Algorithmic Cross-Complexity and Relative Complexity

Daniele Cerra[1] and Mihai Datcu[1,2]
[1] German Aerospace Center (DLR), Remote Sensing Technology Institute, 82234 Wessling, Germany
[2] Télécom Paris, 46 rue Barrault, 75634 Paris, France
Email: {daniele.cerra, mihai.datcu}@dlr.de

Abstract – Information content and compression are tightly related concepts that can be addressed by classical and algorithmic information theory. Several entities in the latter have been defined relying upon notions of the former, such as entropy and mutual information, since the basic concepts of these two approaches present many common tracts. In this work we further expand this parallelism by defining the algorithmic versions of cross-entropy and relative entropy (or Kullback-Leibler divergence), two well-known concepts in classical information theory. We define the cross-complexity of an object x with respect to another object y as the amount of computational resources needed to specify x in terms of y, and the complexity of x related to y as the compression power which is lost when using such a description for x, with respect to its shortest representation. Since the main drawback of these concepts is their uncomputability, a suitable approximation based on data compression is derived for both and applied to real data. This allows us to improve the results obtained by similar previous methods which were intuitively defined.

1. Introduction

Both classical and algorithmic information theory aim at defining, under different points of view, such concepts as information content and shared information, and at fixing lower bounds for compression capabilities. Classical Shannon's information theory [1] has a probabilistic approach, being based on the uncertainty of the outcomes of random variables: given that it is an ensemble notion, it cannot reveal anything about the information content of an isolated object, if no a priori knowledge is available. In algorithmic information theory this limitation is discarded, since the primary concept is instead the information content of an individual object, which is a measure of how difficult it is to specify or describe how to construct or calculate that object. This notion is also known as Kolmogorov complexity. The original formulation of the concept of algorithmic information is independently due to A. N. Kolmogorov [2], G. J. Chaitin [3], and R. J. Solomonoff [4]. This area of study allowed formal definitions of concepts which were previously vague, such as randomness, Occam's razor, simplicity and complexity. The basics of classical and algorithmic information theory are indeed similar, and many concepts do exist in both frames defined in parallel: Shannon's entropy has its counterpart in Kolmogorov complexity, and this correspondence holds for the conditional and joint versions of these notions, allowing a representation of mutual information in both frames, sharing many properties [5].
The aim of this work is to consolidate this parallelism by expanding it with the definition of cross-complexity and relative complexity, the algorithmic versions of cross-entropy and relative entropy (also known as Kullback-Leibler divergence). We define them between any two strings x and y respectively as the computational resources needed to specify x only in terms of y, and the compression power which is lost when using this representation for x instead of its most compact one, which has length equal to its Kolmogorov complexity.
Since these newly introduced concepts are uncomputable, we rely on previous works in the literature which assimilate the complexity of an object to the size of its compressed version [6], and adopt a compression based algorithm to derive a computable approximation of both cross and relative complexity, enabling their application to real data.

In the past a similar approach was used in [7], where the relative entropy between two strings was intuitively defined and successfully applied through data compression to cluster and classify texts: that work can now be better understood, and its results improved, by introducing the relative complexity and its compression-based approximation.
This paper is organized as follows. We give a brief reminder of the basic concepts of Shannon entropy and Kolmogorov complexity in section 2, focusing on their shared properties and their relation with data compression. Section 3 introduces the algorithmic cross-complexity and relative complexity, while in section 4 we define their computable approximations using compression-based techniques. Practical applications and a comparison with previous similar methods are reported in section 5. We conclude in section 6.

