ALGORITHMIC PROBABILITY OF LARGE DATASETS AND THE SIMPLICITY BUBBLE PROBLEM IN MACHINE LEARNING

FELIPE S. ABRAHÃO, HECTOR ZENIL, FABIO PORTO, AND KLAUS WEHMUTH

(Felipe S. Abrahão, Klaus Wehmuth, Fabio Porto) DEXL LAB, NATIONAL LABORATORY FOR SCIENTIFIC COMPUTING (LNCC), 25651-075, Petrópolis, RJ, Brazil.

(Felipe S. Abrahão, Hector Zenil) ALGORITHMIC NATURE GROUP, LABORES FOR THE NATURAL AND DIGITAL SCIENCES, 75005, Paris, France.

(Hector Zenil) OXFORD IMMUNE ALGORITHMS, RG1 3EU, READING, U.K.

The Alan Turing Institute, British Library, 2QR, 96 Euston Rd, London NW1 2DB. ALGORITHMIC DYNAMICS LAB, UNIT OF COMPUTATIONAL MEDICINE, DEPARTMENT OF MEDICINE SOLNA, KAROLINSKA INSTITUTE, SE-171 77, STOCKHOLM, SWEDEN.

E-mail addresses: fsa@lncc.br, hector.zenil@ki.se, fporto@lncc.br, klaus@lncc.br.

Authors acknowledge the partial support from MCTI and CNPq through their individual grants: F. S. Abrahão (302185/2021-6), K. Wehmuth (302211/2021-7), F. Porto (PQ-313961/2018-2).
Abstract. When mining large datasets in order to predict new data, limitations of the principles behind statistical machine learning pose a serious challenge not only to the Big Data deluge, but also to the traditional assumptions that data generating processes are biased toward low algorithmic complexity. Even when one assumes an underlying algorithmic-informational bias toward simplicity in finite dataset generators, we show that fully automated, with or without access to pseudo-random generators, computable learning algorithms, in particular those of statistical nature used in current approaches to machine learning (including deep learning), can always be deceived, naturally or artificially, by sufficiently large datasets. In particular, we demonstrate that, for every finite learning algorithm, there is a sufficiently large dataset size above which the algorithmic probability of an unpredictable deceiver is an upper bound (up to a multiplicative constant that only depends on the learning algorithm) for the algorithmic probability of any other larger dataset. In other words, very large and complex datasets are as likely to deceive learning algorithms into a “simplicity bubble” as any other particular dataset. These deceiving datasets guarantee that any prediction will diverge from the high-algorithmic-complexity globally optimal solution while converging toward the low-algorithmic-complexity locally optimal solution. We discuss the framework and empirical conditions for circumventing this deceptive phenomenon, moving away from statistical machine learning towards a stronger type of machine learning based on, or motivated by, the intrinsic power of algorithmic information theory and computability theory.

Keywords: algorithmic information; computability theory; machine learning; Big Data; bias toward simplicity; data analysis
1. Introduction

The success of statistical machine learning is undeniable, but the expectation of what it can achieve requires an investigation into the fundamental limitations of computable methods on which popular methods are based upon for finding globally optimal parameters that minimise the generalisation (or prediction) errors.

One of the fundamental problems encountered in computational analyses of large datasets is that data mining algorithms are prone to finding statistically significant correlations that are in fact spurious (i.e., correlations that would have occurred in completely randomly generated data anyway) [1], if a proper and meaningful feature selection is not previously embedded into such algorithms by the scientist. A blind or naive application of data mining algorithms to sufficiently large datasets increases the chance of these algorithms finding spurious regularities. This gives rise to a “paradox” that underlies the Big Data paradigm [2]: although computational analysis benefits from larger datasets that are accurate and fine grained, the larger the dataset the more likely it is that spurious correlations will be discovered, with the consequence that the data mining process yields meaningless conclusions. In other words, “too much information tends to behave like very little information” [1].

In the context of machine learning, a central problem is to avoid both overfitting and underfitting. While underfitting (i.e., when the value of the loss/cost/error function obtained from the training set is considered high) is usually easier to solve, e.g., by increasing the model capacity or complexity, avoiding overfitting (i.e., when the loss function value obtained from the test set is considered high in comparison to the training error) demands much more sophisticated methods [3, 4]. Parsimony, Occam’s razor, or the bias toward simplicity can be thought of as underlying principles or heuristics that machine learning methods use in order to tackle the overfitting problem [3, 4]. Indeed, recent methods and results based on the formalised notion of such principles have shown that not only can results similar to the traditional learning methods in differential spaces be obtained (by loss functions, optimisation, and regularisation that directly stem from the bias toward low algorithmic complexity), but also that the same can be obtained for non-differential spaces [5]. Due to the inherent bias toward simplicity in the universal probability distribution of computably generated objects [6], models and parameters that yield a perfect fit with a minimum training error for every data point in the training set tend to have greater algorithmic complexity. Thus, models and parameters with low
algorithmic probability would naturally be penalised in the parameter and hyperparameter optimisation stage of the learning process, and such penalisation would in turn avoid overfitting.

In this article we investigate fundamental limitations and conditions in a context that combines both the “too much information tends to behave like very little information” phenomenon cited in the first paragraph and the bias toward simplicity referenced in the second paragraph. In particular, given an arbitrarily chosen machine learning algorithm, we study worst-case scenarios in which overfitting is expected to always occur once the dataset get sufficiently large, even if the datasets are generated following the universal distribution. This reveals a fundamental limitation in the Big Data paradigm that not only corroborates the results in [6], but also demonstrates the necessity of new algorithmic information-based conditions if we are to avoid finding of “spurious optimal models” when data mining very large datasets.

