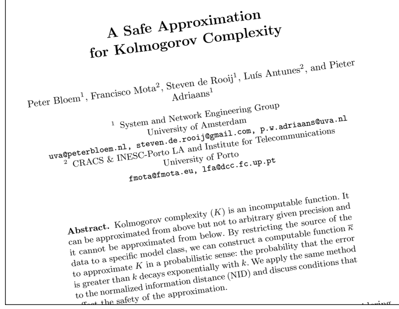
Kolmogorov complexity

approximation, separation and identification

Peter Bloem 6 July 2022 p@peterbloem.nl @pbloemesquire

VU VRIJE UNIVERSITEIT AMSTERDAM

This presentation was given in 2022 as part of the AIT & ML symposium.

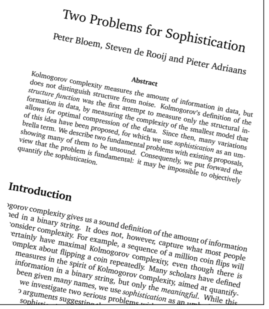


A Safe Approximation for Kolmogorov Complexity

Peter Bloem[1], Francisco Mota[2], Steven de Rooij[1], Luís Antunes[2], and Pieter Adriaans[1]

[1] System and Network Engineering Group
University of Amsterdam
uva@peterbloem.nl, steven.de.rooij@gmail.com, p.w.adriaans@uva.nl
[2] CRACS & INESC-Porto LA and Institute for Telecommunications
University of Porto
fmota@fmota.eu, lfa@dcc.fc.up.pt

Abstract. Kolmogorov complexity (K) is an incomputable function. It can be approximated from above but not to arbitrary given precision and it cannot be approximated from below. By restricting the source of the data to a specific model class, we can construct a computable function $\overline{\kappa}$ to approximate K in a probabilistic sense: the probability that the error is greater than k decays exponentially with k. We apply the same method to the normalized information distance (NID) and discuss conditions that



Two Problems for Sophistication

Peter Bloem, Steven de Rooij and Pieter Adriaans

Abstract

Kolmogorov complexity measures the amount of information in data, but does not distinguish structure from noise. Kolmogorov's definition of the *structure function* was the first attempt to measure only the structural information in data, by measuring the complexity of the smallest model that allows for optimal compression of the data. Since then, many variations of this idea have been proposed, for which we use *sophistication* as an umbrella term. We describe two fundamental problems with existing proposals, showing many of them to be unsound. Consequently, we put forward the view that the problem is fundamental: it may be impossible to objectively quantify the sophistication.

Introduction

gorov complexity gives us a sound definition of the amount of information ned in a binary string. It does not, however, capture what most people onsider complexity. For example, a sequence of a million coin flips will ertainly have maximal Kolmogorov complexity, even though there is omplex about flipping a coin repeatedly. Many scholars have defined measures in the spirit of Kolmogorov complexity, aimed at quantify- information in a binary string, but only the *meaningful*. While this been given many names, we use *sophistication* as an um we investigate two serious problems arguments suggesting

It includes content from these two papers, published in 2014 and 2015.

---

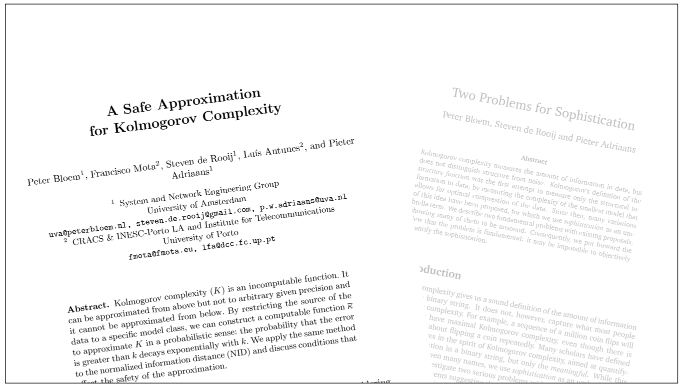
talk structure

Part one: Approximating Kolmogorov complexity

Part two: Separating structure from noise

Part three: Outlook

Approximating Kolmogorov complexity



A Safe Approximation for Kolmogorov Complexity

Peter Bloem[1], Francisco Mota[2], Steven de Rooij[1], Luís Antunes[2], and Pieter Adriaans[1]

[1] System and Network Engineering Group
University of Amsterdam
uva@peterbloem.nl, steven.de.rooij@gmail.com, p.w.adriaans@uva.nl
[2] CRACS & INESC-Porto LA and Institute for Telecommunications
University of Porto
fmota@fmota.eu, lfa@dcc.fc.up.pt

Abstract. Kolmogorov complexity (K) is an incomputable function. It can be approximated from above but not to arbitrary given precision and it cannot be approximated from below. By restricting the source of the data to a specific model class, we can construct a computable function $\overline{\kappa}$ to approximate K in a probabilistic sense: the probability that the error is greater than k decays exponentially with k. We apply the same method to the normalized information distance (NID) and discuss conditions that

Two Problems for Sophistication

Peter Bloem, Steven de Rooij and Pieter Adriaans

Abstract

Kolmogorov complexity measures the amount of information in data, but does not distinguish structure from noise. Kolmogorov's definition of the *structure function* was the first attempt to measure only the structural information in data, by measuring the complexity of the smallest model that allows for optimal compression of the data. Since then, many variations of this idea have been proposed, for which we use *sophistication* as an umbrella term. We describe two fundamental problems with existing proposals, showing many of them to be unsound. Consequently, we put forward the view that the problem is fundamental: it may be impossible to objectively antify the sophistication.