2. Preliminaries

2.1. Shannon entropy and Kolmogorov complexity

The Shannon entropy in classical information theory [1] is an ensemble concept; it is a measure of the degree of ignorance about the outcomes of a random variable X with a given a priori probability distribution $p(x) = P(X=x)$:

$$H(X) = -\sum_x p(x)\log(p(x)). \quad (1)$$

This definition can be interpreted as the average length in bits, which can be obtained for example through the Shannon-Fano code, needed to encode the outcomes of X; this allows achieving compression on the long run. Nevertheless, such approach related to probabilistic assumptions does not provide the informational content of individual objects and their possible regularity. This lacuna is filled by the concept of Kolmogorov complexity $K(x)$, that evaluates an intrinsic complexity for any isolated string x, independently of any description formalism. In this work we consider the "prefix" complexity, which is the size in bits (binary digits) of the shortest self-delimiting program q used as input by a universal Turing machine to compute x and halt [5]:

$$K(x) = \min_{q \in Qx} |q|, \quad (2)$$

with Qx being the set of instantaneous codes that generate x. One interpretation of $K(x)$ is as the quantity of information needed to recover x from scratch. A formal link between entropy and algorithmic complexity has been established in the following theorem [5].
Theorem 1: The sum of the expected Kolmogorov complexities of all the code words x which are output of a random source X, weighted by their probabilities $p(x)$, equals the statistical Shannon entropy of X, up to an additive constant:

$$H(X) \le \sum_x p(x)K(x) \le H(X) + K(p) + O(1). \quad (3)$$

Thus, for low complexity distributions lowering the impact of $K(p)$, which represents the complexity of the probability function itself, the expected complexity is close to the entropy.

2.2. Mutual information and other parallelisms

An important issue of the informational content analysis is the estimation of the amount of information shared by two objects. From Shannon's probabilistic point of view, this is done

via the mutual information $I(X,Y)$ between two random variables X and Y, defined in terms of entropy as: $I(X,Y) = H(X) - H(X \mid Y) = H(X) + H(Y) - H(X,Y)$, where $H(X|Y)$ is the *conditional entropy* of X given Y and $H(X,Y)$ is the *joint entropy* of X and Y.
Is it possible to obtain similar laws and entities in the Kolmogorov complexity frame? The answer is given by defining the *algorithmic mutual information* between two strings x and y as: $I_w(x:y) = K(x) - K(x \mid y) = K(x) + K(y) - K(x,y)$, valid up to an additive constant, where the *conditional complexity* $K(x|y)$ of x related to y quantifies the information needed to recover x given y, while the *joint complexity* $K(x,y)$ is the length of the shortest program which outputs x followed by y. Note that if y carries information which is shared with x, $K(x|y)$ will be considerably smaller than $K(x)$. For all these definitions, the desirable properties of analogous quantities in classical information theory hold [5]. If $I_w(x:y) = 0$, then $K(x,y) = K(x) + K(y)$, and x and y are for definition *algorithmically independent.*
Another parallelism is found for the rate-distortion theory [1]: it has as its equivalent in the algorithmic frame the Kolmogorov structure functions [8], which aim at separating the meaningful (structural) information contained in an object from its random part (its randomness deficiency), characterized by less meaningful details and noise.

2.3. Compression-based approximations and the similarity metric

Since the complexity $K(x)$ is not a computable function of x, a suitable approximation is defined by Li and Vitányi by considering it as the size of the ultimate compressed version of x, and a lower bound for what a real compressor can achieve [6]. This allows approximating $K(x)$ with $C(x)$, i.e. the length of the compressed version of x obtained with any off-the-shelf lossless compressor C. It is important to remark that it is not possible to estimate how close to its lower bound this approximation is. The conditional complexity $K(x|y)$ can be also estimated through compression [9] [10], while the joint complexity $K(x,y)$ is approximated by compressing the concatenation of x and y. The probably greatest practical success of these notions is the ultimate estimation of shared information between two objects: the Normalized Information Distance (NID). The NID is a similarity metric minimizing any admissible metric, and it is proportional to the length of the shortest program that computes x given y, as well as computing y given x. It can be approximated by the Normalized Compression Distance (NCD) [6] as follows:

$$NID(x,y) = \frac{K(x,y) - \max\{K(x),K(y)\}}{\min\{K(x),K(y)\}} \quad \overset{K(x)\to C(x)}{\to} \quad NCD(x,y) = \frac{C(x,y) - \max\{C(x),C(y)\}}{\min\{C(x),C(y)\}} \qquad (4)$$

The NCD and its variants may be used to perform clustering, classification and anomaly detection on diverse data types and with a totally parameter-free approach [11] [12].