Given an arbitrarily chosen machine learning algorithm for regression (or prediction) problems and a sufficiently large size of datasets, the main idea of Theorem 3.2 (which follows from Theorem 3.1) is that, although the locally optimal model found by the learning algorithm has an algorithmic complexity arbitrarily smaller than that of the available data, the algorithmic probability of the deceiving dataset still dominates the algorithmic probability of any other particular randomly generated dataset of size greater than or equal to the size of the deceiving dataset. On the one hand, this enables the deceiving dataset to limit the learning algorithm’s capabilities of accurately finding the globally optimal model. On the other hand, the locally optimal model can be found with arbitrary precision.

Formally, we demonstrate in Theorem 3.2 that, for every P, there are sufficiently large datasets $D_{total} = D_a \cup D_{new}$ whose algorithmic probability of the respective $D_a$ being unpredictable deceivers (which satisfy Theorem 3.1 with $|D_a| \geq k$) is higher than (except for a multiplicative constant that depends on P) the algorithmic probability of any other particular dataset of size greater than or equal to the size of the deceiving dataset. P is the arbitrary learning algorithm deceived by the available dataset $D_a$. The available dataset $D_a$ (from which P calculates an optimal model $M_{(P,D_a)}$) is sufficiently large so that there is $k \in \mathbb{N}$ such that $|D_a| \geq k = BB (K (D'_{total}) - O (K (P) + K (\delta)))$, where $D'_{total}$ is any dataset that satisfies Theorem 3.1 $BB (\cdot)$ is a Busy Beaver function, $K (\cdot)$ is the (prefix) algorithmic complexity, and $\delta > 0$ is a rational number such that $\delta$ is smaller than or equal to the maximum generalization/prediction error with respect to $M_{(P,D_a)}$, for any $D_a'$. 
The underlying idea of the proof of Theorems 3.1 and 3.2 is to construct a sufficiently large dataset (which includes the training set, test set, validation set, etc.) available to the learning algorithm so that the prediction error (i.e., the overall generalisation error taking into account fresh data) cannot be minimised beyond a certain level by the learning algorithm already trained by the available data. The key step to achieving this limitation is to ensure that data is generated in such a way that the joint algorithmic complexity of the learning algorithm and the available dataset combined is smaller than the algorithmic complexity of the globally optimal model by a large enough constant.

This particular way in which large enough data can deceive learning algorithms into accurately finding any (locally) optimal solution while still being over-fitted has to do with a phenomenon that appears from a complexity-centered interplay between data and algorithms. As available data becomes sufficiently large, its underlying generating process (or mechanism) likely becomes more complex than the learning algorithm. In this scenario, the data generating process may “trap” the learning algorithm into a “simplicity bubble” in which any solution found is necessarily not complex enough to match the complexity of the actual data generating process. We call this phenomenon the *simplicity bubble effect*.

Section 2 sets the background for our forthcoming results and explains the specific issues in the literature that motivate this article. A brief overview of the related problems in machine learning is presented in Section 2.1. In Section 2.2 we present previous methods and results for machine learning based on algorithmic information theory. In Section 3 we demonstrate our main results. Finally, Section 4 concludes this article by presenting a discussion of additional assumptions to avoid the learning-deceiving phenomenon of the simplicity bubble effect.

2. Background

2.1. Machine learning in large complex datasets. Learning in large complex datasets D, as discussed in Section 1, potentially introduces spurious patterns or may lead to overfitted models. As a matter of fact, although large datasets may exhibit a reasonable quantity of samples of the learning domain, inferences are usually interested in a particular region of such a large domain. In this scenario, finding the weights $W$, for a learner $L$ that fits the data in D involves optimising the error function over a much larger pattern space than a given

---

1A learner is a training algorithm such as a convolutional neural network.
model M may be interested in. As a result, model M may perform sub-optimally when inferring from a sub-region of the domain space.

Another interesting point concerns the training cost associated with large datasets, such as D, and the opportunity to find subspaces \( S \subset D \), considerably smaller than D, and producing the same error guarantees as the complete large dataset \([7]\). Thus, one may want to find a subspace \( S_j \), with \( S = \{S_1, S_2, ..., S_n\} \), and \( D = \cup_{i=1}^n S_i \), such that two machine learning models (\( M_i, M_j \)), using the same learner \( L \), and trained on the complete domain \( S \) and on a subspace \( S_j, j \in \{1..n\} \), respectively, will exhibit prediction errors from inferring from an input \( x \), \( M_i(x) \) and \( M_j(x) \), that differ at most by a threshold \( \epsilon \). A question that naturally arises is how we may compute the subspaces in \( S \) from \( D \).

In \([8]\), subspace computation occurs in a pre-processing stage, before training and inferencing. A clustering process partitions the domain \( D \) into subspaces presenting samples sharing similar values, according to a given distance function. Moreover, for each partition one may compute a representative point, for instance, one whose distance function to all points in the partition is the smallest. Each subspace is a better representation of patterns within its data space. Thus models aiming at predictions in a subspace would fit to patterns in the respective subspace, placing the loss function the patterns of interest. Conversely, once a prediction is to be computed on an input \( x_\xi \), one may look for models whose trained subspace offers a representative point closest to \( x_\xi \). This, in fact, is in line with the No-Free-Lunch theorem \([9]\) for machine learning: for every learner, there exists a task at which it fails, even though that task can be successfully learned by another learner. In other words, no learning algorithm is any better than any other equally likely learning algorithm \([3]\).