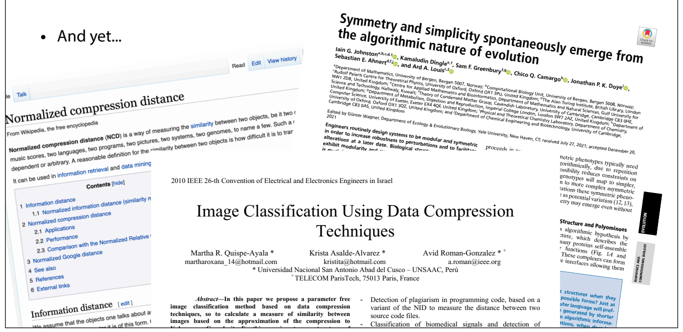
oduction

omplexity gives us a sound definition of the amount of information binary string. It does not, however, capture what most people complexity. For example, a sequence of a million coin flips will have maximal Kolmogorov complexity, even though there is about flipping a coin repeatedly. Many scholars have defined es in the spirit of Kolmogorov complexity, aimed at quantifytion in a binary string, but only the *meaningful*. While this ven many names, we use *sophistication* as an umestigate two serious problems

---

Kolmogorov complexity cannot (really) be approximated

- And yet...



Talk | Read | Edit | View history

Normalized compression distance

From Wikipedia, the free encyclopedia

Normalized compression distance (NCD) is a way of measuring the similarity between two objects, be it two music scores, two languages, two programs, two pictures, two systems, two genomes, to name a few. Such a dependent or arbitrary. A reasonable definition for the similarity between two objects is how difficult it is to tran

It can be used in information retrieval and data mining

Contents [hide]

1 Information distance
1.1 Normalized information distance (similarity m
2 Normalized compression distance
2.1 Applications
2.2 Performance
2.3 Comparison with the Normalized Relative
3 Normalized Google distance
4 See also
5 References
6 External links

Information distance [edit]

We assume that the objects one talks about a

Symmetry and simplicity spontaneously emerge from the algorithmic nature of evolution

Iain G. Johnston[a,b,c,d,1], Kamaludin Dingle[e,1], Sam F. Greenbury[f,g], Chico Q. Camargo[h], Jonathan P. K. Doye[i], Sebastian E. Ahnert[d,f,j], and Ard A. Louis[c,2]

[a]Department of Mathematics, University of Bergen, Bergen 5007, Norway; [b]Computational Biology Unit, University of Bergen, Bergen 5008, Norway; [c]Rudolf Peierls Centre for Theoretical Physics, University of Oxford, Oxford OX1 3PU, United Kingdom; [d]The Alan Turing Institute, British Library, London NW1 2DB, United Kingdom; [e]Centre for Applied Mathematics and Bioinformatics, Department of Mathematics and Natural Sciences, Gulf University for Science and Technology, Hallwaly, Kuwait; [f]Theory of Condensed Matter Group, Cavendish Laboratory, University of Cambridge, Cambridge CB3 0HE, United Kingdom; [g]Department of Metabolism, Digestion and Reproduction, Imperial College London, London SW7 2AZ, United Kingdom; [h]Department of Computer Science, University of Exeter, Exeter EX4 4QF, United Kingdom; [i]Physical and Theoretical Chemistry Laboratory, Department of Chemistry, University of Oxford, Oxford OX1 3QZ, United Kingdom; and [j]Department of Chemical Engineering and Biotechnology, University of Cambridge, Cambridge CB3 0AS, United Kingdom

Edited by Günter Wagner, Department of Ecology & Evolutionary Biology, Yale University, New Haven, CT; received July 27, 2021; accepted December 20, 2021

Engineers routinely design systems to be modular and symmetric in order to increase robustness to perturbations and to facilitate alterations at a later date. Biological structu exhibit modularity and

proceeds in

netric phenotypes typically need gorithmically, due to repetition essibility reduces constraints on genotypes will map to simpler, n to more complex asymmetric tations these symmetric phenoe as potential variation (12, 13), etry may emerge even without

Structure and Polyominoes

s algorithmic hypothesis by cture, which describes the nany proteins self-assemble r functions (Fig. 1A and These complexes can form e interfaces allowing them

c structures when they possible forms? Just as uter language will prefe generated by shorter n algorithmic informations, when

EVOLUTION

2010 IEEE 26-th Convention of Electrical and Electronics Engineers in Israel

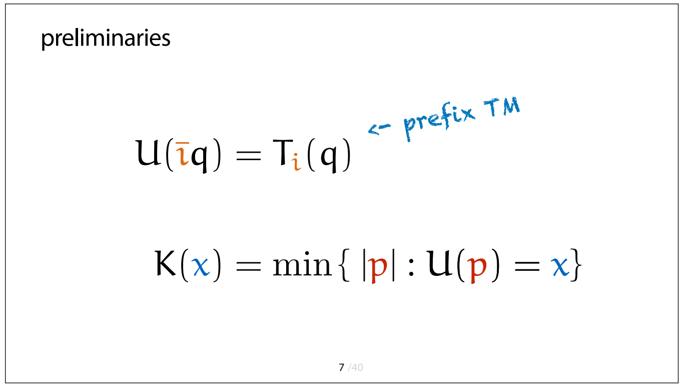
Image Classification Using Data Compression Techniques

Martha R. Quispe-Ayala * martharoxana_14@hotmail.com
Krista Asalde-Alvarez * kristita@hotmail.com
Avid Roman-Gonzalez * + a.roman@ieee.org
* Universidad Nacional San Antonio Abad del Cusco – UNSAAC, Perú
+ TELECOM ParisTech, 75013 Paris, France

***Abstract*—In this paper we propose a parameter free image classification method based on data compression techniques, so to calculate a measure of similarity between images based on the approximation of the compression to**

- Detection of plagiarism in programming code, based on a variant of the NID to measure the distance between two source code files.
- Classification of biomedical signals and detection of

More precisely, we can approach Kolmogorov complexity from above, but not to a given precision, and we cannot approximate it from below.



