A Short Introduction to Kolmogorov Complexity

Volker Nannen

April, 2003

Abstract

This is a short introduction to Kolmogorov complexity and information theory. It covers the concepts that were essential to my master's thesis [Nan03][1] on machine learning. The interested reader is referred to the literature, especially the textbooks [CT91] and [LV97] which cover the fields of information theory and Kolmogorov complexity in depth and with all the necessary rigor. They are well to read and require only a minimum of prior knowledge.

Kolmogorov complexity. The concept of Kolmogorov complexity was developed independently and with different motivation by Andrei N. Kolmogorov [Kol65], Ray Solomonoff [Sol64] and Gregory Chaitin [Cha66], [Cha69].[2]

The Kolmogorov complexity $C(s)$ of any binary string $s \in \{0,1\}^n$ is the length of the shortest computer program s^* that can produce this string on the Universal Turing Machine UTM and then halt. In other words, on the UTM $C(s)$ bits of information are needed to encode s. The UTM is not a real computer but an imaginary reference machine. We don't need the specific details of the UTM. As every Turing machine can be implemented on every other one, the minimum length of a program on one machine will only add a constant to the minimum length of the program on every other machine. This constant is the length of the implementation of the first machine on the other machine and is independent of the string in question. This was first observed in 1964 by Ray Solomonoff.

$C(\cdot)$

UTM

Experience has shown that every attempt to construct a theoretical model of computation that is more powerful than the Turing machine has come up with something that is at the most just as strong as the Turing machine. This has been codified in 1936 by Alonzo Church as Church's Thesis: the class of algorithmically computable numerical functions coincides with the class of partial recursive functions. Everything we can compute we can compute by a Turing machine and what we cannot compute by a Turing machine we cannot compute at all. This said, we can use Kolmogorov complexity as a universal measure that will assign the same value to any sequence of bits regardless of the model of computation, within the bounds of an additive constant.

[1] online available at http://volker.nannen.com/work/mdl/

[2] *Kolmogorov complexity* is sometimes also called *algorithmic complexity* and *Turing complexity.* Though Kolmogorov was not the first one to formulate the idea, he played the dominant role in the consolidation of the theory.

Incomputability of Kolmogorov complexity. Kolmogorov complexity is not computable. It is nevertheless essential for proving existence and bounds for weaker notions of complexity. The fact that Kolmogorov complexity cannot be computed stems from the fact that we cannot compute the output of every program. More fundamentally, no algorithm is possible that can predict of every program if it will ever halt, as has been shown by Alan Turing in his famous work on the halting problem [Tur36]. No computer program is possible that, when given any other computer program as input, will always output `true` if that program will eventually halt and `false` if it will not. Even if we have a short program that outputs our string and that seems to be a good candidate for being the shortest such program, there is always a number of shorter programs of which we do not know if they will ever halt and with what output.

Plain versus prefix complexity. Turing's original model of computation included special delimiters that marked the end of an input string. This has resulted in two brands of Kolmogorov complexity:

- **plain Kolmogorov complexity:** the length $C(s)$ of the shortest binary string that is delimited by special marks and that can compute x on the UTM and then halt. $C(\cdot)$
- **prefix Kolmogorov complexity:** the length $K(s)$ of the shortest binary string that is *self-delimiting* [LV97] and that can compute x on the UTM and then halt. $K(\cdot)$

The difference between the two is logarithmic in $C(s)$: the number of extra bits that are needed to delimit the input string. While plain Kolmogorov complexity integrates neatly with the Turing model of computation, prefix Kolmogorov complexity has a number of desirable mathematical characteristics that make it a more coherent theory. The individual advantages and disadvantages are described in [LV97]. Which one is actually used is a matter of convenience. We will mostly use the prefix complexity $K(s)$.

Individual randomness. A. N. Kolmogorov was interested in Kolmogorov complexity to define the individual randomness of an object. When s has no computable regularity it cannot be encoded by a program shorter than s. Such a string is truly random and its Kolmogorov complexity is the length of the string itself plus the commando `print`[3]. And indeed, strings with a Kolmogorov complexity close to their actual length satisfy all known tests of randomness. A regular string, on the other hand, can be computed by a program much shorter than the string itself. But the overwhelming majority of all strings of any length are random and for a string picked at random chances are exponentially small that its Kolmogorov complexity will be significantly smaller than its actual length.