To tackle this latter problem, assumptions or heuristics based on variants of the minimum description length, parsimony, Occam’s razor, or the bias toward simplicity are employed in machine learning methods \([3, 4, 10]\). By assuming skewed distributions in empirical data gathering that favor simpler models (e.g., by assigning penalties into the error functions, as done in regularisation), one could avoid the average of equally likely datasets and/or models that constitutes one of conditions for the no free lunch theorem. In this regard, ranging over all computably constructible objects, the algorithmic coding theorem \([6, 11]\) sets a foundational result that connects algorithmic complexity and probability (semi-)measures, formalising the bias toward simplicity in algorithmic information theory (AIT). Indeed, as we will discuss
in the next Section 2.2, recent methods and results have shown the overarching potential of this formal version of bias toward simplicity.

2.2. Machine learning, algorithmic information and model discovery. A suite of algorithms based on the mathematical notions able to characterise the concept of randomness were shown to enable the study of evolving systems based on the first mathematical principles of randomness [12, 13].

In [5], it was shown how those tools can be introduced and exploited in the context of machine learning to help bring together and expand apparently disparate areas of current AI research. For example, it was shown that these model-driven approaches require less training data and can be e.g. more generalisable as they show greater resilience to random attacks. We investigated the shape of a discrete algorithmic space when performing regression or classification using a loss function parametrised by algorithmic complexity, demonstrating that the property of differentiation is not required to achieve results similar to those obtained using differentiable programming such as deep learning.

The chief advantage of such an approach is that these methods within the framework of algorithmic information [14] do not depend on mathematical constructs (such as differentiability), and are better equipped to deal with causal chains and compared to statistical approaches to machine learning. This is because (1) it avoids over-fitting on the available dataset by design (due to an inherent bias toward low-algorithmic-complexity models), and (2) can unveil deeper algorithmic structures not related to obvious redundancies, such as simple regularities that other entropic-like measures would miss and on which most loss functions are based. As shown in [15], entropy is a weak measure of randomness that lacks the robustness of an invariance result as in algorithmic information theory (AIT) [6, 11, 16, 17].

Thus, it follows from these results the question whether or not assuming an underlying universal distribution as in the algorithmic coding theorem is sufficient for guaranteeing that generalisation and prediction would continue to be reliable not only for the available data, but also for fresh data to which the learning process has no access prior to the estimation of the optimal model.

3. A DECEIVING DATASET FOR LEARNING ALGORITHMS

Let $M$ be an arbitrary model defined on the set of parameters $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$ and the set of hyperparameters $\Sigma' = \{\sigma'_1, \sigma'_2, \ldots, \sigma'_{n'}\}$, where $n, n' \in \mathbb{N}$ depend on $M$. Thus, $M$ is an abstract form of the
computably effective procedure that performs the task that the learning algorithm is trying to learn, including not only the parameters to be optimised, but also the hyperparameters (and, possibly any other procedure, data structure, formal theory, etc, that define the model).

For the present purposes, and without loss of generality in most machine learning problems\(^2\), we assume that:

1. the parameter and hyperparameter spaces are either non-denumerable, contained in \(\mathbb{R}^n\) and \(\mathbb{R}^n'\), or infinitely computably enumerable;
2. the number of distinct datasets that a learning algorithm can receive is infinitely computably enumerable;
3. the number of distinct optimal models \(M\) that a learning algorithm can return is infinitely computably enumerable;
4. for every \(M\), there is \(\delta\) and there are infinite computably enumerable models \(M'\) such that the prediction error (or generalisation error) from \(M'\) with respect to \(M\) is larger than \(\delta\), as long as \(\delta\) is less than or equal to the maximum prediction error for any model \(M\).

In this article, we focus on the tasks that are regression (or prediction) problems. While generalising the forthcoming results to classification problems is straightforward, we leave for future research the generalisation of our results to other tasks.

Let \(P\) be an arbitrary learning algorithm for regression problems, which includes the learner and all the phases that end up returning the model, final parameters and/or hyperparameters from the available data. Thus the algorithms for sampling and retrieving the training set, test set and validation set, calculating the loss/cost/error functions, applying gradient descent, cross-validation, regularisation, dropouts, early stopping, bias-variance trade-off, hyperparameter tuning, etc are already embedded into \(P\). In summary, \(P\) constitutes a fully automated learning process without external (human) intervention. That is, the whole learning process is computable.

Note that the learning algorithm can make use of re-sampling, randomised dropouts, noise introduction, or any other call to random trials in order to estimate the optimal model for a given available dataset. If the random events generators called by the learning algorithm are actually pseudo-random instead of truly stochastically random, then the learning process is still computable, but one should make sure to

---

\(^2\)For example, note that those four assumptions clearly hold when both the parameter and hyperparameter spaces belongs to \(\mathbb{R}^n\) and the error function is a mean squared error or a KL divergence.
already include within P the underlying computable process that generates the pseudo-random events. Therefore, in this latter case, the algorithmic complexity of P should already encompass not only the algorithmic information content of the original learning algorithm, but also the algorithmic information content of the computable process that generates the pseudo-random events. In this article, we only deal with learning processes that are computable with or without access to pseudo-random generators. The investigation of the results in the context of learning processes in which there are calls to truly stochastically random events is an interesting open problem and should be the subject of a future research.

Let D denote an arbitrary dataset. Let \( D_a \) be a notation for a dataset that is available to P such that P builds the training, test, and validation sets from \( D_a \). Let \( D_{\text{total}} = D_a \cup D_{\text{new}} \) denote the dataset composed of the available dataset \( D_a \) and any fresh data \( D_{\text{new}} \) to which the learning algorithm has no access before the learning process ends.

