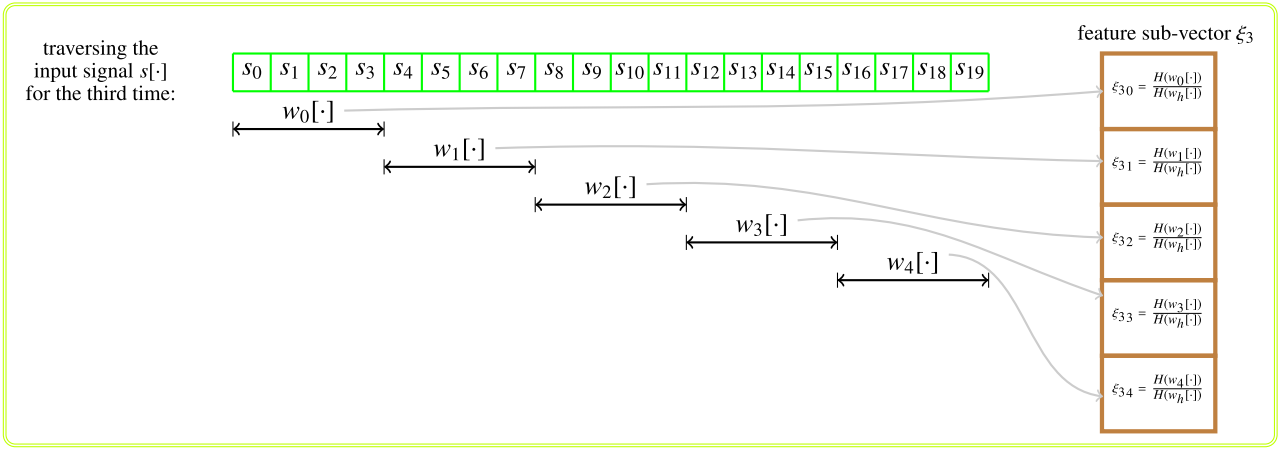
Figs. 7 and 8 illustrate, respectively, the sliding window for 1D and the sliding rectangle for 2D over a hypothetical signal. Complementarily, Algorithms 6 and 7 contain the respective implementations for 1D and 2D.



(a) windowing the input signal to form the first sub-vector



(b) windowing the input signal to form the second sub-vector



(c) windowing the input signal to form the third sub-vector

Fig. 7. 1D example for C_2 assuming $Q = 3$: (a) sliding window, with length $L = \lfloor \frac{M}{2} \rfloor = \lfloor \frac{20}{2} \rfloor = 10$ traversing $s[\cdot]$ in order to compose $\xi_1[\cdot]$; (b) sliding window with length $L = \lfloor \frac{M}{3} \rfloor = \lfloor \frac{20}{3} \rfloor = 6$ traversing $s[\cdot]$ in order to compose $\xi_2[\cdot]$; (c) sliding window with length $L = \lfloor \frac{M}{5} \rfloor = \lfloor \frac{20}{5} \rfloor = 4$ traversing $s[\cdot]$ in order to compose $\xi_3[\cdot]$. The window positions do not overlap and the symbols w_i indicate the ith window position, for $i = 0, 1, 2, \ldots, T-1$. Since A_2, described in [23], inspires C_2, this diagram is inherited from that article, in which the only differences are the components of the feature vector $f[\cdot]$.

3. Numerical examples

In order to clarify the proposed approaches, one numerical example follows for each case: methods C_1 and C_2, both in 1D and 2D, based on hypothetical data.

3.1. Numerical example for C_1 in 1D

Problem statement: Let $s[\cdot] = \{1, 3, 2, 4, 4, 5\}$, implying in $M = 6$, and $L = 4$ be the window length, with overlaps of $V = 50\%$. Obtain the feature vector, $f[\cdot]$, according to the method C_1 using the base $\beta = 2$.

Solution: The feature vector, which has length $T = \lfloor \frac{(100 \cdot M) - (L \cdot V)}{(100 - V) \cdot L} \rfloor = \lfloor \frac{(100 \cdot 6) - (4 \cdot 50)}{(100 - 50) \cdot 4} \rfloor = 2$, is obtained as follows:

- $w_0[\cdot]$, which covers the sub-signal {1, 3, 2, 4}, contains four distinct elements implying that $f_0 = -\sum_{i=0}^{3} p_i \cdot log_2(p_i) = (\frac{1}{4} log_2(4) + \frac{1}{4} log_2(4) + \frac{1}{4} log_2(4) + \frac{1}{4} log_2(4)) = (\frac{1}{2} + \frac{1}{2} + \frac{1}{2} + \frac{1}{2}) = 2$;
- $w_1[\cdot]$, which covers the sub-signal {2, 4, 4, 5}, contains three distinct elements implying that $f_0 = -\sum_{i=0}^{2} p_i \cdot log_2(p_i) = (\frac{1}{4} log_2(4) + \frac{2}{4} log_2(\frac{4}{2}) + \frac{1}{4} log_2(4)) = (\frac{1}{2} + \frac{1}{2} + \frac{1}{2}) = 1.5$;

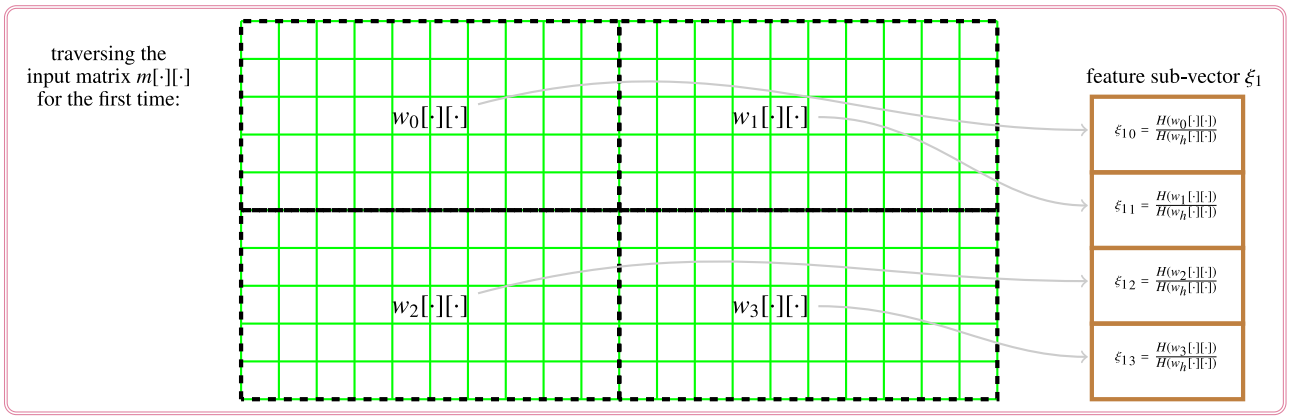
In order to normalise the feature vector, obtaining the relative entropies, each component of $f[\cdot]$ is divided by its highest. Thus, it becomes $\{\frac{2}{2}, \frac{1.5}{2}\} = \{1, 0.75\}$.