3. Expanding the Parallelism Shannon/Kolmogorov

3.1. Cross-entropy and cross-complexity

Let us start by recalling the definition of cross-entropy in Shannon's frame:

$$H(X \oplus Y) = -\sum_i P_X(i) \log P_Y(i), \qquad (5)$$

with $P_X(i) = P(X = i)$ and $P_Y(i) = P(Y = i)$. The cross-entropy may be regarded as the average number of bits needed to specify an object i generated by a variable X when using as a priori knowledge to encode it the probability distribution of another variable Y. This notion

can be brought in the algorithmic frame to determine how to measure the computational resources needed to specify an object *x* in terms of another one *y*.
To define the algorithmic cross-complexity we will rely on a class of Turing machines similar to the one described by Chaitin in his definition of algorithmic complexity [13]. This is done without loss of generality, since Solomonoff showed that every Turing Machine may be programmed to behave like any other one by adding a constant needed to pass from a representation to another[4]. Consider a Turing machine with three tapes: a program tape, a work tape, and an output tape. The unbounded work tape initially contains the program $y*$ of length $K(y)$, which is the shortest description of *y*, followed by the string *x*, both encoded in a prefix-free way. The program tape contains a program *P*, while the output tape is initially blank. The programs *P* and $y*$ and the string *x* are sequences of squares containing 0 or 1. No blank is needed to separate $y*$ and *x* since $y*$ is an instantaneous code. The work tape may be shifted in either direction and may be read, written and erased, while the program tape and the output tape are respectively read-only and write-only, and can be shifted only to the right. This class of Turing machine has a finite number *n* of states, defined in an *nx3* table, giving the action to be performed and the next state as a function of the current state, plus the content of the square of the working tape currently scanned. Only two commands are available for *P*: write in output a substring $\{y*_{i,j}\}$ of $y*$ from index *i* to index *j*, or write in output a constant *c* representing the command "print", followed by a substring $\{x_{i,j}\}$ of *x* from index *i* to index *j*. All the $\{y*_{i,j}\}$ and $\{x_{i,j}\}$ are required to contain a set of instantaneous code words. The following operations are allowed on this set of Turing machines: Halt; shift left or right the Work tape; write Blank, 0, or 1 in the Work tape; read a square from the Program tape, copy it in the Work tape, then shift right the Program tape; and write a 0 or a 1 on the Output tape, then shift right the Output tape. The program *P* yields the minimum size of the output, under the constraint of using only the described limited commands. For each *x* and *y*, the final output of the machine is a program $(x \oplus y)*$, that can be copied in the program tape of the Turing machine described in [13], outputting a sequence *x*. Since the code words of $(x \oplus y)*$ are prefix-free, so is $(x \oplus y)*$. We state the length of $(x \oplus y)*$ to be equal to $K(x \oplus y)$. The cross-complexity $K(x \oplus y)$ is radically different from the conditional complexity $K(x \mid y)$: while in the latter the object *y* is an auxiliary input that is given "for free" and does not count in the estimation of the computational resources needed to specify *x*, in the former we force *x* to be expressed in terms of a description tailored for *y* and do not allow any other way of compactly representing *x*.