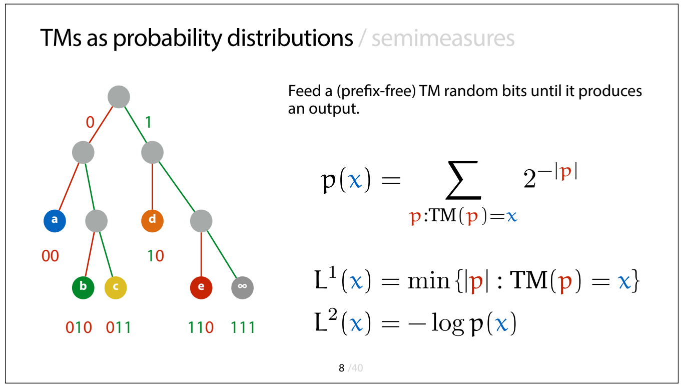
preliminaries

$$U(\bar{\imath}q) = T_i(q)$$ <- prefix TM

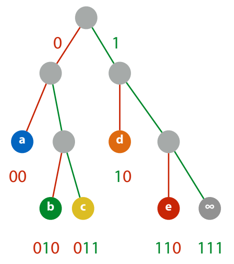
$$K(x) = \min\{\, |p| : U(p) = x\}$$

7 /40

Finite strings only, no prediction



TMs as probability distributions / semimeasures



Feed a (prefix-free) TM random bits until it produces an output.

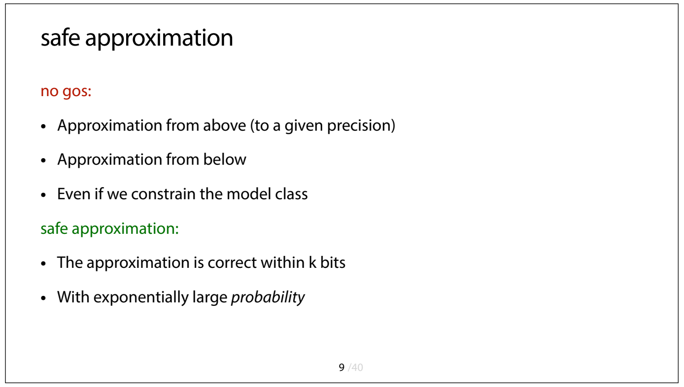
$$p(x) = \sum_{p:\mathrm{TM}(p)=x} 2^{-|p|}$$

$$L^1(x) = \min\{|p| : \mathrm{TM}(p) = x\}$$
$$L^2(x) = -\log p(x)$$

8 /40

If your TM is a universal Turing machine, then L1 and L2 correspond (up to a constant). But for other TMs, they may disagree arbitrarily much.

Taken over all TMs the set of probability distributions we can define this way (or more accurately, probability semimeasures) corresponds to the lower semicomputable semimeasures.



safe approximation

no gos:

- Approximation from above (to a given precision)
- Approximation from below
- Even if we constrain the model class

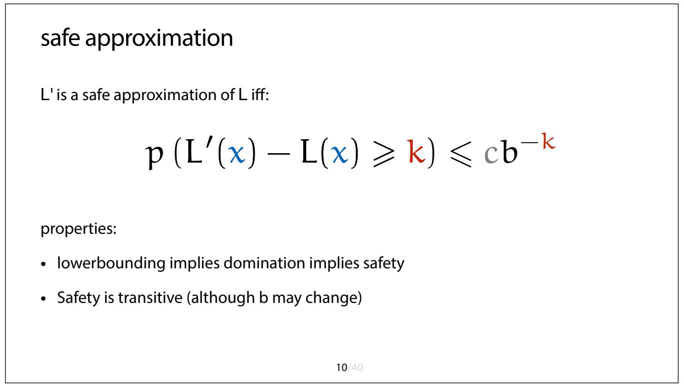
safe approximation:

- The approximation is correct within k bits
- With exponentially large *probability*

9 /40

The no-go's have long been established as approximations for K that cannot be computable.

Safe approximation is an alternative that is still possible.



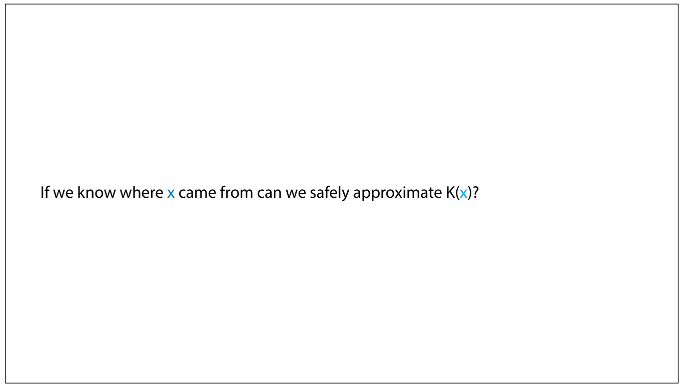
safe approximation

L' is a safe approximation of L iff:

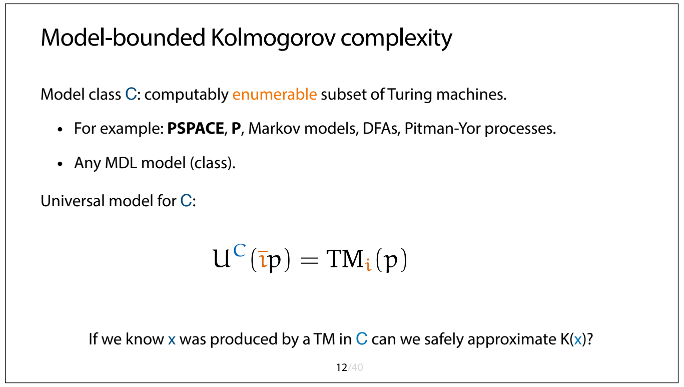
$$p\left(L'(x) - L(x) \geqslant k\right) \leqslant cb^{-k}$$

properties:

- lowerbounding implies domination implies safety
- Safety is transitive (although b may change)



If we know where x came from can we safely approximate K(x)?



