

The Uses and Limitations of Occam Algorithms: a response to Herrmann

Zaman Keinath-Esmail¹

Ard A. Louis¹

¹*Rudolf Peierls Centre for Theoretical Physics, University of Oxford, Beecroft
Building Parks Road, Oxford, OX1 3PU, UK*

March 13, 2025

Abstract

In a recent paper, Daniel Herrman uses probably approximately correct (PAC) learning theory to argue that Occam algorithms do not justify a preference for simpler hypotheses. He claims to derive equally efficient "Anti-Occam" algorithms favouring the most complex hypotheses. We argue that Herrmann's analysis omits key elements of Occam algorithms, which eliminate the possibility of "Anti-Occam" algorithms and counter many of his arguments. These elements clarify the intrinsic connection of Occam algorithms to theories of learnability. Occam algorithms are not a failed epistemic justification of Occam's razor but rather a pragmatic base for practical algorithms

1 Introduction

Occam's razor, often attributed to William of Ockham, states that, all else being equal, it is best not to multiply entities beyond necessity. In essence, simpler explanations, or hypotheses, should be preferred. While versions of the razor have been widely applied in scientific and philosophical arguments over the centuries, there is no consensus on whether there are metaphysical, epistemic, or pragmatic justifications for a bias favouring simplicity (Sober, 2015; Chen, 2023).

The advent of statistical and computational learning theory (Pearl, 1978; Valiant, 1984; Harman and Kulkarni, 2007; Sober, 2015) along with computational complexity theory (Aaronson, 2013) – provides new routes to address some of these questions, albeit in restricted settings. By translating abstract notions such as simplicity and learning to well-defined mathematical formulations, these fields offer opportunities to disambiguate these concepts and the relations between them, and to potentially resolve ambiguities that have persisted in philosophical debates. Statistical and computational learning theory, in particular, is often employed to establish constraints on the range of hypotheses that must be considered and to define bounds on the type and quantity of data necessary for effectively learning a concept. Insights derived from these approaches, frequently grounded in worst-case scenarios and information-theoretic principles, can help elucidate the role, if any, of simplicity in learning within such controlled environments.

Moreover, statistical and computational learning theory is not restricted solely to silicon chips. It may also be used to describe the necessary conditions for human learning (Pearl, 1978; Valiant, 1984; Harman and Kulkarni, 2007). While these theories are unlikely to settle the metaphysical question of whether the world itself is simple, they may shed light on the

epistemic and pragmatic utility of Occam's razor.

In this context, it is interesting to explore the connection between probably approximately correct (PAC) learning and Occam algorithms. Briefly, PAC learning is a framework for inductive learning developed by Valiant (1984) that combines notions of computational feasibility with the ability to learn non-trivial classes of logical rules. It is considered one of the most important theories of learning in computer science and is thought to hold for learners more generally (Valiant, 1984). Occam algorithms (a term coined by Blumer et al. (1987)) are an important development in this sub-field because they demonstrate how prioritising simple hypotheses can be shown to be equivalent to PAC learning in polynomial time. Key passages in the original literature (Littlestone and Warmuth, 1986; Blumer et al., 1987, 1989; Board and Pitt, 1990; Li, Tromp, and Vitanyi, 2003), as well as the way Occam algorithms are described in influential textbooks (Kearns and Vazirani, 1994), could easily lead one to assume that the authors also consider these mathematical results relevant for epistemological justifications of Occam's razor, albeit within the constrained environment of PAC learning, with noiseless samples, and independent and identically distributed (IID) data.

In an interesting recent paper, Daniel Herrmann sets out to provide a critical philosophical analysis of Occam algorithms and their purported relevance for Occam's razor (Herrmann, 2020). He argues that the current literature misinterprets the implications of Occam algorithms for Occam's Razor. In particular, he argues that Occam algorithms require an unstated assumption "that each hypothesis space in the family exhibits a certain similarity property" for the algorithm to function. He then defines an "Anti-Occam" algorithm which considers only the C most complex hypothesis in a class and, he says, functions just as well as an Occam algorithm (Herrmann, 2020). Such an algorithm would suggest that there is nothing special about simplicity, and that Occam algorithms do not provide any epistemic justification of

Occam's razor. In fact, Herrmann's argument doesn't hinge on being able to make an anti-Occam algorithm with the most complex hypotheses; it relies only on finding non-simplicity-related restrictions of the hypothesis space, since any learning algorithm guarantees polynomial PAC learning on an arbitrary bounded set of hypotheses (Kearns and Vazirani, 1994).