Lemma 1. The algorithmic cross-complexity of *x* given *y* is lower-bounded by the Kolmogorov complexity of *x*, i.e. $K(x \oplus y) \geq K(x), \quad \forall x, y$.
Proof. Assume that $\exists\, x, y \mid \quad K(x \oplus y) < K(x)$: this means that $K(x \oplus y)$ is a more compact representation of *x* with respect to $K(x)$. But, unlike conditional complexity, i.e. $K(x \mid y)$, the program $(x \oplus y)*$ of length $K(x \oplus y)$ does not need any additional input to compute *x*. So, $(x \oplus y)*$ would be a self-contained representation of *x* shorter than $K(x)$, which is in contradiction with the statement that $K(x)$ is a lower bound for all such representations.
The lower bound of $K(x \oplus y) = K(x)$ is reached for the trivial case of $x = y$, if only the full $y*$ is copied in $(x \oplus y)*$.

Lemma 2. $K(x \oplus y)$ and $K(x)$ share the same upper bound, i.e. $K(x \oplus y) \leq\mid x \mid + O(\log x)$.
Proof. If any portion of the code $y*$ can be used to shorten the description of *x* the above is true. In the worst case, *x* and *y* are algorithmically independent [13]. Therefore, the shortest

description of x relative to y is the string x itself. The term of $O(\log x)$ would be needed to encode x in a self-delimiting way [5], if x had not this property.
We can now resume the cross-entropy's properties which hold for our definition of algorithmic cross-complexity:

1. $H(X \oplus Y) \geq H(X) \quad \rightarrow \quad K(x \oplus y) \geq K(x)$,
2. $H(X \oplus Y) = H(X), \quad if\ X = Y \quad \rightarrow \quad K(x \oplus y) = K(x), \quad if \quad x = y$. Note that the strongest condition $H(X \oplus Y) = H(X), \quad iff\ X = Y$ was so far not demonstrated to hold in the algorithmic frame.
3. The cross-entropy $H(X \oplus Y)$ of X given Y and the entropy $H(X)$ of X share the same upper bound - $\log(N)$, where N is the number of possible outcomes of X - as algorithmic complexity and algorithmic cross-complexity do.

3.2. Relative entropy and relative complexity

The definition of algorithmic relative complexity derives from the definition of relative entropy (or Kullback-Leibler divergence) related to two probabilistic distributions X and Y. It may be regarded as a distance between X and Y, or as the expected difference in the number of bits required to code an outcome i of X when using an encoding based on Y, instead of X [14]:

$$D(X \,||\, Y) = -\sum_i P_X(i) \log \frac{P_X(i)}{P_Y(i)} \tag{6}$$

We recall the most important properties of this distance: $D(X \,||\, Y) \geq 0$, satisfying Gibb's inequality, with equality $iff \quad X = Y$, and $D(X \,||\, Y) \neq D(Y \,||\, X)$. $D(X \,||\, Y)$ is not a metric, as it is not symmetric and the triangle inequality does not hold [15]. What is more of interest for our purposes is the definition of the relative entropy expressed in terms of difference between cross-entropy and entropy:

$$D(X \,||\, Y) = H(X \oplus Y) - H(X) \tag{7}$$

From this definition we switch to the "complexity world", and derive the equation for the algorithmic relative complexity from its previously defined components:

$$K(x \,||\, y) = K(x \oplus y) - K(x), \tag{8}$$

which represents the compression power lost when compressing x by describing it only in terms of y, instead of using its most compact representation. We may also regard $K(x \,||\, y)$, as for its counterpart in Shannon frame, as a quantification of the distance between x and y. It is desirable that the main properties of (6) hold also for (8): this is the case, as shown by the following lemmas.

Lemma 3. The algorithmic relative complexity of x given y is positively defined in the interval $[0, \quad |x| + O(\log x)], \ \forall x, y$.
Proof. Ref. lemmas 1 and 2: to obtain the lower bound of 0, substitute the lower bound for $K(x \oplus y)$ in (8); the upper bound is given by the upper bound for $K(x \oplus y)$ and the lower bound for $K(x) \approx \log(x)$, considering that $O(\log(x)) - \log(x) = O(\log(x))$.

Lemma 4. The algorithmic relative complexity is not symmetric: for some x and y, $K(x \,||\, y) \neq K(y \,||\, x)$.