Model-bounded Kolmogorov complexity

Model class C: computably enumerable subset of Turing machines.

- For example: **PSPACE**, **P**, Markov models, DFAs, Pitman-Yor processes.
- Any MDL model (class).

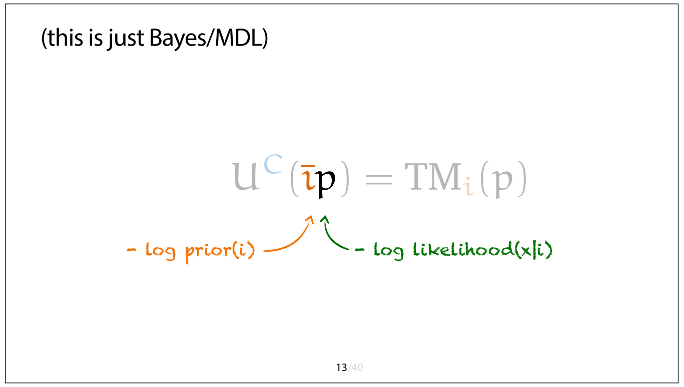
Universal model for C:

$$U^{C}(\bar{\imath}p) = TM_{i}(p)$$

If we know x was produced by a TM in C can we safely approximate K(x)?

We get the well known resource-bounded Kolmogorov complexity as a special case when we define C with a resource bound on our TMs, but really any model class commonly used in statistics can be a model class in class bounded KC. We just have to frame it in terms of a distribution on bitstrings. A lot of this work has been done already in MDL.

We lose a lot of the nice properties of KC. Most importantly, the invariance.

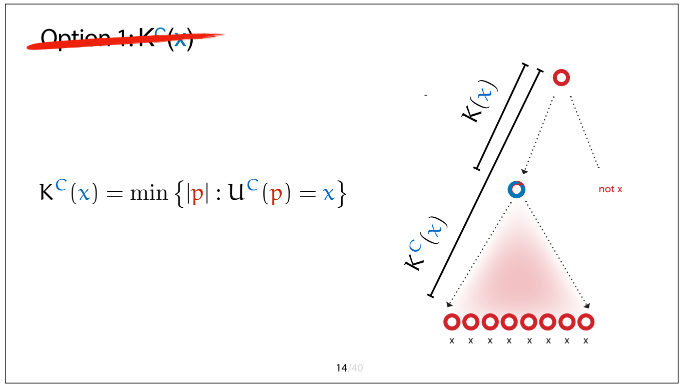


(this is just Bayes/MDL)

$$U^C(\bar{\imath}p) = TM_i(p)$$

- log prior(i)
- log likelihood(x|i)

Correspondence to Bayes



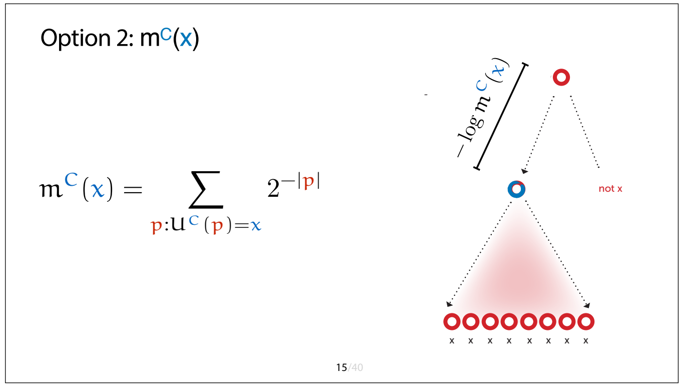
~~Option 1: $K^C(x)$~~

$$K^C(x) = \min\{|p| : U^C(p) = x\}$$

First up, the naive way to define class-bounded KC—the smallest program on our class universal TM that produces x— does not result in a safe approximation.

If we can just point to the internal node highlighted in blue, it doesn't matter what else we feed the TM, the outcome beyond that will always be x. The non-limited Kolmogorov complexity can take advantage of this fact, and find some representation that just points to this internal node. The model bounded Kolmogorov complexity cannot do this: it can only pick leaf nodes in this tree. By making the subtree below the internal bigger, we can make the difference between $K^C(x)$ and $K(x)$ arbitrarily big.

In short, the shortest program for x on U^C is not a safe approximation for the Kolmogorov complexity, even is x

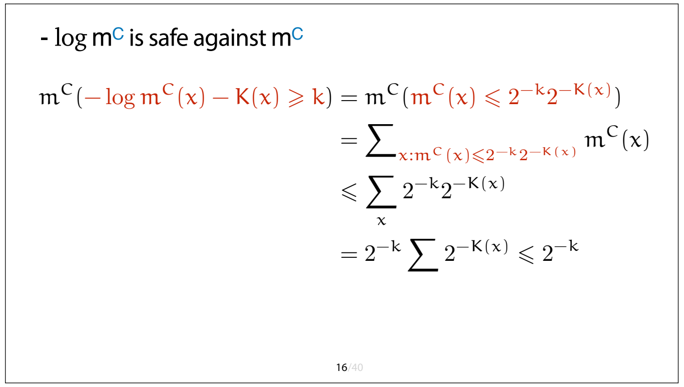


Option 2: $m^C(x)$

$$m^C(x) = \sum_{p:U^C(p)=x} 2^{-|p|}$$

Each path from the root to an x node represents a code, and with it some probability mass. Adding up all that probability mass gives us a codelength equivalent to the shorter code in the previous slide.

