Exact Expression For Information Distance

Vitányi, P.M.B.

Publication date
2014
Document Version
Submitted manuscript

Exact Expression For Information Distance [1]

Paul M.B. Vitányi

Abstract

Information distance can be defined not only between two strings but also in a finite multiset of strings of cardinality greater than two. We give an elementary proof for expressing the information distance in conditional Kolmogorov complexity. It is exact since the lower bound equals the upper bound up to a constant additive term.

Index Terms— Information distance, multiset, Kolmogorov complexity, similarity, pattern recognition, data mining.

I. Introduction

In pattern recognition, learning, and data mining the shortest binary program to compute from one object to another object and vice versa expresses the amount of information that separates the objects. Normalized in the appropriate manner it quantifies a similarity between objects [3]. Extending this approach we can ask how much the objects in a set of objects are alike, that is, the common information they share. All objects we discuss are represented as finite binary strings. We use Kolmogorov complexity [2], Informally, the Kolmogorov complexity of a string is the length of a shortest binary program from which the string can be computed. Therefore it is a lower bound on the length of a compressed version of that string for any current or future computer. The text [5] introduces the notions, develops the theory, and presents applications.

We write *string* to denote a finite binary string. Other finite objects, such as pairs of strings, may be encoded into strings in natural ways. The length of a string x is denoted by $|x|$. Let X be a finite multiset (a set where each member can occur more than once) of strings ordered length-increasing lexicographic. In this paper $|X| \geq 2$. Examples are $X = \{x, x\}$ and $X = \{x, y\}$ with $x \neq y$.

Let U be a fixed universal prefix Turing machine for which the programs are binary. The prefix property involved guaranties that set of programs is a prefix code (no program is a proper prefix of

Paul Vitányi is with the Center for Mathematics and Computer Science (CWI), and the University of Amsterdam. Address: CWI, Science Park 123, 1098XG Amsterdam, The Netherlands. Email: Paul.Vitanyi@cwi.nl.

another program). Since computability is involved, such a program is called *self-delimiting*. The minimal length of a self-delimiting program computing a string x is the *prefix Kolmogorov complexity* $K(x)$ of that string. We can define $K(X)$ as the length of a shortest self-delimiting program p computing all the members of X and a means to tell them apart. Similarly we define $K(X|x)$. The quantity $ID(X) = \min\{|p| : U(p,x) = X \text{ for all } x \in X\}$. (We also denote $ID(X)$ by $E_{\max}(X)$.)

A. Related Work

In the seminal [1] the information distance $ID(x,y)$ between pairs of strings x and y was introduced as the length of a shortest binary program p for the reference universal prefix Turing machine U such that $U(p,x) = y$ and $U(p,y) = x$. It was shown that $ID(x,y) = \max\{K(x|y), K(y|x)\} + O(\log \max\{K(x|y), K(y|x)\})$. In [6] it was shown how to reduce the $O(\log \max\{K(x|y), K(y|x)\})$ additive term to $O(1)$. In [4] the information distance $ID(x_1,\ldots,x_n)$ between a multiset of strings $(x_1,\ldots,x_n)$ was introduced as the length of a shortest binary program p for U such that $U(p,x_i,j) = x_j$ for all $1 \leq i,j \leq n$. It was shown that $ID(x_1,\ldots,x_n) = \max_{1\leq i\leq n} K(x_1,\ldots,x_n|x_i) + O(\log n)$. Obviously, $ID(x_1,\ldots,x_n)$ equals $|p|$ such that $U(p,x_j) = (x_1,\ldots,x_n)$ for any j ($1 \leq j \leq n$) but for the added task of computing a member of $(x_1,\ldots,x_n)$ which takes at most $K(j)+O(1)$ bits extra. (The proof ignores this quantity anyway.) Note that this also reduces the $O(\log \max\{K(x|y), K(y|x)\})$ additive term to $O(1)$ for $n = 2$. In [10] information distance is made uniform by denoting $X = (x_1,\ldots,x_n)$ and defining $ID(X)$ as the length of a shortest program to compute X from any $x \in X$. If a program computes from every $x \in X$ to any $y \in X$ then it must compute X on the way. We thus define $ID(X)$ as the length of a shortest binary program computing X from any $x \in X$. One can indicate y by its index in X.