3.2. Numerical example for C_1 in 2D

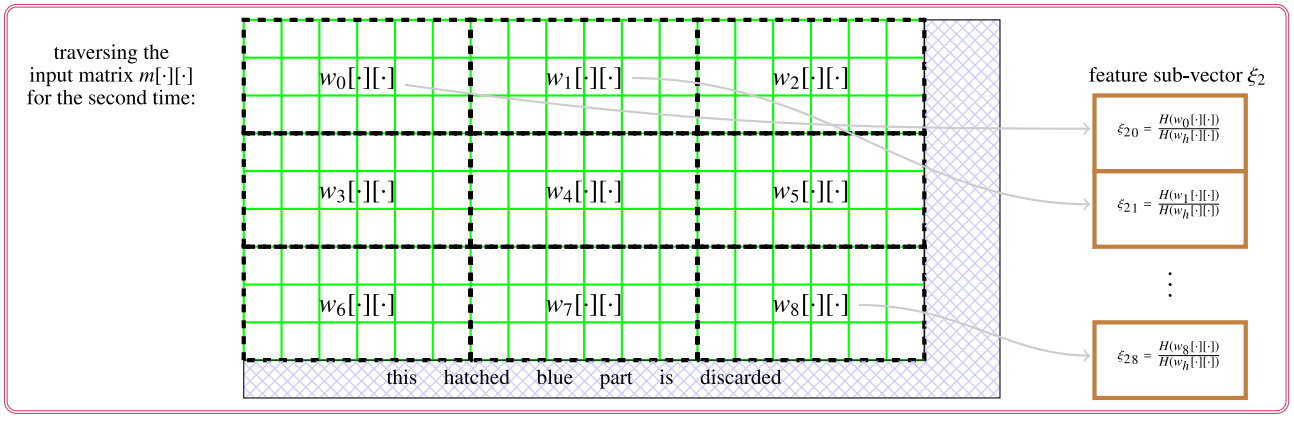
Problem statement: Let $m[\cdot][\cdot] = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 2 & 4 & 6 \\ 7 & 8 & 8 & 10 \end{pmatrix}$, implying in $N = 3$ and $M = 4$. Assume that the square window has size $L = 2$ with overlaps of $V = 50\%$. Obtain the feature vector, $f[\cdot]$, following method C_1 using $\beta = 2$.

Solution: The feature vector with length $T \cdot P = \lfloor \frac{(100 \cdot M) - (L \cdot V)}{(100 - V) \cdot L} \rfloor \cdot \lfloor \frac{(100 \cdot N) - (L \cdot V)}{(100 - V) \cdot L} \rfloor = \lfloor \frac{(100 \cdot 4) - (2 \cdot 50)}{(100 - 50) \cdot 2} \rfloor \cdot \lfloor \frac{(100 \cdot 3) - (2 \cdot 50)}{(100 - 50) \cdot 2} \rfloor = 3 \cdot 2 = 6$ is obtained as follows:

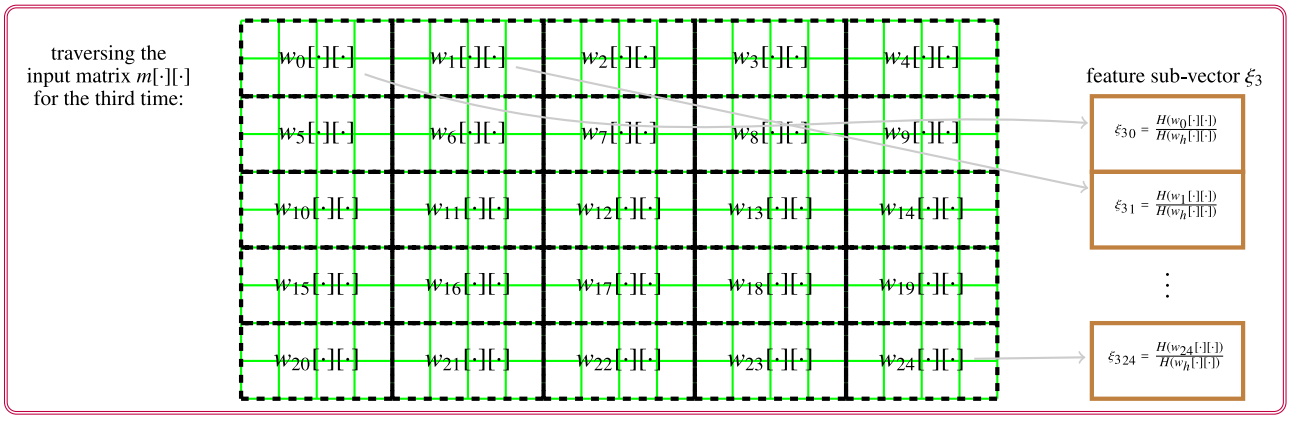
- $w_0[\cdot][\cdot]$ covers the sub-matrix $\begin{pmatrix} 1 & 2 \\ 4 & 2 \end{pmatrix}$, which contains three distinct elements implying that $f_0 = -\sum_{i=0}^{2} p_i \cdot log_2(p_i) = (\frac{1}{4} log_2(4) + \frac{2}{4} log_2(\frac{4}{2}) + \frac{1}{4} log_2(4)) = (\frac{1}{2} + \frac{1}{2} + \frac{1}{2}) = 1.5$;



(a) windowing the input signal to form the first sub-vector



(b) windowing the input signal to form the second sub-vector



(c) windowing the input signal to form the third sub-vector

Fig. 8. 2D example for C_2 assuming $Q = 3$ subvectors: [above] sliding square with length $\{\lfloor \frac{M}{2} \rfloor x \lfloor \frac{N}{2} \rfloor\} = \{\lfloor \frac{10}{2} \rfloor x \lfloor \frac{20}{2} \rfloor\} = 5x10$ traversing $m[\cdot][\cdot]$ in order to compose $\xi_1[\cdot]$; [middle] sliding square with length $\{\lfloor \frac{M}{3} \rfloor x \lfloor \frac{N}{3} \rfloor\} = \{\lfloor \frac{10}{3} \rfloor x \lfloor \frac{20}{3} \rfloor\} = 3x6$ traversing $m[\cdot][\cdot]$ in order to compose $\xi_2[\cdot]$; [below] sliding square with length $\{\lfloor \frac{M}{5} \rfloor x \lfloor \frac{N}{5} \rfloor\} = \{\lfloor \frac{10}{5} \rfloor x \lfloor \frac{20}{5} \rfloor\} = 2x4$ traversing $m[\cdot][\cdot]$ in order to compose $\xi_3[\cdot]$. Again, w_i indicates the ith window position, for $i = 0, 1, 2, \ldots, (T \cdot P) - 1$, with no overlap. Dashed squares represent the sliding window in all possible positions. Once again, since A_2, described in [23], inspires C_2, this diagram is inherited from that article, in which the only differences are the components of the feature vector $f[\cdot]$.