If we want a good approximation of K(x) we need to consider the mass of all programs on our UTM.

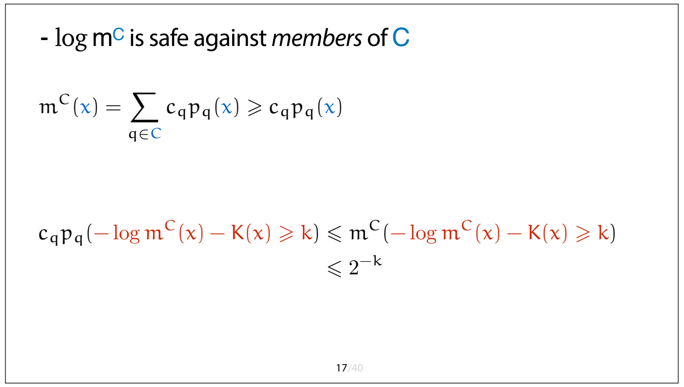


- log m^C is safe against m^C

$$
\begin{aligned}
m^C(-\log m^C(x) - K(x) \geqslant k) &= m^C(m^C(x) \leqslant 2^{-k}2^{-K(x)}) \\
&= \sum\nolimits_{x:m^C(x)\leqslant 2^{-k}2^{-K(x)}} m^C(x) \\
&\leqslant \sum_x 2^{-k}2^{-K(x)} \\
&= 2^{-k}\sum 2^{-K(x)} \leqslant 2^{-k}
\end{aligned}
$$

16/40

With this we have a safe approximation.



- log m^C is safe against *members* of C

$$m^C(x) = \sum_{q\in C} c_q p_q(x) \geqslant c_q p_q(x)$$

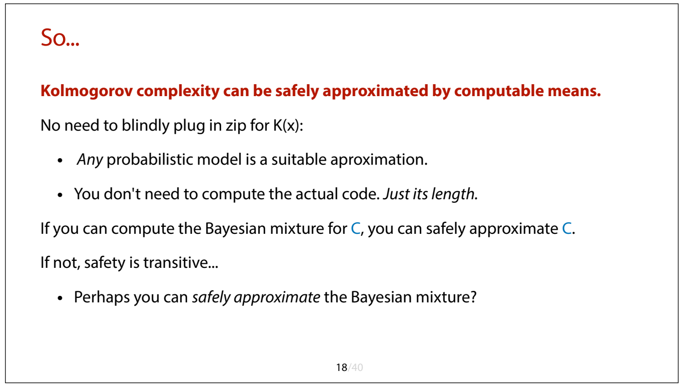
$$
\begin{aligned}
c_q p_q(-\log m^C(x) - K(x) \geqslant k) &\leqslant m^C(-\log m^C(x) - K(x) \geqslant k) \\
&\leqslant 2^{-k}
\end{aligned}
$$

17/40

We are also safe against any member of C. This is the more important property. We can now assume that our data cam from a member of C and safely approximate it.

For example, if we assume that the process that generated our data can be simulated on a polynomial-time Turing Machine, we can take P as our model class and we have shown that the Kolmogorov complexity of any such data may be safely approximated.

The approximation may be still be very expensive to compute (likely not polynomial), but it is at least computable.



So...

Kolmogorov complexity can be safely approximated by computable means.

No need to blindly plug in zip for K(x):

- *Any* probabilistic model is a suitable aproximation.
- You don't need to compute the actual code. *Just its length.*

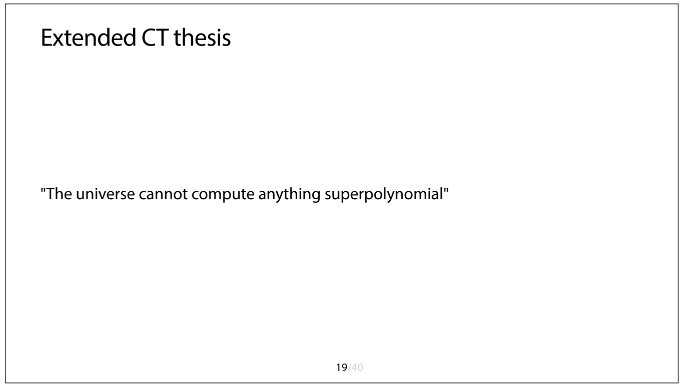
If you can compute the Bayesian mixture for C, you can safely approximate C.

If not, safety is transitive...

- Perhaps you can *safely approximate* the Bayesian mixture?

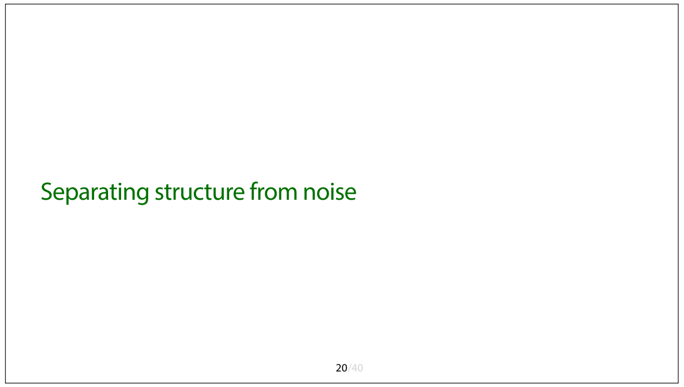
18/40

It's exponentially unlikely that we'll ever see any data for which the Kolmogorov complexity differs substantially from a good computable approximation.