Proof. It is enough to find two strings x and y for which $K(x \| y)$ is not symmetric. Given two algorithmically independent sequences A and $B \in \{0,1\}^*$ of the same length, consider the strings obtained by appending A to B and A to A, i.e. $x = \{A + B\}$ and $y = \{A + A\}$. If B is a very simple sequence with respect to A such that $K(x) \approx K(y) \approx K(A)$, then we have: $K(x \| y) - K(y \| x) = K(x \oplus y) - K(x) - K(y \oplus x) + K(y) \approx K(x \oplus y) - K(y \oplus x)$. It is easy to notice that y can be totally reconstructed by the optimal code to generate x, while the contrary is not true. Therefore we can assume $K(x \oplus y) >> K(y \oplus x)$, and the lemma is proved since $K(x \| y) > K(y \| x)$.

4. Compression-based Computable Approximations

4.1. Computable algorithmic cross-complexity

The uncomputability of the algorithmic cross-complexity and relative complexity is a direct consequence of the uncomputability of their Kolmogorov complexity components. We once again rely on data compression based techniques to derive an approximation $K(x \oplus y) \to C(x \oplus y)$. Consider two strings x and y and suppose to have available a dictionary *Dic*(*y*,*i*) extracted scanning y from the beginning until position i with the *LZW* algorithm [16] for each i, using an unbounded buffer. A representation $C(x \oplus y)^*$ of x, which has initial length $|x|$, is computed as in the pseudo-code in Fig.1. The output of this computation has length $C(x \oplus y)$, which is then the size of x compressed by the dictionary generated from y, if a parallel processing of x and y is simulated.

1. Position p=0.
2. If $p = |x|$, then Halt.
3. Consider the symbol x_p at position p. If the partial dictionary *Dic*(*y*,*p*) contains a word starting with x_p, then:
    a. Output the code of a pattern c of length n contained in *Dic*(*y*,*p*) matching a substring of x starting at x_p, chosen so that n is maximal .
    b. p=p+n
    c. Go to 2
4. Output x_p.
5. p=p+1
6. Go to 2

Fig. 1. Pseudo-code to generate an approximation $C(x \oplus y)$ of the cross-complexity $K(x \oplus y)$ between two strings *x* and *y*.

It is possible to create a unique dictionary before-hand for a string y as a hash table containing couples (*key, value*), where *key* is the position in which the pattern occurs the first time, while *value* contains the full pattern. Then $C(x \oplus y)^*$ can be computed by matching the patterns in x with the portions of the dictionary of y with *key* < actual position in x. Note that $C(x \oplus y)$ is a cheap approximation of $K(x \oplus y)$ that constitutes its lower bound: it is not possible to know how much this lower bound is approached. We report in the following tables 1 and 2 a simple practical example.

A	*Dict*(*A*)	*A**	(*A*⊕*B*)*	***B***	*Dict*(*B*)	*B**	(*B*⊕*A*)*
a				a			
b	ab=<256>	a	a	b	ab=<256>	a	a
c	bc=<257>	b	b	a	ba=<257>	b	b
a	ca=<258>	c	c	b			
b				a	aba=<258>	<256>	<256>
c	abc=<259>	<256>	<256>	b			
a			c	a			<256>
b	cab=<260>	<258>		b	abab=<259>	<258>	
c			<256>	a			<256>
a	bca=<261>	<257>	c	b	bab=<260>	<257>	
b				a			<256>
c			<256>	b			
		<259>	c			<260>	<256>

Table 1. Extracted dictionaries and compressed versions of *A* and *B*, plus cross-compressions between *A* and *B*, computed with the algorithm reported in Fig. 1.

	A	*B*	*A**	*B**	(*A*⊕*B*)*	(*B*⊕*A*)*
Symbols	12	12	7	6	9	7
Bits/Symbol	8	8	9	9	9	9
Size(bits)	96	96	63	54	81	63

Table 2. Estimated complexities and cross-complexities for the sample strings *A* and *B*. Since *A* and *B* share common patterns, compression is achieved, and it is more effective when *B* is expressed in terms of *A* due to the fact that *A* contains all the relevant patterns within *B*.