This can easily be shown. For any given integer n there are exactly 2^n binary strings of that length and $2^n - 1$ strings that are shorter than n: one empty string, 2^1 strings of length one, 2^2 of length two and so forth. Even if all strings shorter than n would produce a string of length n on the UTM we would still

[3] Plus a logarithmic term if we use prefix complexity

be one string short of assigning a $C(s) < n$ to every single one of our 2^n strings. And if we want to assign a $C(s) < n-1$ we can maximally do so for $2^{n-1}-1$ strings. And for $C(s) < n-10$ we can only do so for $2^{n-10}-1$ strings which is less than 0.1% of all our strings. Even under optimal circumstances we will never find a $C(s) < n-c$ for more than $\frac{1}{2^c}$ of our strings.

Conditional Kolmogorov complexity. The conditional Kolmogorov complexity $K(s|a)$ is defined as the shortest program that can output s on the UTM if the input string a is given on an auxiliary tape. $K(s)$ is the special case $K(s|\epsilon)$ where the auxiliary tape is empty. $K(\cdot|\cdot)$

The universal distribution. When Ray Solomonoff first developed Kolmogorov complexity in 1964 he intended it to define a universal distribution over all possible objects. His original approach dealt with a specific problem of Bayes' rule, the unknown prior distribution. Bayes' rule can be used to calculate $P(m|s)$, the probability for a probabilistic model to have generated the sample s, given s. It is very simple. $P(s|m)$, the probability that the sample will occur given the model, is multiplied by the unconditional probability that the model will apply at all, $P(m)$. This is divided by the unconditional probability of the sample $P(s)$. The unconditional probability of the model is called the prior distribution and the probability that the model will have generated the data is called the posterior distribution.

$$P(m|s) = \frac{P(s|m)\,P(m)}{P(s)} \tag{1}$$

Bayes' rule can easily be derived from the definition of conditional probability:

$$P(m|s) = \frac{P(m,s)}{P(s)} \tag{2}$$

and

$$P(s|m) = \frac{P(m,s)}{P(m)} \tag{3}$$

The big and obvious problem with Bayes' rule is that we usually have no idea what the prior distribution $P(m)$ should be. Solomonoff suggested that if the true prior distribution is unknown the best assumption would be the universal distribution $2^{-K(m)}$ where $K(m)$ is the prefix Kolmogorov complexity of the model[4]. This is nothing but a modern codification of the age old principle that is wildly known under the name of Occam's razor: the simplest explanation is the most likely one to be true.

Entropy. Claude Shannon [Sha48] developed information theory in the late 1940's. He was concerned with the optimum code length that could be given to different binary words w of a source string s. Obviously, assigning a short code

[4] Originally Solomonoff used the plain Kolmogorov complexity $C(\cdot)$. This resulted in an improper distribution $2^{-C(m)}$ that tends to infinity. Only in 1974 L.A. Levin introduced prefix complexity to solve this particular problem, and thereby many other problems as well [Lev74].

length to low frequency words or a long code length to high frequency words is a waste of resources. Suppose we draw a word w from our source string s uniformly at random. Then the probability $p(w)$ is equal to the frequency of w in s. Shannon found that the optimum overall code length for s was achieved when assigning to each word w a code of length $-\log p(w)$. Shannon attributed the original idea to R.M. Fano and hence this code is called the Shannon-Fano code. When using such an optimal code, the average code length of the words of s can be reduced to

$$H(s) \ = \ -\sum_{w \in s} p(w) \log p(w) \tag{4}$$

where $H(s)$ is called the entropy of the set s. When s is finite and we assign a code of length $-\log p(w)$ to each of the n words of s, the total code length is $H(\cdot)$

$$-\sum_{w \in s} \log p(w) \ = \ n\, H(s) \tag{5}$$

Let s be the outcome of some random process W that produces the words $w \in s$ sequentially and independently, each with some known probability $p(W = w) > 0$. $K(s|W)$ is the Kolmogorov complexity of s given W. Because the Shannon-Fano code is optimal, the probability that $K(s|W)$ is significantly less than $nH(W)$ is exponentially small. This makes the negative log likelihood of s given W a good estimator of $K(s|W)$:

$$\begin{aligned} K(s|W) \ &\approx \ n\, H(W) \\ &\approx \ \sum_{w \in s} \log p(w|W) \\ &= \ -\log p(s|W) \end{aligned} \tag{6}$$