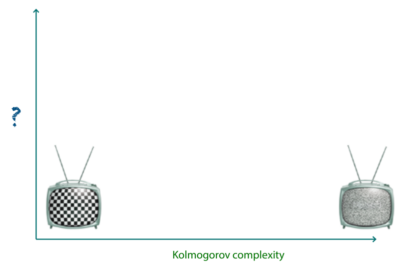
Extended CT thesis

"The universe cannot compute anything superpolynomial"



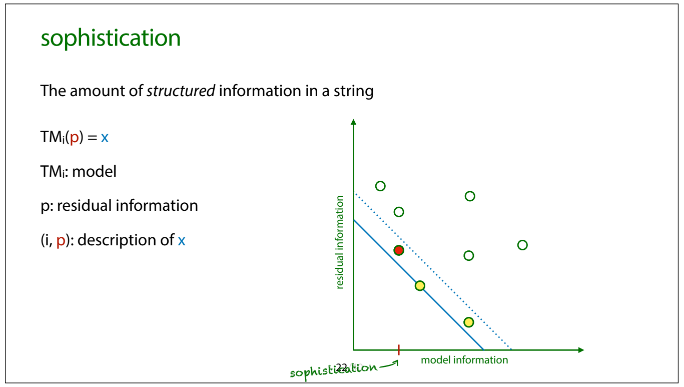
Separating structure from noise

NB: We're moving away from approximation land for this part. We're happy with uncomputable functions for the time being.



Interesting data lives somewhere between very compressible and highly uncompressible data. This is not a defining feature, however, since we can also make very uninteresting data that is medium-compressible.

Can we quantity this vertical axis? What makes data interesting. Could this be an objective quantity?



sophistication

The amount of *structured* information in a string

$TM_i(p) = x$

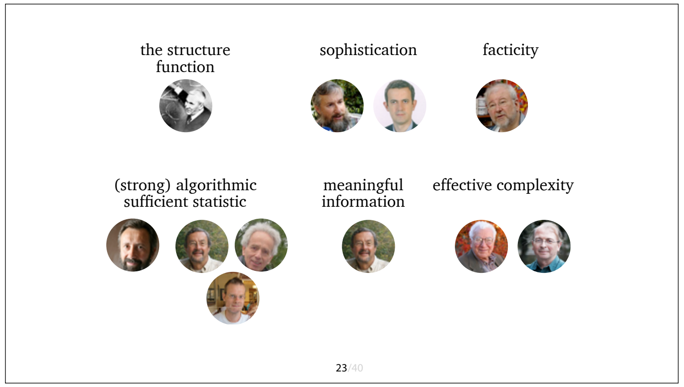
TM_i: model

p: residual information

(i, p): description of x

The broad idea of sophistication is that we can *two-part code* the string x. We first describe a TM and the the input to that TM that will cause it to produce x. If this two-part description is *optimal* (i.e. close to K(x)) then the amount of bits we spend on the TM is an indication of the interestingness of the string.

We allow a small amount of slack (indicated by the dotted line), usually some constant number of bits, and then pick the two-part representation with with the smallest model description inside this slack region.



the structure function

sophistication

facticity

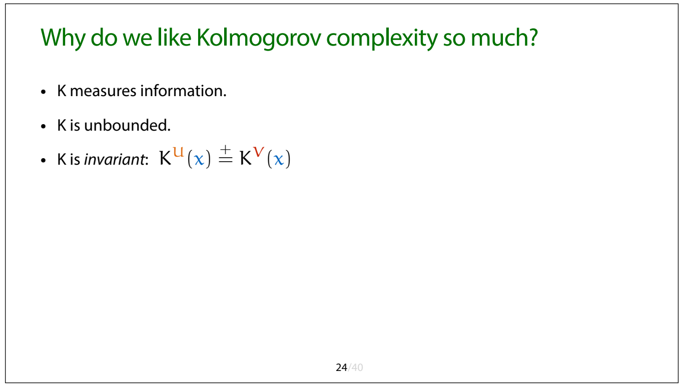
(strong) algorithmic sufficient statistic

meaningful information

effective complexity

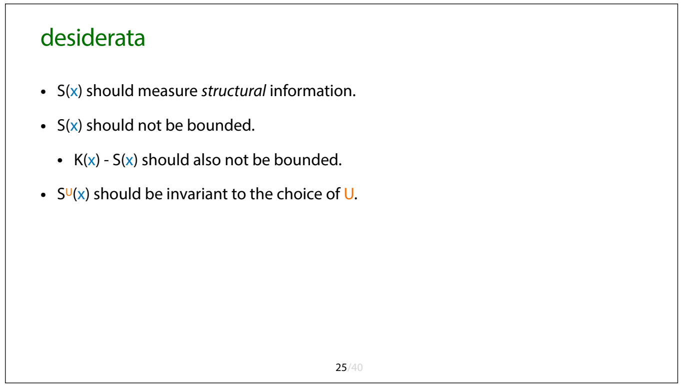
A lot of very smart and famous people have made the case for this kind of separation of structure and noise.

Despite that fact, we will take a critical perspective here.