A model \( M_{(P,D_a)} \) calculated by the learning algorithm P from the available dataset \( D_a \) is considered to be optimal if it satisfies the arbitrarily chosen optimal trade-off between test error and training error (and/or the bias-variance trade-off criteria) and any other hyperparameter optimisation criteria [3, 4], both cases which would already be embedded into (and calculated by) the learning algorithm P.

For the sake of simplifying our forthcoming proofs, we employ a slight variation of the Busy Beaver function: let \( BB : \mathbb{N} \to \mathbb{N} \) be a Busy Beaver function that calculates the largest integer that a program \( p \) with length \( \leq N \in \mathbb{N} \) can output running on universal Turing machine \( U \) and add 1 to this largest value [18, 19, 20]. That is, \( BB(n) = \max \{ U(p) + 1 \mid n \geq |p| \} \). Now, in order to achieve the following proofs, it is important to remember the usual properties of such a function \( BB(\cdot) \) and the halting probability (or Chaitin’s Omega number) [6, 11, 21].

The next lemma shows that there is an algorithm that can build \( D_a \) satisfying Lemma [3,1] with arbitrary \( n \) such that \( |D_a| \) is greater than any value any program of length \( \leq n \) can calculate and the algorithmic complexity of the optimal model \( M_{(P,D_a)} \) returned by P from \( D_a \) is upper bounded by an arbitrary value \( k \leq n \) where the value of \( k \) is considered to be small by the learning algorithm so that \( M_{(P,D_a)} \) would avoid overfitting [5]. In other words, even if the underlying data generating process of the available dataset is more complex than the learning algorithm itself, the algorithmic information necessary to allow this process to generate a dataset whose model calculated by the
Lemma 3.1. Let \( P \) be an arbitrary learning algorithm. Let \( n \in \mathbb{N} \) be an arbitrary constant such that \( f(n) \leq n \), where \( f : \mathbb{N} \to \mathbb{N} \) is a total computable function already included in \( P \). Then, there is an available dataset \( D_a \) such that \( |D_a| \geq BB(n) \) and \( K(M_{(P,D_a)}) \leq f(n) \), where the model \( M_{(P,D_a)} \) is optimal.

Proof. Let \( h \) be a bit string that represents an algorithm running on a prefix universal Turing machine \( U \) that receives \( P \) and \( \Omega \restriction n \) as inputs, where \( \Omega \) is the halting probability and \( x \restriction n \) denotes the sequence of length \( n \) corresponding to the first \( n \) fractional bits of the real number \( x \). Next, \( h \) computably enumerates the halting programs until it finds the first \( n' \geq n \) such that \( \Omega \restriction n \leq \left( \sum_{U(p)} 2^{-|p|} \right) \restriction n' \). In the next step, \( h \) starts a dovetailing enumeration of all \( D_a \) such that \( |D_a| \geq BB(n') \) and of their respective optimal models \( M_{(P,D_a)} \) calculated by \( P \) so that \( K(M_{(P,D_a)}) \leq f(n) \leq n \). Finally, \( h \) returns the first \( D_a \) that satisfies these criteria. Now, remember that, for every \( x \in \mathbb{N} \), \( \left( \sum_{U(p)} 2^{-|p|} \right) \restriction x \) is sufficient for computing \( BB(x) \) and deciding whether or not a program \( p \) halts, where \( |p| \leq x \). Therefore, program \( h \) is well defined. \( \square \)

The last step before reaching the central Theorems 3.1 and 3.2 is to show in Lemma 3.2 below, that for every computable learning process, there is a sufficiently large dataset \( D_{\text{total}} \) whose underlying data generating process can be as complex as one wishes in comparison to the learning algorithm \( P \). In addition, even though the available dataset \( D_a \subseteq D_{\text{total}} \) might suffice to allow \( P \) to estimate an optimal model \( M_{(P,D_a)} \), fresh data in \( D_{\text{new}} \subseteq D_{\text{total}} \) can render the algorithmic complexity of the actual globally optimal model \( M_{(P,D_{\text{total}})} \) as large as one wishes.

Lemma 3.2. Let \( P \) be an arbitrary learning algorithm. Let \( D_a \) be an arbitrary dataset that is available to the learning algorithm \( P \), allowing it to calculate the optimal model \( M_{(P,D_a)} \). Let \( \delta > 0 \) be a rational number such that \( \delta \) is less than or equal to the maximum generalisation/prediction error with respect to \( M_{(P,D')} \), for any \( D' \). Let \( m \in \mathbb{N} \) be an arbitrary constant. Then, there is a dataset \( D_{\text{total}} = D_a \cup D_{\text{new}} \) such that the prediction error of the actual globally optimal model \( M_{(P,D_{\text{total}})} \)
with respect to $M_{(P,D_a)}$ is greater than or equal to $\delta$ and the inequality

$$m - K(P) - K(D_a) - K(\delta) - O(1) < K(M_{(P,D_{\text{total}})}) \leq K(D_{\text{total}}) + K(P) + O(1) \leq m + 2K(P) + K(\delta) + K(D_a) + O(1)$$

holds, where $D_{\text{new}}$ is the dataset composed of the fresh data that $P$ has no access to in the learning process, and $M_{(P,D_{\text{total}})}$ is the actual globally optimal model that $P$ would calculate if it had access to $D_{\text{total}}$ in the first place.