Relative entropy. The relative entropy $D(p||q)$ tells us what happens when we use the wrong probability to encode our source string s. If $p(w)$ is the true distribution over the words of s but we use $q(w)$ to encode them, we end up with an average of $H(p) + D(p||q)$ bits per word. $D(p||q)$ is also called the Kullback Leibler distance between the two probability mass functions p and q. It is defined as $D(\cdot||\cdot)$

$$D(p||q) \ = \ \sum_{w \in s} p(w) \log \frac{p(w)}{q(w)} \tag{7}$$

Fisher information. Fisher information was introduced into statistics some 20 years before C. Shannon introduced information theory [Fis25]. But it was not well understood without it. Fisher information is the variance of the score V of the continuous parameter space of our models m_k. This needs some explanation. At the beginning of this thesis we defined models as binary strings that discretize the parameter space of some function or probability distribution. For the purpose of Fisher information we have to temporarily treat a model m_k as a vector in $\mathbb{R}^k$. And we only consider models where for all samples s the mapping $f_s(m_k)$ defined by $f_s(m_k) = p(s|m_k)$ is differentiable. Then the score V can be defined as

$$\begin{aligned} V &= \frac{\partial}{\partial\, m_k} \ln p(s|m_k) \\ &= \frac{\frac{\partial}{\partial\, m_k}\, p(s|m_k)}{p(s|m_k)} \end{aligned} \tag{8}$$

The score V is the partial derivative of ln $p(s|m_k)$, a term we are already familiar with. The Fisher information $J(m_k)$ is $J(\cdot)$

$$J(m_k) \;=\; E_{m_k}\left[\frac{\partial}{\partial\, m_k} \ln p(s|m_k)\right]^2 \tag{9}$$

Intuitively, a high Fisher information means that slight changes to the parameters will have a great effect on $p(s|m_k)$. If $J(m_k)$ is high we must calculate $p(s|m_k)$ to a high precision. Conversely, if $J(m_k)$ is low, we may round $p(s|m_k)$ to a low precision.

Kolmogorov complexity of sets. The Kolmogorov complexity of a set of strings $\mathcal{S}$ is the length of the shortest program that can output the members of $\mathcal{S}$ on the UTM and then halt. If one is to approximate some string s with $\alpha < K(s)$ bits then the best one can do is to compute the smallest set $\mathcal{S}$ with $K(\mathcal{S}) \leq \alpha$ that includes s. Once we have some $\mathcal{S} \ni s$ we need at most $\log|\mathcal{S}|$ additional bits to compute s. This set $\mathcal{S}$ is defined by the Kolmogorov structure function $h_s(\cdot)$

$$h_s(\alpha) \;=\; \min_{\mathcal{S}} \big[\log|\mathcal{S}| : \mathcal{S} \ni s,\ K(\mathcal{S}) \leq \alpha\big] \tag{10}$$

which has many interesting features. The function $h_s(\alpha) + \alpha$ is non increasing and never falls below the line $K(s) + O(1)$ but can assume any form within these constraints. It should be evident that

$$h_s(\alpha) \;\geq\; K(s) - K(\mathcal{S}) \tag{11}$$

Kolmogorov complexity of distributions. The Kolmogorov structure function is not confined to finite sets. If we generalize $h_s(\alpha)$ to probabilistic models m_p that define distributions over $\mathbb{R}$ and if we let s describe a real number, we obtain

$$h_s(\alpha) \;=\; \min_{m_p} \big[-\log p(s|m_p) : p(s|m_p) > 0,\ K(m_p) \leq \alpha\big] \tag{12}$$

where $-\log p(s|m_p)$ is the number of bits we need to encode s with a code that is optimal for the distribution defined by m_p. Henceforth we will write m_p when the model defines a probability distribution and m_k with $k \in \mathbb{N}$ when the model defines a probability distribution that has k parameters. A set $\mathcal{S}$ can be viewed as a special case of m_p, a uniform distribution with

$$p(s|m_p) \;=\; \begin{cases} \frac{1}{|\mathcal{S}|} & \text{if } \; s \in \mathcal{S} \\ 0 & \text{if } \; s \notin \mathcal{S} \end{cases} \tag{13}$$

Minimum randomness deficiency. The randomness deficiency of a string s with regard to a model m_p is defined as $\delta(\cdot|m_p)$

$$\delta(s|m_p) \;=\; -\log p(s|m_p) - K(\, s|m_p,\, K(m_p)\,) \tag{14}$$

for $p(s) > 0$, and ∞ otherwise. This is a generalization of the definition given in [VV02] where models are finite sets. If $\delta(s|m_p)$ is small, then s may be considered a *typical* or *low profile* instance of the distribution. s satisfies *all* properties of low Kolmogorov complexity that hold with high probability for the support set of m_p. This would not be the case if s would be exactly identical to the mean, first momentum or any other special characteristic of m_p.