- $w_1[\,\cdot\,][\,\cdot\,]$ covers the sub-matrix $\begin{pmatrix} 2 & 3 \\ 2 & 4 \end{pmatrix}$, which contains three distinct elements implying the same previous result, i.e., 1.5;
- $w_2[\,\cdot\,][\,\cdot\,]$ covers the sub-matrix $\begin{pmatrix} 3 & 4 \\ 4 & 6 \end{pmatrix}$, which also contains three distinct elements implying the same previous result, i.e., 1.5;
- $w_3[\,\cdot\,][\,\cdot\,]$ covers the sub-matrix $\begin{pmatrix} 4 & 2 \\ 7 & 8 \end{pmatrix}$, which contains four distinct elements implying that $f_0 = -\sum_{i=0}^{3} p_i \cdot log_2(p_i) = (\frac{1}{4}log_2(4) + \frac{1}{4}log_2(4) + \frac{1}{4}log_2(4) + \frac{1}{4}log_2(4)) = (\frac{1}{2} + \frac{1}{2} + \frac{1}{2} + \frac{1}{2}) = 2$;
- $w_4[\,\cdot\,][\,\cdot\,]$ covers the sub-matrix $\begin{pmatrix} 2 & 4 \\ 8 & 8 \end{pmatrix}$, which contains three distinct elements implying, as shown above, in the result 1.5;
- $w_5[\,\cdot\,][\,\cdot\,]$ covers the sub-matrix $\begin{pmatrix} 4 & 6 \\ 8 & 10 \end{pmatrix}$, which also contains four distinct elements implying in the result 2;

Algorithm 5 C/C++ source-code for C_1 in 2D.

```
// . . .
// ensure that m[·][·], with height N and width M, is available as input
int L = /* the desired positive value, not higher than the higher between M and N */;
int V = /* the desired positive value, lower than 100 */;
int T = (int)((100 * M − L * V)/((100 − V)*L));
int P = (int)((100 * N − L * V)/((100 − V)*L));
double highest_H = 0;
double *f = new double[T * P]; // dynamic vector declaration
int k = 0;
for(int i = 0; i < T * L; i+ = L)
      for(int j = 0; j < P * L; j+ = L)
            {
            f[k] = entropy_2D(m, i, j, L, L);
            if (f[k] > highest_H)
                  highest_H = f[k];
             k++;
            }
for(int i = 0; i < T * P; i++)
      f[i]/ = highest_H;
// at this point, the feature vector, f[·], is ready
// . . .
```

Algorithm 6 Fragment of C++ code for method C_2 in 1D.

```
// . . .
// ensure that s[·], with length M, is available as input
int L; // window length
double highest_H;
int X[] = {2, 3, 5, 7, 11, 13, 17}; /* vector containing the prime numbers of interest. It can be changed according to the experiment */
int total_size_of_f = 0;
for(int i = 0; i <(int)(sizeof(X)/sizeof(int));i++) // number of elements in X[·]
      total_size_of_f+=X[i];
double *f = new double[total_size_of_f]; /* The total size of f[·] is the sum of the elements in X[·] */
int jump = 0; // helps to control the correct positions to write in f[·]
for(int j = 0; j <(int)(sizeof(X)/sizeof(int)) ; j++)
      {
      highest_H = 0;
      for(int k = 0; k < X[j]; k++)
            {
            L = (int)(M/X[j]);
            f[jump + k] =entropy_1D(&s[K * L], L);
            if(f[jump + k] > highest_H)
                  highest_H = f[jump + k];
            }
      for(int k = 0; k < X[j]; k++)
            f[jump + k]/ = highest_H;
       jump+ = X[j];
      }
// at this point, the feature vector, f[·], is ready.
// . . .
```

Algorithm 7 Fragment of C++ code for method C_2 in 2D.

```
// . . .
// ensure that m[·][·], with height N and width M, is available as input
int L1, L2;
double highest_H;
int X[] = {2, 3, 5, 7, 11, 13, 17}; /* vector containing the prime numbers of interest. It can be changed according to the experiment */
int total_size_of_f = 0;
for(int i = 0; i <(int)(sizeof(X)/sizeof(int));i++) // number of elements in X[·]
      total_size_of_f+=pow(X[i], 2);
double *f = new double[total_size_of_f]; /* The total size of f[·] is the sum of the squares of the elements in X[·] */
int jump = 0; // helps to control the correct positions to write in f[·]
for(int i = 0;i <total_size_of_f;i++)
      f[i] = 0;
int w = 0;
for(int k = 0;k <(int)(sizeof(X)/sizeof(int));k++)
      {
      w = 0;
      highest_H = 0;
      L1 =(int)(N/X[k]);
      L2 =(int)(M/X[k]);
      for(int i = 0;i < N − L1;i+ = L1)
            for(int j = 0;j < M − L2;j+ = L2)
                  {
                  f[jump+w] = entropy_2D(m, i, j, L1, L2);
                  if(f[jump+w]> highest_H)
                        highest_H =f[jump+w];
                  w = 0;
                  }
      for(int i =jump;i <jump + pow(X[k], 2);i++)
            f[i]/ = highest_H;
      jump+=pow(X[k], 2);
      }
// at this point, the feature vector, f[·], is ready.
// . . .
```

Accordingly, in order to normalise the feature vector to get the relative entropies, each component of $f[\cdot]$ is divided by its highest, i.e., 2, thus becoming $\{\frac{1.5}{2}, \frac{1.5}{2}, \frac{1.5}{2}, \frac{2}{2}, \frac{1.5}{2}, \frac{2}{2}\} = \{0.75, 0.75, 0.75, 1, 0.75, 1\}$.

3.3. Numerical example for C_2 in 1D

Problem statement: Let $s[\cdot] = \{1, 2, 4, 6, 6, 6, 6, 5, 3, 1\}$, implying in $M = 10$. Assuming that $Q = 2$, with no overlaps between window positions, obtain the feature vector, $f[\cdot]$, following the method C_2 and using $\beta = 2$.

Solution: The feature vector is composed by the concatenation of $Q = 2$ sub-vectors, i.e., $f[\cdot] = \{\xi_1[\cdot]\} \cup \{\xi_2[\cdot]\}\}$, which are obtained as follows:

- the first subvector, $\xi_1[\cdot]$, arises from two non-overlapping windows, $w_0[\cdot] = \{1, 2, 4, 6, 6\}$ and $w_1[\cdot] = \{6, 6, 5, 3, 1\}$, which are positioned over $s[\cdot]$. Both the windows contain four different elements, among five. The corresponding results are:
  $\xi_{10} = \sum_{i=0}^{3} p_i \cdot log_2(\frac{1}{p_i}) = (\frac{1}{5} \cdot log_2(5)) + (\frac{1}{5} \cdot log_2(5)) + (\frac{1}{5} \cdot log_2(5)) + (\frac{2}{5} \cdot log_2(\frac{5}{2})) = \frac{2.3220}{5} + \frac{2.3220}{5} + \frac{2.3220}{5} + \frac{2 \cdot 1.3220}{5} = 1.922$
  and
  $\xi_{11} = \sum_{i=0}^{3} p_i \cdot log_2(\frac{1}{p_i}) = (\frac{2}{5} \cdot log_2(\frac{5}{2})) + (\frac{1}{5} \cdot log_2(5)) + (\frac{1}{5} \cdot log_2(5)) + (\frac{1}{5} \cdot log_2(5)) = \frac{2 \cdot 1.3220}{5} + \frac{2.3220}{5} + \frac{2.3220}{5} + \frac{2.3220}{5} = 1.922$