Consider two ASCII-coded strings A={abcabcabcabc} and B={abababababab}. By applying the *LZW* algorithm, we extract and use two dictionaries *Dict*(A) and *Dict*(B) to compress A and B into two strings A* and B* of length $C(A)$ and $C(B)$, respectively. By applying the pseudo-code in Fig. 1 we compute (A⊕B)* and (B⊕A)*, of lengths $C(A \oplus B)$ and $C(B \oplus A)$.

4.2. Computable algorithmic relative complexity

We define an approximation of the relative complexity between two strings x and y as:

$$C(x \,||\, y) = C(x \oplus y) - C(x), \tag{9}$$

with $C(x \oplus y)$ computed as described above and $C(x)$ representing the length of x after being compressed with the *LZW* algorithm. Finally, we introduce an approximated normalized relative complexity as following:

$$\overline{C}(x \,||\, y) = \frac{C(x \oplus y) - C(x)}{|\, x \,| - C(x)}. \tag{10}$$

The distance (10) ranges from 0 to 1, representing respectively maximum and minimum similarity between x and y.

4.2.1. Relative entropy, revised

In the work that paved the way for practical applications of compression based similarity measures, Benedetto et al. [7] defined the relative entropy of a string x related to a string y as:

$$H_r(x \,||\, y) = \frac{C(x+\Delta y) - C(x) - (C(y+\Delta y) - C(y))}{|\Delta y|}. \quad (11)$$

Their intuition was correct, and showed the power and adaptability of compression at discovering similarities in general data with a parameter-free approach. The proposed concept of relative complexity allows a better understanding of their pioneering work. To establish an informal correspondence between (11) and (10), in order for the equations to resemble more each other, consider (10) with Δy, a fraction of *y*, and *x* as its arguments:

$$\overline{C}(\Delta y \,||\, x) = \frac{C(\Delta y \oplus x) - C(\Delta y)}{|\Delta y| - C(\Delta y)}. \quad (12)$$

We can now highlight the differences between these two distance measures.

A. The term $C(x+\Delta y) - C(x)$ in (11) is intuitively close to $C(\Delta y \oplus x)$ in (12), since both aim at expressing a small fraction of *y* only in terms of *x*. Nevertheless, note that in (11) also a small dictionary extracted from Δy itself is used in the compression step: this means that this term presents interferences with the conditional compression $C(\Delta y \,|\, x)$, resulting in an underestimation of the cross-complexity. This is undesired since, in the case of Δy being algorithmically independent from *x* and characterized by low complexity, Δy would be compressed by its own dictionary rather than by the model learned from *x*. Furthermore, the dictionary extracted from *x* would grow and grow according to the length of *x*: this could generate confusion, as patterns which are not relevant would be taken into account and used to compress Δy. Finally, if *x* is longer than *y*, Δy could be better compressed by *x* than by *y* itself, yielding a negative result for (11).

B. The term $C(y+\Delta y) - C(y)$ in (11) is intuitively close to $C(\Delta y)$ in (12). In the first case a representative dictionary extracted from *y* is used to code the fraction Δy, while the definition (10) allows us to discard any limitation regarding the size of the analyzed objects and to consider the full string *y*.

C. The normalization term in the two equations is different: the equation (11) is not upper bounded by 1 in the case of *x* and *y* being algorithmically independent, which is desired, but by a smaller quantity; in fact, $|\Delta y| > \max\{C(x+\Delta y) - C(x) - (C(y+\Delta y) - C(y))\}$, since $|\Delta y| > \Delta y - \min\{C(y+\Delta y) - C(y)\}$, due to the monotonicity property of *C*, which ensures that the quantity $C(y+\Delta y) - C(y)$ is strictly positive, $\forall y$. Therefore, the maximum distance in (11) also depends on the complexity of Δy, while it should in principle be independent.