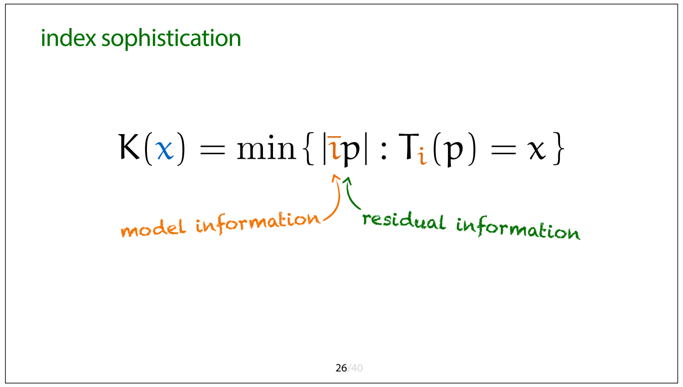
Why do we like Kolmogorov complexity so much?

- K measures information.
- K is unbounded.
- K is *invariant*: $K^{u}(x) \overset{+}{=} K^{v}(x)$



desiderata

- S(x) should measure *structural* information.
- S(x) should not be bounded.
  - K(x) - S(x) should also not be bounded.
- $S^U(x)$ should be invariant to the choice of U.



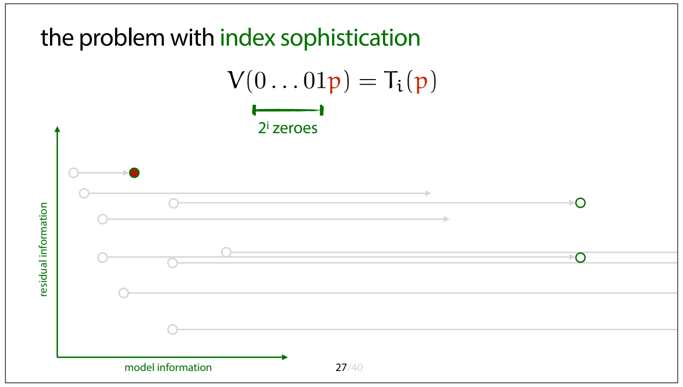
index sophistication

$$K(x) = \min\{\, |\bar{\imath}p| : T_i(p) = x \,\}$$

model information

residual information

We will start with a type of sophistication that goes catastrophically wrong very quickly.



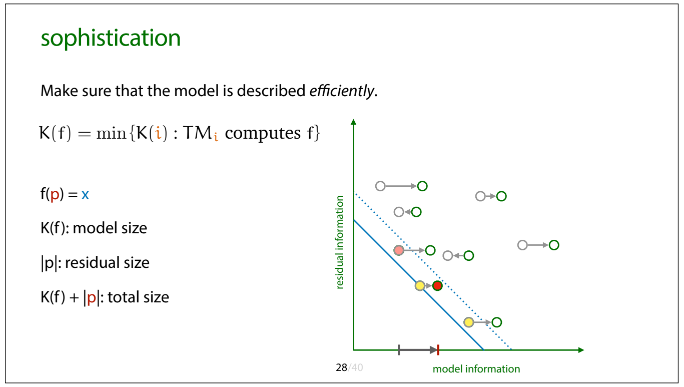
the problem with index sophistication

$$V(\underbrace{0\ldots0}_{2^i \text{ zeroes}}1p) = T_i(p)$$

With index sophistication, picking a bad index representation causes the structural information in every string to blow up massively.

Note that the resulting K(x) is still efficient. We just have to describe a UTM with a better index representation and switch to that one.

This jump to a more efficient UTM will happen for every string, so the index sophistication becomes bounded to the size of the first efficient UTM in the enumeration.



sophistication

Make sure that the model is described *efficiently*.

$K(f) = \min\{K(i) : TM_i \text{ computes } f\}$

f(p) = x

K(f): model size

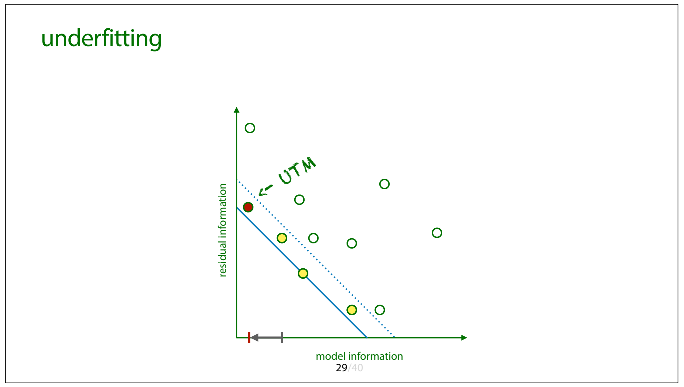
|p|: residual size

K(f) + |p|: total size

residual information
model information

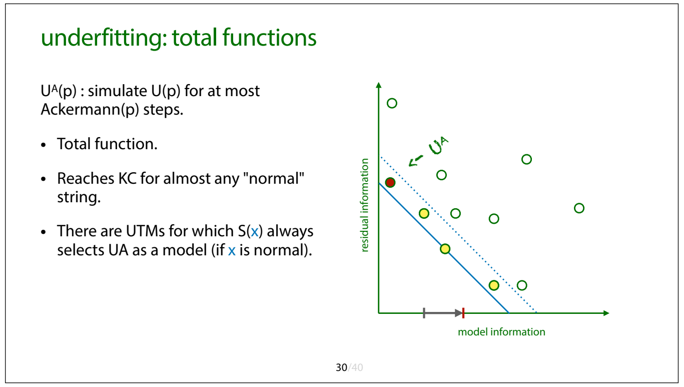
To prevent this problem, most authors requires an efficient description of the model: we can not use more bits than the Kolmogorov complexity of the model.

This solve the catastrophe of index sophistication, but we can still have two problems: undercutting and overfitting.



underfitting

Underfitting happens when the UTM is small enough to appear in the slack region. If that happens, we have a two-part representation for every string where the model part is just the UTM. I.e. the sophistication is never bigger than the size of the UTM.



underfitting: total functions

$U^A(p)$: simulate U(p) for at most Ackermann(p) steps.