- the second subvector, $\xi_2[\cdot]$, comes from three non-overlapping windows, $w_0[\cdot] = \{1, 2, 4\}$, $w_1[\cdot] = \{6, 6, 6\}$ and $w_2[\cdot] = \{6, 5, 3\}$, which are positioned over $s[\cdot]$, discarding its last element, i.e., the amplitude 1. The three windows contain three, one and three different elements, among three. The corresponding results are:
  $\xi_{20} = \sum_{i=0}^{2} p_i \cdot log_2(\frac{1}{p_i}) = (\frac{1}{3} \cdot log_2(3)) + (\frac{1}{3} \cdot log_2(3)) + (\frac{1}{3} \cdot log_2(3)) = \frac{1.5850}{3} + \frac{1.5850}{3} + \frac{1.5850}{3} = 1.5850$
  ;
  $\xi_{21} = \sum_{i=0}^{0} p_i \cdot log_2(\frac{1}{p_i}) = (1 \cdot log_2(1)) = 0$
  and
  $\xi_{22} = \sum_{i=0}^{2} p_i \cdot log_2(\frac{1}{p_i}) = (\frac{1}{3} \cdot log_2(3)) + (\frac{1}{3} \cdot log_2(3)) + (\frac{1}{3} \cdot log_2(3)) = \frac{1.5850}{3} + \frac{1.5850}{3} + \frac{1.5850}{3} = 1.5850$

The concatenation of both sub-vectors produce $f[\cdot] = \{1.922, 1.922, 1.5850, 0, 1.5850\}$. Now, each sub-vector is normalised separately. Expressly, each component of ξ_1 in $f[\cdot]$ is divided by 1.922 and each component of ξ_2 in $f[\cdot]$ keeps divided by 1.5850, i.e., the highest values in each group. Thus, $f[\cdot]$ becomes $\{\frac{1.922}{1.922}, \frac{1.922}{1.922}, \frac{1.5850}{1.5850}, \frac{0}{1.5850}, \frac{1.5850}{1.5850}\} = \{1, 1, 1, 0, 1\}$.

3.4. Numerical example for C_2 in 2D

Problem statement: Let $m[\cdot][\cdot] = \begin{pmatrix} 0 & 1 & 2 & 2 \\ 4 & 5 & 6 & 7 \\ 8 & 8 & 10 & 12 \\ 3 & 0 & 1 & 2 \end{pmatrix}$, implying that $N = 4$ and $M = 4$. Assume that $Q = 2$ with no overlaps between windows. Obtain the feature vector, $f[\cdot]$, following method C_2 using $\beta = 2$.

Solution: The feature vector is composed by the concatenation of $Q = 2$ sub-vectors, i.e., $f[\cdot] = \{\xi_1[\cdot]\} \cup \{\xi_2[\cdot]\}$. They are obtained as follows:

- for $\xi_1[\cdot]$, a total of $2 \cdot 2 = 4$ non-overlapping windows, $w_0[\cdot][\cdot] = \begin{pmatrix} 0 & 1 \\ 4 & 5 \end{pmatrix}$, $w_1[\cdot][\cdot] = \begin{pmatrix} 2 & 2 \\ 6 & 7 \end{pmatrix}$, $w_2[\cdot][\cdot] = \begin{pmatrix} 8 & 8 \\ 3 & 0 \end{pmatrix}$ and $w_3[\cdot][\cdot] = \begin{pmatrix} 10 & 12 \\ 1 & 2 \end{pmatrix}$, are positioned over $m[\cdot][\cdot]$. The four mini matrices contain four, three, three and four different elements, among four. Thus, the results are:
  $\xi_{10} = \sum_{i=0}^{3} p_i \cdot log_2(\frac{1}{p_i}) = (\frac{1}{4} \cdot log_2(4)) + (\frac{1}{4} \cdot log_2(4)) + (\frac{1}{4} \cdot log_2(4)) + (\frac{1}{4} \cdot log_2(4)) = \frac{1}{2} + \frac{1}{2} + \frac{1}{2} + \frac{1}{2} = 2$
  ;
  $\xi_{11} = \sum_{i=0}^{2} p_i \cdot log_2(\frac{1}{p_i}) = (\frac{2}{4} \cdot log_2(\frac{4}{2})) + (\frac{1}{4} \cdot log_2(4)) + (\frac{1}{4} \cdot log_2(4)) = \frac{1}{2} + \frac{1}{2} + \frac{1}{2} = 1.5$
  ;
  $\xi_{12} = \sum_{i=0}^{2} p_i \cdot log_2(\frac{1}{p_i}) = (\frac{2}{4} \cdot log_2(\frac{4}{2})) + (\frac{1}{4} \cdot log_2(4)) + (\frac{1}{4} \cdot log_2(4)) = \frac{1}{2} + \frac{1}{2} + \frac{1}{2} = 1.5$
  and
  $\xi_{13} = \sum_{i=0}^{3} p_i \cdot log_2(\frac{1}{p_i}) = (\frac{1}{4} \cdot log_2(4)) + (\frac{1}{4} \cdot log_2(4)) + (\frac{1}{4} \cdot log_2(4)) + (\frac{1}{4} \cdot log_2(4)) = \frac{1}{2} + \frac{1}{2} + \frac{1}{2} + \frac{1}{2} = 2.$
- for $\xi_2[\cdot]$, a total of $3 \cdot 3 = 9$ non-overlapping windows, $w_0[\cdot][\cdot] = (0)$, $w_1[\cdot][\cdot] = (1)$, $w_2[\cdot][\cdot] = (2)$, $w_3[\cdot][\cdot] = (4)$, $w_4[\cdot][\cdot] = (5)$, $w_5[\cdot][\cdot] = (6)$, $w_6[\cdot][\cdot] = (8)$, $w_7[\cdot][\cdot] = (8)$ and $w_8[\cdot][\cdot] = (10)$, are positioned over $m[\cdot][\cdot]$, whereas its fourth row and fourth column discarded. Since the symbols are thoroughly predictable due to their uniqueness for all nine mini matrices, the results are $\xi_{20} = 0$; $\xi_{21} = 0$; $\xi_{22} = 0$; $\xi_{23} = 0$; $\xi_{24} = 0$; $\xi_{25} = 0$; $\xi_{26} = 0$; $\xi_{27} = 0$ and $\xi_{28} = 0$.

The concatenation of both sub-vectors produce $f[\cdot] = \{2, 1.5, 1.5, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0\}$. As in the previous example, each sub-vector is normalised separately, i.e., each component of ξ_1 in $f[\cdot]$ is divided by 2; and each component of ξ_2 in $f[\cdot]$ keeps unchangeable because its higest element is 0. Thus, $f[\cdot]$ becomes $\{\frac{2}{2}, \frac{1.5}{2}, \frac{1.5}{2}, \frac{2}{2}, 0, 0, 0, 0, 0, 0, 0, 0, 0\} = \{1, 0.75, 0.75, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0\}$.