**Proof.** Let $h'$ be a bit string that represents an algorithm running on a prefix universal Turing machine $U$ that receives $P$, $\delta$, $D_a$, and $\Omega \upharpoonright m$ as inputs. In the next step, $h'$ starts a dovetailing enumeration of all $D_{\text{total}} = D_a \cup D_{\text{new}}$ and their respective optimal models $M_{(P,D_{\text{total}})}$ calculated by $P$ so that the prediction error with respect to $M_{(P,D_a)}$ is larger than or equal to $\delta$. Next, $h'$ computably enumerates the halting programs until it finds the first $m' \geq m$ such that $\Omega \upharpoonright m \leq \left(\sum_{U(p)} 2^{-|p|}\right) \upharpoonright m'$. Then, $h'$ stops when it finds the $BB(m')$-th distinct optimal model $M_{(P,D_{\text{total}})}$ such that the prediction error with respect to $M_{(P,D_a)}$ is greater than or equal to $\delta$, and $h'$ finally returns this $D_{\text{total}}$ as output. Hence, we have it that the inequality $m - K(P) - K(D_a) - K(\delta) - O(1) < K(M_{(P,D_{\text{total}})})$ holds because of our construction of $m$, the properties of the function $BB(\cdot)$, and the construction of the program $h'$. The inequality $K(M_{(P,D_{\text{total}})}) \leq K(D_{\text{total}}) + K(P) + O(1)$ follows from the fact that $U((D_{\text{total}}, P)) = M_{(P,D_{\text{total}})}$, where $\langle \cdot, \cdot \rangle$ denotes an arbitrarily chosen pairing function. Finally, the inequality $K(D_{\text{total}}) + K(P) + O(1) \leq m + 2K(P) + K(\delta) + K(D_a) + O(1)$ also follows from our construction of the program $h'$.

Now, we can introduce the concept of deceiving datasets:

**Definition 3.1.** We say that a dataset $D_a$ is a *deceiver* for a learning algorithm $P$ if the prediction error on fresh data (calculated from $M_{(P,D_a)}$ on $D_{\text{total}}$ in comparison to the actual optimal model $M_{(P,D_{\text{total}})}$ on $D_{\text{total}}$) is greater than $\delta$, where $D_{\text{total}} = D_a \cup D_{\text{new}}$, $D_{\text{new}}$ is the dataset composed of the fresh data that $P$ has no access to in the learning process, and $\delta > 0$ is a rational number such that $\delta$ is less than or equal to the maximum generalisation/prediction error with respect to $M_{(P,D'_a)}$, for any $D'_a$. In addition, we say a dataset $D_a$ is an *unpredictable deceiver* for a learning algorithm $P$ if it is a deceiver and...
the globally optimal model \( M_{(P,D_{total})} \) is uncomputable from \( P, \delta, \) and \( M_{(P,D_a)} \) combined.

The main idea of Theorem 3.1 is that even though the available dataset \( D_a \) is constructed in such a way that it satisfies Lemma 3.1, the fresh dataset \( D_{new} \) is added to \( D_a \) so that the algorithm that computes the underlying data generating process of \( D_{total} = D_a \cup D_{new} \) searches for the extended dataset from \( D_a \) that satisfies Lemma 3.2 and from which the prediction error given by the learning algorithm is maximal. Thus, the underlying data generating process that constructs \( D_{total} \) deceives the learning algorithm \( P \) into “thinking” that an optimal model can be found from \( D_a \), while in fact the actual globally optimal model is unpredictable and its value diverges from the locally optimal model \( M_{(P,D_a)} \) by an arbitrary value \( \delta \).

**Theorem 3.1.** For every learning algorithm \( P \), there is a sufficiently large dataset \( D_{total} = D_a \cup D_{new} \) satisfying Lemma 3.2 such that:

1. \( P \) calculates an optimal model \( M_{(P,D_a)} \) from the available dataset \( D_a \), which satisfies Lemma 3.1.
2. the prediction error on fresh data (calculated from \( M_{(P,D_a)} \) on \( D_{total} \) in comparison with the actual globally optimal model \( M_{(P,D_{total})} \) on \( D_{total} \)) is greater than or equal to \( \delta \), where \( \delta > 0 \) is a rational number such that \( \delta \) is less than or equal to the maximum generalisation/prediction error with respect to \( M_{(P,D_a')} \), for any \( D_a' \);
3. \( |D_a| \geq BB(K(D_{total}) - O(K(P) + K(\delta))) \) holds;
4. and the dataset \( D_a \) is an unpredictable deceiver, so that the globally optimal model \( M_{(P,D_{total})} \) is uncomputable from \( P, \delta, \) and \( M_{(P,D_a)} \) combined.

**Proof.** Let \( q \) be a bit string that represents an algorithm running on a prefix universal Turing machine \( U \) that receives \( P, \delta, \Omega \downarrow_n \), and \( \Omega \downarrow_m^\delta \) as inputs. Then, it runs the program \( h \) from the proof of Lemma 3.1 and calculates \( U((P, \Omega_n, h)) = D_a \). Finally, \( q \) returns \( D_{total} = U((P, \delta, \Omega \downarrow_m, D_a, h')) \), where \( h' \) is the program as in the proof of Lemma 3.2. Now, let \( c' \) be arbitrary. Let \( n \) be sufficiently large in comparison to \( K(P) \) and \( K(\delta) \) such that \( O(K(P) + K(\delta)) \leq f(n) \leq n \), where \( f \) is a total computable function and \( n \in \mathbb{N} \) that satisfy Lemma 3.1. Let \( m \) be sufficiently greater than \( n \) such that there is \( c \) such that \( c \geq m - n > c_\Omega + c' > 2K(P) + K(\delta) + O(1) \) holds, where \( \delta \) satisfies Lemma 3.2 for \( P \) and \( m \) and \( K(\Omega \downarrow_x) \geq x - c_\Omega \) holds for every \( x \). Therefore, from our construction of programs \( q, h \) and \( h' \) and constant \( c \), we have it that \( K(D_{total}) \leq K(P) + K(\delta) + n + c + O(1) \).
From Lemma 3.2, we have it that the prediction error calculated from $M_{(P, Da)}$ on $D_{total}$ in comparison with the actual optimal model $M_{(P, D_{total})}$ on $D_{total}$ is larger than or equal to $\delta$ such that $c_\Omega + c' - 2K(P) - K(\delta) - O(1) < K(M_{(P, D_{total})})$. Since $c'$ was arbitrarily large and $D_a$ satisfies Lemma 3.1, we can choose the minimal and sufficiently large value of $c'$ so that $K(P) + K(\delta) + K(M_{(P, D_a)}) + O(1) < K(M_{(P, D_{total})})$. From our construction of $c$, since $f(n)$ depends linearly on $K(P)$ and $K(\delta)$, we have it that $c$ depends linearly on $K(P)$ and $K(\delta)$. Finally, from basic inequalities in AIT and Lemma 3.1,