Randomness deficiency is a key concept to any application of Kolmogorov complexity. As we saw earlier, Kolmogorov complexity and conditional Kolmogorov complexity are not computable. We can never claim that a particular string s does have a conditional Kolmogorov complexity

$$K(s|m_p) \;\approx\; -\log p(s|m_p) \tag{15}$$

The technical term that defines all those strings that do satisfy this approximation is *typicality*, defined as a small randomness deficiency $\delta(s|m_p)$. typicality

Minimum randomness deficiency turns out to be important for lossy data compression. A compressed string of minimum randomness deficiency is the most difficult one to distinguish from the original string. The best lossy compression that uses a maximum of α bits is defined by the minimum randomness deficiency function $\beta_s(\cdot)$

$$\beta_s(\alpha) \;=\; \min_{m_p} \left[\delta(s|m_p) : p(s|m_p) > 0,\; K(m_p) \leq \alpha\right] \tag{16}$$

Minimum Description Length. The Minimum Description Length or short MDL of a string s is the length of the shortest two-part code for s that uses less than α bits. It consists of the number of bits needed to encode the model m_p that defines a distribution and the negative log likelihood of s under this distribution. MDL $\lambda_s(\cdot)$

$$\lambda_s(\alpha) \;=\; \min_{m_p} \left[\, -\log p(s|m_p) + K(m_p) : p(s|m_p) > 0,\; K(m_p) \leq \alpha\right] \tag{17}$$

It has recently been shown by Nikolai Vereshchagin and Paul Vitányi in [VV02] that a model that minimizes the description length also minimizes the randomness deficiency, though the reverse may not be true. The most fundamental result of that paper is the equality

$$\beta_s(\alpha) \;=\; h_s(\alpha) + \alpha - K(s) \;=\; \lambda_s(\alpha) - K(s) \tag{18}$$

where the mutual relations between the Kolmogorov structure function, the minimum randomness deficiency and the minimum description length are pinned down, up to logarithmic additive terms in argument and value.

References

[Cha66] Gregory J. Chaitin. On the length of programs for computing finite binary sequences. *J. Assoc. Comput. Mach.*, 13:547–569, 1966.

[Cha69] Gregory J. Chaitin. On the length of programs for computing finite binary sequences: statistical considerations. *J. Assoc. Comput. Mach.*, 16:145–159, 1969.

[CT91] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory.* John Wiley & Sons, Inc., New York, 1991.

[Fis25] R.A. Fisher. Theory of statistical estimation. *Proc. Cambridge Phil. Society*, 22:700–725, 1925.

[Kol65] Andrei Nikolaevich Kolmogorov. Three approaches to the quantitative definition of information. *Problems of Information Transmission*, 1:1–7, 1965.

[Lev74] L.A. Levin. Laws of information conservation (non-growth) and aspects of the foundation of probability theory. *Problems Inform. Transmission*, 10:206–210, 1974.

[LV97] Ming Li and Paul Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications.* Springer, Berlin, second edition, 1997.

[Nan03] Volker Nannen. The Paradox of Overfitting. Master's thesis, Rijksuniversiteit Groningen, the Netherlands, April 2003.

[Sha48] Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948.

[Sol64] Ray Solomonoff. A formal theory of inductive inference, part 1 and part 2. *Information and Control*, 7:1–22, 224–254, 1964.

[Tur36] Alan M. Turing. On computable numbers with an application to the Entscheidungsproblem. *Proc. London Math. Soc., Ser. 2*, 42:230–265, 1936.

[VV02] Nikolai Vereshchagin and Paul Vitányi. Kolmogorov's Structure Functions with an Application to the Foundations of Model Selection. *Proc. 47th IEEE Symp. Found. Comput. Sci. (FOCS'02)*, 2002.

Index