4. Example experiments and applications

This section presents two applications for entropy: the first on image synthesis and the second on restricted-vocabulary speech recognition. Unusual in the current literature, they are respectively based on each one of the proposed approaches, i.e., C_1 and C_2.

4.1. Image synthesis

Despite its modesty, this experiment is uncommon and peculiar. Particularly based on the 2D version of C_1 with $\beta = 2$, adopting no re-quantisation and using different possibilities for L, it analyses consecutive parts of the original Elsevier logo, assumed as being the example input, extracting their entropies and then synthesizing new images from that set of values. The synthesized signals correspond just to the $(T \cdot P)$-sample long feature vectors, obtained from the corresponding N x M input raw data but graphically shown as being T x P matrices.

The first five subfigures contained in Fig. 9 allow to intuitively comprehend that, in fact, entropy somehow describes the information contained in the signal from which it is calculated, otherwise the synthesized images would not be similar to its original version, the one shown in subfigure (f). Complementarily and evidently, the more L decreases the more the resolution obtained allows a better synthesized image, as shown in subfigures (a)–(e). In addition to $\beta = 2$, the same procedure was repeated for $\beta = e$, $\beta = \pi$ and $\beta = 10$, resulting in no graphical differences due to the normalisations required for painting the images.

The proposed approach permits an interesting connection with the ideas described in the introductory section of this text, specifically those explained on the basis of Fig. 1. At that time, I used stars, bullets, circles and diamonds as being the input symbols to be analysed with entropy. To carry out such a task, I possibly dilated each original unit of space and then produced a novel set based on the proportional area each normalized element required in relation to the total. Comparatively, the analysis and synthesis procedure I describe in this subsection conveniently “dilates” the value of each image pixel over the window under analysis so that a unique normalized pixel arises, creating a blurred subimage analogous to its original version. Thus, the synthesized data does not equal but reasonably approximates the original 2D signal under analysis.

Concluding, this experiment demonstrates that a conveniently normalized set of entropies, obtained from segments of a particular input signal, is capable of approximating the raw data they were calculated from. It also allows the definition of a peculiar scheme for multiresolution analysis, analogous to that provided on the basis of the Discrete Wavelet Transform [25–28]. Furthermore, since $(T \cdot P) < (N \cdot M)$, an image synthesized on the basis of the proposed approach can be interpreted as one of its possible compressed versions, illustrating the traditional relationship between entropy and compression [13,49,70].

4.2. Restricted-vocabulary speech recognition

On one hand and as widely known, large-vocabulary speech recognisers are usually designed to identify hundreds or thousands of different words [16,31]. On the other, restricted-vocabulary speech recognisers do the same job but for distinguishing among a few words only, being this my intention. Particularly, raw variable-length data in wave format [9] from twenty speakers, including young and elderly people, both male and female, were acquired in such a way that each subject pronounced the Brazilian Portuguese

(a) synthesis using $L = 128$ (b) synthesis using $L = 64$ (c) synthesis using $L = 32$ (d) synthesis using $L = 16$ (e) synthesis using $L = 8$ (f) the original picture.

Fig. 9. Five images synthesized using method C_1 in 2D using $\beta = 2$ and, lastly, the original Elsevier logo.

Table 1
Results for speech recognition when using C_2 with $Q = 2$.

Actual/predicted	"Down"	"Right"	"Left"	"Up"
"Down"	7	3	0	0
"Right"	0	9	0	1
"Left"	0	0	6	4
"Up"	0	1	1	8

Table 2
Results for speech recognition when using C_2 with $Q = 3$.

Actual/predicted	"Down"	"Right"	"Left"	"Up"
"Down"	10	0	0	0
"Right"	0	10	0	0
"Left"	0	0	9	1
"Up"	0	0	0	10

Table 3
Results for speech recognition when using C_2 with $Q = 4$.

Actual/predicted	"Down"	"Right"	"Left"	"Up"
"Down"	10	0	0	0
"Right"	0	10	0	0
"Left"	0	0	10	0
"Up"	0	0	0	10

Table 4
Results for speech recognition when using A_2 with $Q = 2$.

Actual/predicted	"Down"	"Right"	"Left"	"Up"
"Down"	9	0	1	0
"Right"	0	8	0	2
"Left"	0	0	8	2
"Up"	2	3	1	4

Table 5
Results for speech recognition when using A_2 with $Q = 3$.

Actual/predicted	"Down"	"Right"	"Left"	"Up"
"Down"	9	1	0	0
"Right"	0	10	0	0
"Left"	1	0	9	0
"Up"	1	1	1	7

Table 6
Results for speech recognition when using A_2 with $Q = 4$.

Actual/predicted	'down"	"right"	"left"	"up"
"Down"	10	0	0	0
"Right"	0	8	0	2
"Left"	0	0	9	1
"Up"	2	0	0	8

Table 7
Results for speech recognition when using B_2 with $Q = 2$.

Actual/predicted	"Down"	'Right'	"Left"	"Up"
"Down"	10	0	0	0
"Right"	0	10	0	0
"Left"	0	0	9	1
"Up"	0	0	0	10

words "left" (*esquerda*), "right" (*direita*), "up" (*sobe*), and "down" (*desce*), totalising eighty words divided in four classes. Ten words from each class were randomly selected to define the template models, with the remaining ten being used to test the classification accuracy.

All the speech files were digitalised at 16000 Hz, 16-bit, monochannel, for which the original quantization D was kept intact. As in [24], the radiation effects from the speakers' lips were removed as a pre-processing stage known as pre-emphasis. To do so, the first-order finite impulse response (FIR) high-pass filter whose coefficients are $g[\cdot] = \{1, -0.95\}$ was used, via convolution. Assuming that $s[\cdot]$, of length M, is the input speech signal, the procedure is $s_k \leftarrow s_k - (0.95 \cdot s_{k-1})$, for $(1 \leq k < M)$.

The eighty signals have variable length, implying that C_1 is not the ideal method for this application. Oppositely, C_2 generates a T sample-long vector $f[\cdot]$ independent of M, being thus chosen to carry out the task in association with a simple pattern-matching classifier of T inputs and one output. Specifically, the absolute distances from each input testing signal to the forty template models are registered, then, the class which the lowest one belongs to, guides the assignment by means of a numeric label corresponding to the classifier output.

Formatted as confusion matrices, Tables 1–3 show, respectively, the highest accuracies obtained in $\binom{20}{10}^4 = \left(\frac{20!}{10!(20-10)!}\right)^4 = 184756$ holdout cross-validations procedures [41] with $Q = 2$, $Q = 3$ and $Q = 4$, implying that, primarily, the feature vectors contain $2 + 3 = 5$ elements, $2 + 3 + 5 = 10$ elements and $2 + 3 + 5 + 7 = 17$ elements. Aiming at a better accuracy and following traditional procedures used for PR, the first-order derivatives of the feature vectors were used instead the original vector themselves. The respective accuracies were $\frac{8+8+8+8}{40} = 75.00\%$, $\frac{10+10+9+10}{40} = 97.50\%$, and $\frac{10+10+10+10}{40} = 100.00\%$, suggesting that, in this specific case, a more detailed partition is relevant to characterise important information.