$$|D_a| \geq BB(n) \geq$$

$$(2) \quad BB(K(D_{total}) - K(P) - K(\delta) - c - O(1)) \geq$$

$$BB(K(D_{total}) - O(K(P) + K(\delta))).$$

3.1. **The algorithmic probability of very large deceiving datasets.**

So far, we have only shown the existence of unpredictable deceivers. The next question would be how likely they are. More precisely, in the context of randomly generated computably constructible datasets, one could ask after the probability of an unpredictable deceiver as the size of the datasets is already known to be very large with respect to the algorithmic complexity of the learning algorithm. We demonstrate in Theorem 3.2 below that, for sufficiently large datasets, the occurrence of an unpredictable deceiver is more likely (except for a constant that only depends on the learning algorithm) than the occurrence of any other particular dataset of interest.

**Theorem 3.2.** Let $P$ be an arbitrary learning algorithm. Let the available datasets $D_a$’s (from which $P$ calculates optimal models) be sufficiently large so that there is $k \in \mathbb{N}$ such that $|D_a| \geq k = BB(K(D'_{total}) - O(K(P) + K(\delta)))$, where $D'_{total}$ is any dataset that satisfies Theorem 3.1. Then, there are sufficiently large datasets $D_{total} = D_a \cup D_{new}$ whose algorithmic probability (or the universal a priori probability) of the respective $D_a$ being unpredictable deceivers (which satisfy Theorem 3.1 with $|D_a| \geq k$) is higher (except for a multiplicative constant that depends on $P$) than the algorithmic probability (or the universal a priori probability) of any other particular dataset of size $\geq k$.

**Proof.** Let $D'_{total} = D'_{a} \cup D'_{new}$ be a particular deceiving dataset that satisfies Theorem 3.1 for the learning algorithm $P$. From basic inequalities in AIT and the definition of the function $BB(\cdot)$, we know that, for every $D_a$ with $|D_a| \geq k$, if $k = BB(K(D'_{total}) - O(K(P) + K(\delta)))$, then
\[ K(D_{total}) + O(1) \geq K(|D_{total}|) \geq K(|D_a|) \geq K(D_{total}') - O(K(P) + K(\delta)) \]

and

\[ K(D_a) + O(1) \geq K(|D_a|) \geq K(D_{total}') - O(K(P + K(\delta))) \]

where \( D_{total} = D_a \cup D_{new} \) is any arbitrary dataset that extends the available dataset \( D_a \). Let \( \delta \) be any arbitrarily fixed value that only depends on \( P \) and it is a value that satisfies Theorem 3.1 for \( P \). Therefore, from the algorithmic coding theorem, we have it that

\[ 2^{-K(D_{total}') - O(K(P) + K(\delta))} \]

is an upper bound for the algorithmic probability (or the universal a priori probability) of occurrence of any dataset of size greater than or equal to \( k \).

\[ \square \]

4. Discussion

We have investigated the existence and algorithmic probability of sufficiently large datasets that are unpredictable deceivers for a given learning algorithm. Estimating the optimal model for these datasets does not guarantee that this optimal model will continue to be the globally optimal model. For such kind of inverse problems [14], finding the best solution is analogous to finding the optimal model. Therefore, our results assures that computable learning processes are not capable of in general solving inverse problems, even under a bias toward simplicity given by the algorithmic coding theorem—which may render our results to appear counter-intuitive at first glance. Being a limitation for optimisation and learning algorithms as the no free lunch theorems discussed in Section 2.1 but under an algorithmic-informational bias toward low algorithmic complexity (or high algorithmic probability), we have shown in this article that there is no computable learning process that can in general give reliable predictions for any randomly generated computably constructible collection of data.

In fact, the condition on the size of the datasets is employed to guarantee that the algorithmic complexity of any other fresh data will be sufficiently large so that it can render the original dataset a deceiver that induces the learning algorithm to find an optimal model whose algorithmic complexity is much smaller. One can understand those deceivers for inverse problems that (as shown in the previous sections) arise from a wide enough gap between the algorithmic complexity of the underlying generative model of the phenomena and the algorithmic complexity of the problem solver (in our case, the learning algorithm) as:

The simplicity bubble effect. We say that an underlying data generating process (mechanism or external source) is trapping an algorithm
(or a formal theory) into a *simplicity bubble* if the data made available by this data generating process is complex enough to ensure that any optimal model proposed by the algorithm (or formal theory) is not complex enough to approximate the globally optimal model that actually corresponds to any fresh data that the algorithm may receive.