Related is the following. The *mutual information* $I(x,y)$ between x and y is defined by $I(x:y) = K(x) + K(y) - K(x,y)$. Let $|X| = n$. In all the above cases the shortest programs p_i to compute X from $x_i \in X$ with $|p_i| = K(X|x_i)$ can be made maximally overlapping in the sense that for all $i \neq j$ the mutual information $I(p_i : p_j)$ is maximal ($1 \leq i,j \leq n$). In [10] that maximum overlap of shortest programs computing X from any $x \in X$ is determined. For $|X| = 2$ reference [1] asked whether we can find shortest programs that are minimally overlapping in the sense that for all $i \neq j$ it holds that $I(p_i : p_j)$ is minimal ($1 \leq i,j \leq n$)? In [9] this question is resolved as follows. For all strings x, y there are binary programs p, q such that $U(p,x) = y$, $U(q,y) = x$, the length of p is $K(y|x)$, the length of q is $K(x|y)$, and $I(p:q) = 0$ where the last three inequalities hold up to an additive $O(\log K(x,y))$ term. In contrast,

for some strings x, y this is not the case when we replace $O(K(x,y))$ with $O(\log(K(x|y)+K(y|x)))$. In [7] the surprising fact is shown that there is a shortest p to compute x from y such that $K(p|x) = O(\log n)$ and $K(x|p,y) = O(\log n)$. That is, this shortest program depends only on x and almost nothing on y. This is an analogue of the Slepian-Wolf result [8] in information theory.

B. Results

Let X be a multiset of strings of finite cardinality greater or equal two. The information distance of X is $ID(X)$ and can be viewed as a diameter of X. For $|X| = 2$ it is a conventional distance between the two members of X. Since the 1990s it was perceived as a nuisance and a flaw that equality between $ID(X)$ and $\max_{x\in X}\{K(X|x)\}$ held only up to a (possibly) logarithmic additive term. We give an elementary proof that for all X holds $ID(X) = \max_{x\in X}\{K(X|x)\}$ plus a constant additive term.

II. The Exact Expression

Theorem 2.1: Let X be a finite multiset of strings and $|X| \geq 2$. Then $ID(X) = \max_{x\in X}\{K(X|x)\} + O(1)$.

Proof:

($\leq$) Let $x_0 \in X$ be a fixed member of X, for example x_0 is the first member of X in lexicographic length-increasing order. Define $k = \max_{x\in X}\{K(X|x)\}$. Then $K(X|x_0) \leq k$. Computably enumerate all Y such that $x_0 \in Y$ and $K(Y|x_0) \leq k$. That is, there is a self-delimiting program p_Y of at most k bits such that p_Y with input x_0 computes output Y. Denote the set of such Y by $\mathcal{Y}$, and the set of p_Y by P. By construction $X \in \mathcal{Y}$, and $p_Y \neq p_Z$ for $Y, Z \in \mathcal{Y}$ and $Y \neq Z$. Define a bipartite graph $G = (V, E)$ with V the vertices and E the edges by

$$
\begin{aligned}
V_1 &= \{Y : Y \in \mathcal{Y}\},\\
V_2 &= \{y : y \in Y \in \mathcal{Y}\},\\
V &= V_1 \bigcup V_2,\\
E &= \{(Y, y) : Y \in V_1,\ y \in V_2\}.
\end{aligned}
$$

We label the edges in E by strings with substrings in P. The labeling satisfies (i) all edges incident with the same vertex in V_1 are labeled with strings with the second self-delimiting substrings identical, and (ii) different vertices in V_1 are labeled with strings of which the second self-delimiting substrings are

different. Conditions (i) and (ii) together imply that all edges incident with the same vertex in V_2 are labeled with strings of which the second self-delimiting substrings are different.