- Total function.
- Reaches KC for almost any "normal" string.
- There are UTMs for which S(x) always selects UA as a model (if x is normal).

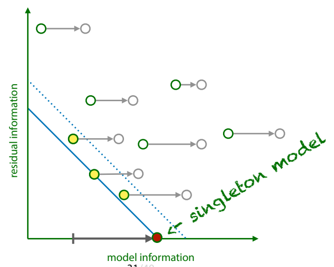
UA
residual information
model information

To prevent this, most authors eliminate the UTM by restricting themselves to total functions (thos that have halting TMs).

This doesn't really fix anything practically. We can very simply set a huge resource bound on the UTM making it a total function.

The result is that there are strings with arbitrarily high sophistication, but only if they take beyond the age of the universe to unpack from their shortest description. Nothing we are likely to encounter in the world will have nontrivial sophistication.

overfitting



overfitting

There exist UTMs for which the singleton models will *always* compress better than any other representation by an arbitrary constant amount.

So either S(x) is always equal to K(x), or it is not invariant.

Who is guilty of what?

Inefficient indices

- Affects some definitions
- Disastrous, S(x) is highly dependent on UTM

Underfitting

- Affects *all* known versions
- S(x) doesn't work as advertised

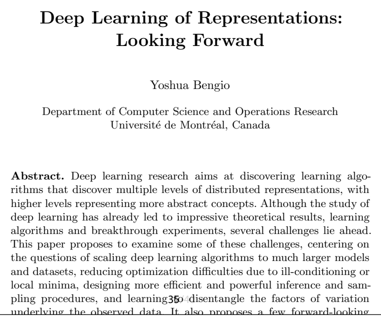
Overfitting

- Affects most versions
- S(x) is not invariant, or equal to K(x)

Outlook

What has changed since 2014, 2015?

2013

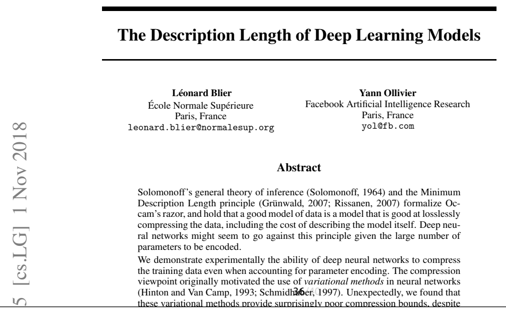

Deep Learning of Representations: Looking Forward

Yoshua Bengio

Department of Computer Science and Operations Research
Université de Montréal, Canada

Abstract. Deep learning research aims at discovering learning algorithms that discover multiple levels of distributed representations, with higher levels representing more abstract concepts. Although the study of deep learning has already led to impressive theoretical results, learning algorithms and breakthrough experiments, several challenges lie ahead. This paper proposes to examine some of these challenges, centering on the questions of scaling deep learning algorithms to much larger models and datasets, reducing optimization difficulties due to ill-conditioning or local minima, designing more efficient and powerful inference and sampling procedures, and learning to disentangle the factors of variation underlying the observed data. It also proposes a few forward-looking

deep neural networks as models /model classes?


The Description Length of Deep Learning Models

Léonard Blier
École Normale Supérieure
Paris, France
leonard.blier@normalesup.org

Yann Ollivier
Facebook Artificial Intelligence Research
Paris, France
yol@fb.com

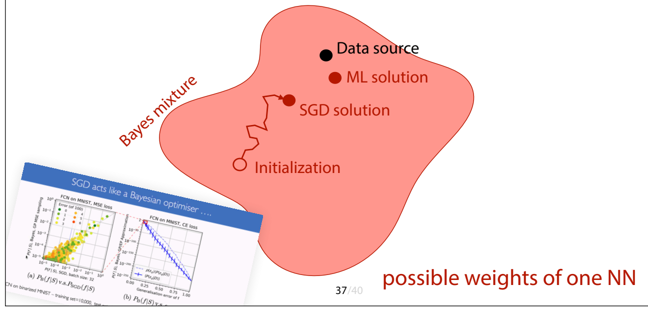
5 [cs.LG] 1 Nov 2018

Abstract

Solomonoff's general theory of inference (Solomonoff, 1964) and the Minimum Description Length principle (Grünwald, 2007; Rissanen, 2007) formalize Occam's razor, and hold that a good model of data is a model that is good at losslessly compressing the data, including the cost of describing the model itself. Deep neural networks might seem to go against this principle given the large number of parameters to be encoded.

We demonstrate experimentally the ability of deep neural networks to compress the training data even when accounting for parameter encoding. The compression viewpoint originally motivated the use of *variational methods* in neural networks (Hinton and Van Camp, 1993; Schmidhuber, 1997). Unexpectedly, we found that these variational methods provide surprisingly poor compression bounds, despite

Is SGD safe against the model class of NNs?

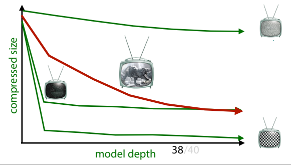

Data source
ML solution
SGD solution
Initialization
Bayes mixture
SGD acts like a Bayesian optimiser ...
possible weights of one NN

NNs as practical complexity hierarchies

Models with arbitrary depth:

- Transformer blocks
- Resnet blocks

Data is *interesting* if a complex model is required to optimally compress it.



No two part coding required

conclusions

Kolmogorov complexity can be **safely approximated**

- We can do better than just plugging in ZIP.

Separating structure from noise objectively is probably impossible.

Deep learning provides some exciting opportunities.