D. The distance (11) is based on Δy, a small fraction of *y*. This could not be enough to consider all the relevant information contained in the string. On the contrary, (10) allows using strings of unbounded length, even though it truncates one of them to have them of the same size, due to the scanning of the two performed in parallel.

4.2.2. Symmetric relative complexity

Kullback and Leibler themselves define their distance in a symmetric way: $J(X,Y) = I_{1:2}(X,Y) + I_{2:1}(X,Y)$ [15]. We define a symmetric version of (10) as:

$$\overline{C_S}(x \,||\, y) = \frac{1}{2}\overline{C}(x \,||\, y) + \frac{1}{2}\overline{C}(y \,||\, x). \quad (13)$$

In our normalized equation we divide both terms by 2 to keep the values between 0 and 1. For the strings A and B considered in the simple example in Tables 1 and 2, we obtain the following estimations: $\overline{C}(A||B)=0.54$, $\overline{C}(B||A)=0.21$, $\overline{C_S}(A \,||\, B) = 0.38$. This means that B can be better expressed in terms of A than vice versa, but overall the strings are quite similar.

5. Applications

Even though our main concern is not the performance of the introduced distance measures, we report some practical application examples in order to show their consistence, and to compare them with their predecessor (11).

5.1. Authorship attribution

The problem to automatically recognize the author of a given text is given. In the following experiment the same procedure as [7], and a dataset as close as possible, have been adopted: in this case $\overline{C_S}(x \,||\, y)$ has been used as a distance measure instead of (11). A collection of 90 texts of 11 known Italian authors spanning the centuries XIII-XX has been considered [17]. Each text T_i was used as an unknown text against the rest of the database, its closest object T_k minimizing $\overline{C_S}(T_i \,||\, T_k)$ was retrieved, and T_i was then assigned to the author of T_k.

Author	Dan	D'An	Del	Fog	Gui	Mac	Man	Pir	Sal	Sve	Ver	TOT
Texts	8	4	15	5	6	12	4	11	11	5	9	90
Successes	8	4	15	3	6	12	4	11	11	5	9	88

Table 3. Autorship attribution. Each text from the 11 authors is used to query the database, and it is considered written by the author of the most similar retrieved work. Overall accuracy is 97.8%. The authors' names: Dante Alighieri, G. D'Annunzio, G. Deledda, A. Fogazzaro, F. Guicciardini, N. Machiavelli, A. Manzoni, L. Pirandello, E. Salgari, I. Svevo, G. Verga.

The results, reported in Table 3, show that the correct author $A(T_i)$ for each T_i has been found correctly in 97.8%, of the cases, while Benedetto et al. reached an accuracy of 93.3%. This confirms that the proposed computable measure is a better approximation of the "relative entropy" than the one described in [7]. On the other hand, our approximation requires more computational resources and cannot be computed by simply compressing a file. Finally, it has to be remarked that, in order to use both methods, it is needed to encode first the data into strings, while distance measures as (4) may be applied directly by using any compressor.

5.2. Satellite images classification

In the second experiment we classified a labelled satellite images dataset, containing 600 optical single band image subsets acquired by the SPOT 5 satellite, of size 64x64, with a spatial resolution of 5 meters. The dataset was divided in 6 classes (clouds, sea, desert, city, forest and fields) and split in 200 training images and a test set composed of the remaining 400. As a first step, the images were encoded into strings by traversing them line by line; then a distance matrix was built by applying (13) between each pair of training and test images; finally, each subset was simply assigned to the class from which the average distance was minimal. Results reported in Table 4 show an overall satisfactory performance, achieved considering only the horizontal information within the image subsets. It has to be remarked that in the process have been skipped both the feature extraction and the parameter tuning

steps, which may hinder the analysis [12] and are often required by conventional classification methodologies for this kind of data.