In comparison with other features previously explored in [23] and [24], entropy is the only which acts, independently, as a deep network, possibly presenting advantages for speech recognition tasks due to the largely variable nature of such signals [16,31]. To example, numerical results obtained with the adoption of energy based on A_2 [23] instead of entropy based on C_2 are listed in Tables 4, 5 and 6. Clearly, the respective accuracies, $\frac{9+8+8+4}{40} = 72.50\%$, $\frac{9+10+9+7}{40} = 87.50\%$ and $\frac{10+8+9+8}{40} = 87.50\%$, are not superior than those obtained with entropy.

B_2-based normalized ZCRs [24] were also used for comparisons, producing the results listed in Tables 7–9. Interestingly, the corresponding accuracies, i.e., $\frac{10+10+9+10}{40} = 97.5\%$, $\frac{10+10+10+10}{40} = 100\%$ and $\frac{10+10+10+10}{40} = 100\%$, are superior than those obtained with the use of entropy for $Q = 2$ and $Q = 3$ only, being equivalent for $Q = 4$. This is expected since, as shown in [24], a normalized ZCR acts as one neuron. Oppositely, entropy, which is a much more dense

Table 8
Results for speech recognition when using B_2 with $Q = 3$.

Actual/predicted	"Down"	"Right'	"Left"	"Up"
"Down"	10	0	0	0
"Right"	0	10	0	0
"Left"	0	0	10	0
"Up"	0	0	0	10

Table 9
Results for speech recognition when using B_2 with $Q = 4$.

Actual/predicted	"Down"	"Right"	"Left"	"Up"
"Down"	10	0	0	0
"Right"	0	10	0	0
"Left"	0	0	10	0
'Up"	0	0	0	10

structure as demonstrated in this article, benefits from a finer partition. Thus, it is reasonable to state that both normalized ZCRs and entropies are capable of doing the job with an acceptable accuracy, however, entropies are more convenient for higher values of Q.

To avoid misunderstandings and to keep the style of my previous tutorials [23,24], attention is required from the readers in regard to the specific and intended objective of this experiment and its corresponding results. Although the use of more dense classifiers, such as a knowledge-based learning algorithm instead of a simple distance metric, would obviously increase the accuracies I reported as being lower than 100%, my intention was to **necessarily** use the humblest existing possibility in such a way that it just **modestly interferes** in the process. Thus, the entropy-based feature vectors are the principal and fundamental entities responsible for the results listed in Tables 1–3. Additionally, the association of entropy-based feature vectors with, for instance, an MLP, an HMM or an independent SVM, is enough to provide a fine tunned deeper structure and, consequently, almost ideal confusion matrices for significantly bigger datasets.

An interesting topic for discussion: the divisions proportionated by C_2 do not necessarily segment the words into isolated phonemes. Despite not being a traditional procedure in most of the speech recognition algorithms, the proposed approach, which is only proper for restricted-vocabulary, was capable of performing the intended and necessary classifications independent of that separation. Thus, H-based features extracted using C_2 successfully circumvent this issue, somehow characterizing the information contained in each speech portion, even though it contains more than one phoneme. Furthermore, intra-speaker and inter-speaker variations [31] are also bypassed, up to a certain level, by means of the fixed-length feature vectors obtained when C_2 is adopted.

5. Conclusions

Taking advantage of my previous tutorials written to serve as references on energy and zero-crossing rates, respectively published in [23] and [24], this subsequent study provided the readers with a smoothly written tutorial on entropy. Although quite well explored in the related literature, a bibliographical review on this topic revealed there was room for supplemental studies, as demonstrated throughout my essay. Consequently, I presented distinct insights on entropy, its calculation and specially the way it may be interpreted, i.e., the outcome of a partially pre-tuned deep neural network.

Aside from the preliminar discussions and inspired on my former tutorials, I introduced two methods for feature extraction based on entropy, namely C_1 and C_2, being both adequate for use with 1D and 2D inputs. Numerical examples based on hypothetical data complemented the explanations, which were enriched with C/C++ source codes that implement the algorithms presented. Lastly, two modest but innovative example applications were included in the text: image synthesis from entropy, that was based on C_1 in 2D, and restricted-vocabulary speech recognition, that was based on C_2 in 1D. On one hand, the former technique shows how to synthesize an image, approximately, from its entropy-based information content. On the other, the latter emphasized the embedded potential entropy brings for pattern recognition tasks even when it is not associated with knowledge-based classifiers.

In comparison with energy and ZCRs, entropy contrasts and can not be, generically, considered better or worse. Instead, it should be adopted whenever the concept of information is significant and shows potential to solve the specific problem at hand. Additionally, entropy may be associated with ZCRs, which were demonstrated to be neurocomputing agents [24], and with energy, that is a more modest concept [23], in order to allow more sophisticated solutions.

Future research directions involving entropy may explore its potential when associated with dense knowledge-based algorithms, creating super deep architectures. Another possibility is to design knowledge-based classifiers using the structure shown in Fig. 3 so that the corresponding weights are assumed as the initial values for an optimized learning, consequently creating "quasi-entropic" features that are capable to treat specific problems effectively. Coherently with the results shown in Section 4 for image synthesis, auto-encoders [22,34] might be planned for compression or feature learning so that the codified features are those "quasi-entropic".

I close this essay by noticing that the proposed approaches offer distinct contributions for both young researchers and experienced professionals, grouping together *creativity*, *simplicity*, and *accuracy*, as in [23] and [24]. All the data I used to conduct the experiments are freely available to the scientific community upon prior request[2] so that the procedures could be reproduced at any time.

Acknowledgements

I am very grateful to CNPQ - "Conselho Nacional de Desenvolvimento Científico e Tecnológico", in Brazil, for the grants provided, through the process 306811/2014-6, to support this research.

References

[1] R. Arnheim, Entropy and Art: An Essay on Disorder and Order, 1.ed. University of California Press, 2010.
[2] J. Ajmera, I. McCowan, H. Bourlard, Speech/music segmentation using entropy and dynamism features in a HMM classification framework, Speech Commun. 40 (3) (2003) 351–363.
[3] S. Arora, Computational Complexity: A Modern Approach, Cambridge University Press, 2009.
[4] A. Aydemir, G. Gunduz, Fractal dimensions and entropies of meragi songs, in: Nonlinear Phenomena in Complex Systems: From Nano to Macro Scale. Book Series: NATO Science for Peace and Security Series C-Environmental Security, 2014, pp. 79–87.
[5] O. Bandyopadhyay, B. Chanda, B.B. Bhattacharya, Automatic segmentation of bones in x-ray images based on entropy measure, Int. J. Image Graph. 16 (1) (2016).
[6] M. Banerjee, N.R. Pal, Feature selection with SVD entropy: some modification and extension, Inf. Sci. (Ny) 264 (2014) 118–134.
[7] A.K. Bhandari, A. Kumar, G.K. Singh, Tsallis entropy based multilevel thresholding for colored satellite image segmentation using evolutionary algorithms, Expert. Syst. Appl. 42 (22) (2015) 8707–8730.
[8] A. Ben-Naim, Information, Entropy, Life and The Universe: What We Know and What We Do Not Know, 1.ed. World Scientific Publishing Co., 2015.
[9] M. Bossi, E. Goldberg, Introduction to Digital Audio Coding and Standards, Kluwer, 2003.
[10] S. Chandra, Energy, Entropy and Engines: An Introduction to Thermodynamics, 1, Wiley, 2016.
[11] G. Chao, S. Sun, Consensus and complementarity based maximum entropy discrimination for multi-view classification, Inf. Sci. (Ny) (2016) 296–310. 367–368

[2] Please, send requests to http://guido@ieee.org.