Whilst the simplicity bubble effect is proved to occur in Theorems 3.1 and 3.2, a necessary future research is to investigate the existence of deceivers that does not arise from very large datasets, but from much smaller ones whose underlying generative model indeed has a sufficiently large algorithmic complexity in comparison to that of the learning algorithm. If the answer is positive, this would reveal instances of the simplicity bubble effect that occur not only in very large datasets, but also in much wider range of inverse problems.

In this way, another immediate question would be whether or not there are conditions that would avoid deceivers. When solving inverse problems from empirical data, the conditions (including the bias toward simplicity) are both empirical and mathematical. In this direction, we now introduce a new assumption on the underlying generative models of the datasets.

### 4.1. A new hypothesis in order to avoid the simplicity bubble.

Besides assuming a bias toward simplicity, i.e., probability distributions of datasets that follows the universal distribution [6] of randomly generated computably constructible objects, one would need another assumption that avoids deceivers like those demonstrated in Section 3.

To this end, note that the first key step in Lemma 3.1 that allows Theorem 3.1 to be proved is the fact that $n$ can be arbitrarily large, which leads $m$ in Lemma 3.2 to also be larger. These two values push the algorithmic complexity of the deceiving dataset up. Thus, one immediate way to avoid this is to assume a condition that ensures the algorithmic complexity of the datasets cannot be much greater than that of the learning algorithm. Furthermore, the algorithmic complexity of the optimal model calculated from the available dataset $D_a$ must be sufficiently smaller than the algorithmic complexity of $D_a$ so as to avoid overfitting [5]—which also was guaranteed in our proofs by the function $f$ in Lemma 3.1.

With this purpose of guaranteeing that the chosen computable method can indeed find the global optimal solution for a given available dataset from which it can analyze and learn, the main idea of the following proposed principle is to limit the phenomena on which the computable method can be considered to give trustworthy solutions. Informally, the complexity of the phenomena about which the learning algorithms
can make predictions cannot be greater (up to a small constant) than the complexity of the learning algorithms themselves, while keeping the complexity of the predicting optimal models (which are produced by the learning algorithms from the available datasets) smaller than the complexity of the available datasets (i.e., the phenomena from which the learning algorithms return the predicting optimal models). In other words, given that the data generating processes are biased toward low complexity, we “cage”: the complexity of the data on which predictions can be made around the complexity of the learning algorithm; and the complexity of the optimal models given by the learning algorithm (from the available data) below the complexity of the available data. Predictions would be reliable in a computably generated universe only if the complexity of the phenomena is trapped into a “complexity cage” that depends on the learning algorithm.

**Complexity caging principle/hypothesis.** Let P be an arbitrary learning algorithm. We say P gives a reliable solution to inverse problems on a (randomly generated or deterministically) computably constructible dataset \( D_{\text{total}} = D_a \cup D_{\text{new}} \), where \( D_a \) is an available dataset from which P outputs the predicting optimal models \( M_{(P,D_a)} \) and \( D_{\text{new}} \) is the dataset composed of the fresh data that P has no access to in the learning process, if there is a constant \( c \) that depends only on P such that:

1. \( K(D_{\text{total}}) \leq K(P) + c \);
2. \( K(M_{(P,D_a)}) \leq f(K(D_a)) \), where \( f : \mathbb{N} \to \mathbb{N} \) is a total computable function already included in P and the model \( M_{(P,D_a)} \) is considered to be optimal by P;
3. the probability of occurrence of \( D_{\text{total}} \) is given by the universal distribution such that there is \( C \in \mathbb{R} \), program \( p \) and a discrete probability measure \( P : \{ D_{\text{total}} | \left( U(\langle n, p \rangle) = D_{\text{total}}, \ n \in \mathbb{N} \} \to [0,1] \subset \mathbb{R} \) such that, for every \( D_{\text{total}}, \ P \left[ \text{“dataset } D_{\text{total}} \text{ occur”} \right] = C \cdot \frac{1}{2^{K(D_{\text{total}})}} \).

While the above “complexity caging” strategy clearly avoids the proofs presented in this article, future research is necessary for demonstrating whether or not it is capable of avoiding every possible deceiving dataset.

For this reason, we raise the complexity caging principle in this article as a hypothesis. In the context of randomly generated (or deterministically) computably constructible data, it posits an open problem, searching for mathematical conditions that, if satisfied, guarantee that

---

3Note that, although \( f(\cdot) \) is computable, the function \( K(\cdot) \) is uncomputable.
the prediction on empirical data is reliable; and allowing one to move away from current limitations of statistical machine learning (including deep learning) towards stronger versions of machine learning based on the computability theory and algorithmic information theory.

With the purpose of achieving reliable predictions from computable learning processes, our results for example show that a pre-processing (as discussed in Section 2.1) of the data to be analyzed should take place so as to ensure that the algorithmic complexity of the collection of data does not surpass that of the learning algorithm itself.

Therefore, we show that new pre-processing methods based on approximations to the value of $K(x)$, such as those that might be based on the complexity caging principle, are a necessary complement to the universally overfitting-free machine learning approach discussed in Section 2.2 so that generalisation becomes not only reliable for the available data, but also for new phenomena.

**Funding**

Authors acknowledge the partial support from MCTI and CNPq through their individual grants: F. S. Abrahão (302185/2021-6), K. Wehmuth (302211/2021-7), F. Porto (PQ-313961/2018-2).

**References**

[1] C. S. Calude, G. Longo, *The Deluge of Spurious Correlations in Big Data*, Foundations of Science 22 (3) (2017) 595–612. doi:10.1007/s10699-016-9489-4 URL [http://link.springer.com/10.1007/s10699-016-9489-4](http://link.springer.com/10.1007/s10699-016-9489-4).