Class	Clouds	Sea	Desert	City	Forest	Fields	**Average**
Accuracy (%)	97	89.5	85	97	100	44.5	85.5

Table 4. Accuracy for satellite images classification (%) using the relative complexity as distance measure. A good performance is reached for all classes except for the class fields, confused with city and desert.

Better results may be obtained on the same dataset if the vertical information within the images is exploited, by using (4) and choosing Jpg2000 as compressor (93.5% accuracy) [18].

6. Conclusions

Two new concepts in the algorithmic information theory frame have been introduced: the ideas of cross-complexity and relative complexity, both defined between any two strings. It is remarkable that the main properties of the corresponding classical information theory concepts, cross-entropy and relative entropy, hold for our definitions. We derived suitable approximations based on data compression for these uncomputable notions. Finally, we tested them on real data in order to have a comparison with techniques which existed in literature, but were intuitively described. These computable methods are not conceived to outperform existing methods in the field: the main aim of this work is to give a contribution in expanding the relations between classical and algorithmic information theory.

References

[1] C. E. Shannon, “A Mathematical Theory of Communication”, *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, 1948.

[2] A.N. Kolmogorov, “Three Approaches to the Quantitative Definition of Information,” *Problems of Information Transmission*, vol. 1, no. 1, pp. 1–7, 1965.

[3] G.J. Chaitin, “On the length of programs for computing finite binary sequences”, *J. Assoc. Comput. Mach*., no. 16, pp. 145-159, 1969.

[4] R. Solomonoff, “A Formal Theory of Inductive Inference”, *Information and Control*, vol. 7, no. 1, pp. 1-22, 1964.

[5] M. Li and P.M.B. Vitányi, “An Introduction to Kolmogorov Complexity and Its Applications”, Chapters 2 and 8, *Springer*, 1997.

[6] M. Li, X. Chen, X. Li, B. Ma, and P.M.B. Vitányi, “The Similarity Metric”, *IEEE Trans. Inf. Theory*, vol. 50, No. 12, pp. 3250-3264, 2004.

[7] D. Benedetto, E. Caglioti, and V. Loreto, “Language trees and zipping”, *Phys. Rev. Lett.,* vol. 88, no.4, 2002.

[8] N. K. Vereshchagin and P. Vitányi, “Kolmogorov’s structure functions and model selection”, *IEEE Trans. Inf. Theory,* vol. 50*,* pp. 3265-3290, 2004.

[9] X. Chen, B. Francia, M. Li, B. McKinnon and A. Seker, “Shared Information and Program Plagiarism detection”, *IEEE Trans. Inf. Theory*, vol. 50, no. 7, pp. 1545-1551, July 2004.

[10] D. Cerra and M. Datcu, “A Model Conditioned Data Compression Based Similarity Measure”, *DCC 2008*, p. 509.

[11] R. Cilibrasi and P.M.B. Vitányi, “Clustering by Compression”, *IEEE Trans. Inf. Theory*, vol. 51, no. 4, pp. 1523-1545, 2005.

[12] E. J. Keogh, S. Lonardi, C. Ratanamahatana, “Towards Parameter-Free Data Mining”, *SIGKDD 2004*.

[13] G. J. Chaitin, “Algorithmic information theory”, *IBM J. Research Develop.* n. 21, pp. 350-359, 496, 1977.

[14] David J.C. MacKay, “Information Theory, Inference, and Learning Algorithms”, Chapter 2, *Cambridge University Press*, 2003.

[15] S. Kullback and R. A. Leibler, “On Information and Sufficiency”, *Annals of Mathematical Statistics,* vol. 22, No. 1, pp. 79-86, 1951.

[16] Welch, T. A., "A technique for high-performance data compression", *Computer*, vol. 17, pp. 8-19, 1984.

[17] http://www.liberliber.it.

[18] D. Cerra and M. Datcu, “Algorithmic Information Theory based Analysis of Earth Observation Images: an Assessment”, *IEEE Geosci. Rem. Sens. Letters*, 2009, in press.