[12] F. Chen, P.C. Loizou, Contributions of cochlea-scaled entropy and consonant-vowel boundaries to prediction of speech intelligibility in noise, J. Acoust. Soc. Am. 131 (5) (2012) 4104–4113.
[13] T.M. Cover, J.A. Thomas, Elements of Information Theory, 2.ed. Wiley-Interscience, 2006.
[14] S.R. Dahmen, Music and entropy, Revista Brasileira de Ensino de Física 34 (2) (2012) 1–2.
[15] A. Daneshpazhouh, A. Sami, Entropy-based outlier detection using semi-supervised approach with few positive examples, Pattern Recognit. Lett. 49 (2014) 77–84.
[16] L. Deng, D. O'Shaughnessy, Speech processing: A Dynamic and Optimization-Oriented Approach, CRC Press, 2003.
[17] D. Dubois, W. Liu, J. Ma, H. Prade, The basic principles of uncertain information fusion: an organised review of merging rules in different representation frameworks, Inf. Fusion. 32-A (2016) 12–39.
[18] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, second ed., Wiley-Interscience, 2000.
[19] J.S. Dugdale, Entropy And Its Physical Meaning, 2.ed. Taylor & Francis, 1996.
[20] B. Farhadinia, Determination of entropy measures for the ordinal scale-based linguistic models, Inf. Sci. (Ny) 369 (2016) 63–79.
[21] B. Fassinut-Mombot, J. Choquel, A new probabilistic and entropy fusion approach for management of information sources, Inf. Fusion 5 (1) (2004) 35–47.
[22] A. Gibson, J. Patterson, Deep Learning: A Practitioner's Approach, O'Reilly Media, 2016.
[23] R.C. Guido, A tutorial on signal energy and its applications, Neurocomputing 179 (2016) 264–282.
[24] R.C. Guido, ZCR-Aided neurocomputing: a study with applications, Knowl. Based Syst. 105 (2016) 248–269.
[25] R.C. Guido, Effectively interpreting discrete wavelet transformed signals, IEEE Signal Process. Mag. 34 (3) (2017) 89–100.
[26] R.C. Guido, P. Addison, J. Walker, Introducing wavelets and time-frequency analysis, IEEE Eng. Biol. Med. Mag. 28 (5) (2009) 13.
[27] R.C. Guido, Practical and useful tips on discrete wavelet transforms, IEEE Sig. Process. Mag. 32 (3) (2015) 162–166.
[28] R.C. Guido, A note on a practical relationship between filters coefficients and the scaling and wavelet functions of the discrete wavelet transform, Appl. Math. Lett. 24 (7) (2011) 1257–1259.
[29] M. Hariharan, C.Y. Fook, R. Sindhu, A.H. Adom, S. Yaacob, Objective evaluation of speech dysfluencies using wavelet packet transform with sample entropy, Digit. Sig. Process. 23 (3) (2013) 952–959.
[30] D. Harris, S. Harris, Digital Design and Computer Architecture, Second ed., Morgan Kaufmann, 2012.
[31] J. Harrington, S. Cassidy, Techniques in Speech Acoustics, The Netherlands: Kluwer Academic Publishers, 1999.
[32] S.O. Haykin, Neural Networks and Learning Machines, in: Third ed., Pearson, 2008.
[33] H. He, B. Chen, J. Guo, Emotion recognition of pop music based on maximum entropy with priors, in: Advances in Knowledge Discovery and Data Mining, Proc. Book Series: Lecture Notes in Artificial Intelligence, 5476, 2009, pp. 788–795.
[34] J. Heaton, Artificial Intelligence for Humans,v.3 Deep Learning and Neural Networks., CreateSpace Independent Publishing Platform, 2015.
[35] I.I. Hirschman, Infinite Series, Dover Publications, 2014.
[36] T. Izumitani, K. Kashino, Frequency component restoration for music sounds using a markov random field and maximum entropy learning, in: In:2006 IEEE International Conference on Acoustics, Speech, and Signal Processing (IEEE ICASSP 2006), 1, 2006, pp. 257–260.
[37] F. Jiang, Y. Sui, L. Zhou, A relative decision entropy-based feature selection approach, Pattern Recognit. 48 (7) (2015) 2151–2163.
[38] E. Joseph, Sorting Algorithm: Analysis and Comparison Performance, Lap Lambert Academic Publishing, 2012.
[39] O. Kafri, H. Kafri, E. Lotem, Entropy - God's Dice Game, CreateSpace Independent Publishing Platform, 2013.
[40] S. Khorram, H. Sameti, F. Bahmaninezhad, S. King, T. Drugman, Context-dependent acoustic modeling based on hidden maximum entropy model for statistical parametric speech synthesis, Eurasip Journal on Audio, Speech and Music Processing. article 12, 2014.
[41] J.H. Kim, Estimating classification error rate: repeated cross-validation, repeated hold-out and bootstrap, Comput. Stat. Data Anal. 53 (11) (2009) 3735–3745.
[42] S.E. Kim, J.J. Jeon, I.K. Eom, Image contrast enhancement using entropy scaling in wavelet domain, Sig. Process. 127 (2016) 1–11.
[43] O.A. Kittaneh, M.A.U. Khan, M. Akbar, H.A. Bayoud, Average entropy: a new uncertainty measure with application to image segmentation, Am. Stat. 70 (1) (2016) 18–24.
[44] L. Knopoff, W. Hutchinson, Entropy as a measure of style, the influence of sample length plus music theory, J. Music Theory 27 (1) (1983) 75–97.
[45] P. Korus, J.W. Huang, Improved tampering localization in digital image forensics based on maximal entropy random walk, IEEE Signal Process Lett 23 (1) (2016) 169–173.
[46] R. Kummel, The second law of economics: Energy, Entropy, and The Origins of Wealth, First ed., Springer, 2011.
[47] L. Laur, P. Rasti, M. Agoyi, Anbarjafari, g. a robust color image watermarking scheme using entropy and QR decomposition., Radioengineering 24 (4) (2015) 1025–1032.
[48] Y.H. Lee, H.K. Kim, Entropy coding of compressed feature parameters for distributed speech recognition, Speech Commun. 52 (5) (2010) 405–412.
[49] D.S. Lemons, A Student's Guide to Entropy, First ed., Cambridge University Press, 2013.
[50] F. Li, Z. Zhang, C. Jin, Feature selection with partition differentiation entropy for large-scale data sets, Inf. Sci. (Ny) 329 (2016) 690–700.
[51] M.K. Mehlawat, Credibilistic mean-entropy models for multi-period portfolio selection with multi-choice aspiration levels, Inf. Sci. (Ny) 345 (2016) 9–26.
[52] F. Nie, et al., A general graph-based semi-supervised learning with novel class discovery, Neural Comput. Appl. 19 (4) (2010) 549–555.
[53] F. Nie, et al., A novel generalized entropy and its application in image thresholding, Sig. Process. 134 (2017) 23–34.
[54] N. Obin, M. Liuni, On the generalization of shannon entropy for speech recognition, in: year 2012 IEEE Workshop on Spoken Language Technology (IEEE SLT year 2012), 2012, pp. 97–102.
[55] A.V. Oppenheim, R.W. Schafer, Discrete-time Signal Processing, Third ed., Prentice-Hall, 2009.
[56] F.A.N. Palmieri, D. Ciuonzo, Objective priors from maximum entropy in data classification, Inf. Fusion 14 (2) (2013) 186–198.
[57] W. Peng, H. Deng, Quantum inspired method of feature fusion based on von neumann entropy, Inf. Fusion 18 (2014) 9–19.
[58] T.D. Pham, The kolmogorov-sinai entropy in the setting of fuzzy sets for image texture analysis and classification, Pattern Recognit 53 (2016) 229–237.
[59] H. Pishro-Nik, Introduction to Probability, Statistics, and Random Processes, Kappa Research, 2014.
[60] V.H. Rallapalli, J.M. Alexander, Neural-scaled entropy predicts the effects of nonlinear frequency compression on speech perception, J. Acoust. Soc. Am. 138 (5) (2015) 3061–3072.
[61] H.V. Ribeiro, L. Zunino, R.S. Mendes, E.K. Lenzi, Complexity-entropy causality plane: a useful approach for distinguishing songs, Physica A Stat. Mech. Appl. 391 (7) (2012) 2421–2428.
[62] M. Saritha, K.P. Joseph, A.T. Mathew, Classification of MRI brain images using combined wavelet entropy based spider web plots and probabilistic neural network, Pattern Recognit. Lett. 34 (16) (2013) 2151–2156.
[63] S. Sarkar, S. Das, S.S. Chaudhuri, A multilevel color image thresholding scheme based on minimum cross entropy and differential evolution, Pattern Recognit.Lett. 54 (2015) 27–35.
[64] S. Sarkar, S. Das, S.S. Chaudhuri, Hyper-spectral image segmentation using renyi entropy based multi-level thresholding aided with differential evolution, Expert Syst. Appl. 50 (2016) 120–129.
[65] J.C. Sanford, Genetic Entropy and the Mystery of the Genome, Third ed., Feed My Sheep Foundation, Inc., 2008.
[66] P.R. Scalassara, M.E. Dajer, C.D. Maciel, R.C. Guido, J.C. Pereira, Relative entropy measures applied to healthy and pathological voice characterization, Appl. Math. Comput. 207 (1) (2009) 95–108.
[67] J.P. Sethna, Statistical Mechanics: Entropy, Order Parameters and Complexity, First ed., Oxford University Press, 2006.
[68] C.E. Shannon, A mathematical theory of communication, Bell Syst. Tech. J. 27 (3) (1948) 379–423.
[69] Y. Song, Y.Q. Wu, Y.M.A. Dai, A new active contour remote sensing river image segmentation algorithm inspired from the cross entropy, Digit. Sig. Process. 48 (2016) 322–332.
[70] J.V. Stone, Information Theory: A Tutorial Introduction., Sebtel Press, 2015.
[71] B. Stroustrup, The C++ Programming Language, Fourth ed., Addison-Wesley Professional, 2013.
[72] S. Sultana, C. Shahnaz, S.A. Fattah, I. Ahmmed, W.P. Zhu, M.O. Ahmad, Speech emotion recognition based on entropy of enhanced wavelet coefficients, in: In year 2014 IEEE Int. Symposium on Circuits and Systems (IEEE ISCAS year 2014), 2014, pp. 137–140.
[73] O. Techakesari, J.J. Ford, Relative entropy rate based model selection for linear hybrid system filters of uncertain nonlinear systems, Sig. Process. 93 (2013) 12–22.
[74] C.C. Tu, C.F. Juang, Recurrent type-2 fuzzy neural network using haar wavelet energy and entropy features for speech detection in noisy environments, Expert Syst. Appl. 39 (3) (2012) 2479–2488.
[75] M. Xia, Z. Xu, Entropy/cross entropy-based group decision making under intuitionistic fuzzy environment, Inf. Fusion 13 (1) (2012) 31–47.
[76] J. Xu, et al., Robust and sparse fuzzy k-means clustering, in: In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16), 1, 2016, pp. 2224–2230.
[77] C.Q. Yin, W. Li, Y.Q. Luo, L.C. Tseng, Robust online music identification using spectral entropy in the compressed domain, in: In year 2014 IEEE Wireless Communications and Networking Conference Workshops (IEEE WCNCW year 2014). Book Series: IEEE Wireless Communications and Networking Conference Workshops, 2014, pp. 128–133.
[78] D. Yu, B. Varadarajan, L. Deng, A. Acero, Active learning and semi-supervised learning for speech recognition: a unified framework using the global entropy reduction maximization criterion, Comput. Speech Lang. 24 (3) (2010) 433–444.
[79] H.Y. Yu, X.B. Zhi, J.L. Fan, Image segmentation based on weak fuzzy partition entropy, Neurocomputing 168 (2015) 994–1010.
[80] R. Wang, T. Zeng, C. Hu, J. Yang, T. Long, Accurate range profile alignment method based on minimum entropy for inverse synthetic aperture radar image formation, IET Radar Sonar Navig. 10 (4) (2016) 663–671.