[2] G. Smith, *The paradox of big data*, SN Applied Sciences 2 (6) (2020) 1041. doi:10.1007/s42452-020-2862-5 URL [http://link.springer.com/10.1007/s42452-020-2862-5](http://link.springer.com/10.1007/s42452-020-2862-5).

[3] I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning*, MIT Press, 2016, [http://www.deeplearningbook.org](http://www.deeplearningbook.org).

[4] I. H. Witten, E. Frank, M. A. Hall, C. J. Pal (Eds.), *Data Mining (Fourth Edition)*, fourth edition Edition, Morgan Kaufmann, 2017. doi:10.1016/C2015-0-02071-8 URL [https://www.sciencedirect.com/science/article/pii/B9780128042915000052](https://www.sciencedirect.com/science/article/pii/B9780128042915000052).

[5] S. Hernández-Orozco, H. Zenil, J. Riedel, A. Uccello, N. A. Kiani, J. Tegnér, *Algorithmic Probability-Guided Machine Learning on Non-Differentiable Spaces*, Frontiers in Artificial Intelligence 3 (2021) 567356. doi:10.3389/frai.2020.567356 URL [https://www.frontiersin.org/articles/10.3389/frai.2020.567356/full](https://www.frontiersin.org/articles/10.3389/frai.2020.567356/full).

[6] C. S. Calude, *Information and Randomness: An algorithmic perspective*, 2nd Edition, Springer-Verlag, 2002.
[7] B. Mirzasoleiman, J. A. Bilmes, J. Leskovec, Coresets for data-efficient training of machine learning models, in: Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event, Vol. 119 of Proceedings of Machine Learning Research, PMLR, 2020, pp. 6950–6960.
URL http://proceedings.mlr.press/v119/mirzasoleiman20a.html

[8] R. Pereira, Y. Souto, A. Chaves, R. Zorilla, B. Tsan, F. Rusu, E. Ogasawara, A. Ziviani, F. Porto, Djensemble: A cost-based selection and allocation of a disjoint ensemble of spatio-temporal models, Association for Computing Machinery, New York, NY, USA, 2021, p. 226–231.
URL https://doi.org/10.1145/3468791.3468806

[9] D. Wolpert, W. Macready, No free lunch theorems for optimization, IEEE Transactions on Evolutionary Computation 1 (1) (1997) 67–82.
doi:10.1109/4235.585893

[10] B. Scholkopf, F. Locatello, S. Bauer, N. R. Ke, N. Kalchbrenner, A. Goyal, Y. Bengio, Toward Causal Representation Learning Proceedings of the IEEE 109 (5) (2021) 612–634.
doi:10.1109/JPROC.2021.3058954
URL https://ieeexplore.ieee.org/document/9363924/

[11] G. Chaitin, Algorithmic Information Theory, 3rd Edition, Cambridge University Press, 2004.

[12] H. Zenil, S. Hernández-Orozco, N. Kiani, F. Soler-Toscano, A. Rueda-Toicen, J. Tegnér, A Decomposition Method for Global Evaluation of Shannon Entropy and Local Estimations of Algorithmic Complexity Entropy 20 (8) (2018) 605.
doi:10.3390/e20080605
URL http://www.mdpi.com/1099-4300/20/8/605

[13] H. Zenil, N. A. Kiani, A. A. Zea, J. Tegnér, Causal deconvolution by algorithmic generative models, Nature Machine Intelligence 1 (1) (2019) 58–66.
doi:10.1038/s42256-018-0005-0
URL http://www.nature.com/articles/s42256-018-0005-0

[14] H. Zenil, N. Kiani, F. Abrahão, J. Tegnér, Algorithmic Information Dynamics, Scholarpedia Journal 15 (7) (2020) 53143.
doi:10.4249/scholarpedia.53143

[15] H. Zenil, N. A. Kiani, J. Tegnér, Low-algorithmic-complexity entropy-deceiving graphs, Physical Review E 96 (1) (2017) 012308.
doi:10.1103/PhysRevE.96.012308
URL http://dx.doi.org/10.1103/PhysRevE.96.012308

[16] R. G. Downey, D. R. Hirschfeldt, Algorithmic Randomness and Complexity Theory and Applications of Computability, Springer New York, New York, NY, 2010. arXiv:http://doi.org/10.1007/978-0-387-68441-3
doi:10.1007/978-0-387-68441-3
URL http://link.springer.com/10.1007/978-0-387-68441-3

[17] M. Li, P. Vitányi, An Introduction to Kolmogorov Complexity and Its Applications, 4th Edition, Texts in Computer Science, Springer, Cham, 2019.
doi:10.1007/978-3-030-11298-1

[18] G. Chaitin, Life as Evolving Software, in: H. Zenil (Ed.), A Computable Universe, World Scientific Publishing, Singapore, 2012, pp. 277–302.
doi:10.1142/9789814374309_0015
URL http://www.worldscientific.com/doi/abs/10.1142/9789814374309_0015
[19] F. S. Abrahão, The “paradox” of computability and a recursive relative version of the Busy Beaver function, in: C. Calude, M. Burgin (Eds.), Information and Complexity, 1st Edition, World Scientific Publishing, Singapore, 2016, Ch. 1, pp. 3–15. doi:10.1142/9789813109032_0001

[20] F. S. Abrahão, K. Wehmuth, A. Ziviani, Algorithmic networks: Central time to trigger expected emergent open-endedness, Theoretical Computer Science 785 (2019) 83–116. doi:10.1016/j.tcs.2019.03.008

[21] G. Chaitin, A Theory of Program Size Formally Identical to Information Theory, Journal of the ACM 22 (3) (1975) 329–340. doi:10.1145/321892.321894