For each $Y \in \mathcal{Y}$ prefix p_Y associated with each edge $(Y, y) \in E$ ($y \in Y$) with an $O(1)$-length self-delimiting program r that makes the universal Turing machine U interpret p_Y as the program to compute Y from y. In this way $|rp_Y| \leq k + O(1)$. Pad rp_Y with nonsignificant 0's ending with a 1 to make a total of 3 concatenated self-delimiting programs. The concatenation is $s_Y = rp_Y 0^{k-|rp_Y|-1+c}1$ where c ($|c| = O(1)$) is a positive constant such that $|s_Y| - k$ is as small as possible but nonnegative. A self-delimiting description of c is included in r. Program r also tells U that s_Y is the concatenation of three self-delimiting programs, to ignore the final padding of nonsignificant 0's ending with a 1, and to retrieve k from $|s_Y|$. Labeling each edge (Y, y) with s_Y satisfies conditions (i) and (ii).

The length of s_X is an upper bound on $ID(X)$. The program s_X computes output X on inputs consisting of *any* $x \in X$. The program s_X works as follows. The universal prefix Turing machine U unpacks the first self-delimiting program r from s_X. This r first retrieves k from $|s_X|$ and generates $\mathcal{Y}$ and G and labels the edges of G until it labels an edge by s_X and incident on vertex x. Since the second self-delimiting substring p_X of s_X is unique for edges (X, y) with $y \in X$ the program r using x finds edge (X, x) and therefore X. Since $|s_X| = k + O(1)$, this implies the $\leq$ side.

($\geq$) By definition. ■

Corollary 2.2: For $|X| = 2$ the theorem shows the result of [1, Theorem 3.3] with error term $O(1)$ instead of $O(\log \max_{x \in X}\{K(X|x)\})$. That is, setting $X = \{x, y\}$ the theorem computes x from y and y from x with the same program of length $\max_{x \in X}\{K(X|x)\} + O(1)$ instead of $\max_{x \in X}\{K(X|x)\} + O(\max_{x \in X}\{K(X|x)\})$. (One simply adds to program r "the other one" in $O(1)$ bits.) This result can also be derived from [4], [6]. Admittedly the maximal overlap overlap property may cause the logarithmic additive term above. But for a long time it was thought that that term was necessary also without maximal overlap.

Corollary 2.3: For $|X| = n \geq 2$ (but less than infinity) the theorem shows that in [4, Theorem 2] the $O(\log n)$ additive term can be replaced by $O(1)$. (Incidentally, the maximal overlap property in [10, Theorem 3.1] seems to require an additive term of $O(\log \max_{x \in X}\{K(X|x)\})$.) To return a $y \in X$ we have to give its position in at most an additive $\log n$ bits (we know X and therefore n). In the proof of [4, Theorem 2] the $O(\log n)$ additive term does not include this quantity.

REFERENCES

[1] C.H. Bennett, P. Gács, M. Li, P.M.B. Vitányi, and W. Zurek. Information distance, *IEEE Trans. Inform. Theory*, 44:4(1998), 1407–1423.

[2] A.N. Kolmogorov, Three approaches to the quantitative definition of information, *Problems Inform. Transmission* 1:1(1965), 1–7.

[3] M. Li, X. Chen, X. Li, B. Ma, P.M.B. Vitanyi, The similarity metric, *IEEE Trans. Inform. Theory*, 50:12(2004), 3250-3264.

[4] M. Li, C. Long, B. Ma, X. Zhu, Information shared by many objects, Proc. 17th ACM Conf. Information and Knowledge Management, 2008, 1213–1220.

[5] M. Li and P.M.B. Vitányi. *An Introduction to Kolmogorov Complexity and its Applications*, Springer-Verlag, New York, Third edition, 2008.

[6] Mahmud, M.M.H.: On Universal Transfer Learning, *Theor. Comput. Sci.*, 410(2009), 1826–1846.

[7] An.A. Muchnik, Conditional complexity and codes, *Theor. Comput. Sci.*, 271(2002), 97–109.

[8] D. Slepian, J.K. Wolf, Noiseless coding of correlated information sources, *IEEE Trans. Inform. Theory*, 19(1973), 471–480.

[9] N.K. Vereshchagin and M.V. Vyugin, Independent minimum length programs to translate between given strings, *Theor. Comput. Sci.*, 271:1–2(2002), 131–143.

[10] P.M.B. Vitanyi, Information distance in multiples, *IEEE Trans. Inform. Theory*, 57:4(2011), 2451-2456.