[81] Y. Wang, et al., EEG signal co-channel interference suppression based on image dimensionality reduction and permutation entropy, Sig. Process. 134 (2017) 113–122.
[82] W. Wien, The temperature and entropy of radiation, Physikalische Zeitschrift 2 (1900). 111–111
[83] M. Zarinbal, M.H.F. Zarandia, I.B. Turksen, Relative entropy collaborative fuzzy clustering method, Pattern Recognit. 48 (3) (2015) 933–940.
[84] B. Zhang, et al., Data stream clustering based on fuzzy c-mean algorithm and entropy theory, Sig. Process. 126 (2016) 111–116.
[85] C. Zhang, J.H.L. Hansen, Whisper-island detection based on unsupervised segmentation with entropy-based speech feature processing, IEEE Trans. Audio Speech Lang. Process. 19 (4) (2011) 883–894.
[86] X. Zhang, C. Mei, D. Chen, J. Li, Feature selection in mixed data: A method using a novel fuzzy rough set-based information entropy, Pattern Recognit. 56 (2016) 1–15.
[87] W.D. Zhao, Z.J. Xu, J. Zhao, Gradient entropy metric and p-laplace diffusion constraint-based algorithm for noisy multispectral image fusion, Inf. Fusion 27 (2016) 138–149.