Here, we will contend that Herrmann's definition and analysis of Occam algorithms neglects essential technical aspects of how they function, and misunderstands the rationale behind their creation. This has radical implications for his analysis. To develop our arguments, we begin by defining certain key terms and algorithms, just as Herrmann (2020) did. We next summarize his work and point out some key problems with his analysis of Occam algorithms. We explain why "Anti-Occam" algorithms as defined by Herrmann (2020) are not well-posed. We also criticise the suggestion that a similarity assumption is necessary for Occam algorithms to be good learners. While we ultimately agree with Herrmann that Occam algorithms do not offer an epistemic justification for Occam's razor, our reasoning diverges. In fact, such a conclusion is already present in the foundational work by Pearl (1978) upon which the derivation of Occam algorithms was based. One can find similar arguments in numerous influential commentaries in the literature on computational and statistical learning theory, see e.g.(Helmbold and Warmuth, 1992; Schaffer, 1993; Domingos, 1999; Montanez, 2017; Shalizi, 2021). We conclude by summarising what these analyses of Occam algorithms can tell us about algorithm design and learning theory, as well as their potential philosophical implications

Finally, we also comment on the practice of incorporating "decorative prose" into technical papers. Such writing has a rhetorical purpose, mainly to provide a broader context to technical findings, which means it is held to radically different epistemic norms than the core content. We argue that it should be approached with particular caution in interdisciplinary settings.

2 Key notation

Detailed mathematical definitions of PAC learning and Occam algorithms can be found in Appendix A. Here we define some key notation used in the main text, following conventions from (Blumer et al., 1987; Kearns and Vazirani, 1994), to

A domain $X = \cup_{n \geq 1} X_n$ is defined where each X_n contains instances—strings of length n . $\mathcal{H} = \cup_{n \geq 1} \mathcal{H}_n$ is a set of functions, or “concepts” that map from a domain X to a range X' , where each \mathcal{H}_n takes as input only instances of length n . For example, Boolean functions will map to the range $X' = \{0, 1\}$. We define a fixed representation scheme which maps from a string $\sigma \in \Gamma^*$ to $h \in \mathcal{H}$, where Γ is a (potentially finite) alphabet. Without loss of generality, one can assume binary encoding, so that $\Gamma = \{0, 1\}$. Then for a given concept $h \in \mathcal{H}$, the length of its representation σ is $\ell(h)$, and $\ell(h)$ represents the “complexity” of the concept. The target concept is defined as a $f \in \mathcal{H}$ that the algorithm is trying to learn. An observation is a point $(x_i, f(x_i))$, $x_i \in X$. A set of observations, or sample, is denoted by M , which contains m observations taken independently according to some probability distribution \mathcal{P} on X . Note that most of statistical and computational learning theory assumes that this data is IID. The effect of relaxing this assumption is often hard to assess. That means that any potential implications for Occam’s razor may be similarly constrained.

In Blumer et al. (1987), the set \mathcal{H} is a countably infinite hypothesis class (we will use ‘hypothesis’ and ‘concept’ interchangeably here). This means that there are an infinite number of hypotheses in the class; it follows that there is no upper limit to the length (i.e. complexity) of the hypotheses. Each \mathcal{H}_n may have only a finite number of hypotheses (at most 2^{2^n} in the Boolean case if X is binary), but there may be a countably infinite number of representations Γ^* for these hypotheses. When we say we are considering the simplest or most complex

hypotheses, we are in fact referring to the simplicity or complexity of their representations.

3 Herrmann's argument

Herrmann (2020) rightly identifies two key components of any learning algorithm. The first is that we must ensure our effective hypothesis space contains at least one hypothesis which is consistent with the sample. Usually this is done by assuming the target concept is included in the hypothesis space: this is called the *realizability assumption*. The second is that our effective hypothesis space must contain a finite number of hypotheses: in Occam algorithms on a binary alphabet, this is done by restricting ourselves to the $C = 2^{\ell(f)^c m^\alpha}$ simplest hypotheses. The following subsections present a summary of his resulting analysis.

3.1 Similarity

Herrmann argues that we cannot know for sure that the target concept is included in the Occam algorithm's effective hypothesis space, as the entire point of learning is that the target function is unknown. If it is not included in the hypothesis space, the realizability assumption is not met. Then, for an Occam algorithm to output a consistent hypothesis, we need another way to ensure that such a hypothesis even exists in our effective hypothesis space. If the realizability assumption does not hold, Herrmann claims, then there must be a tacit "similarity assumption" which states that there are hypotheses which are similar enough to the target hypothesis that they would be consistent with the sample.

This similarity property would have to be a property of the entire original hypothesis class. Such a requirement would impose a tight restriction on the types of hypothesis classes for which Occam algorithms exist, limiting the scope of its potential implications for the validity

of Occam’s razor.

3.2 “Anti-Occam” Algorithms

Herrmann observes that, if the similarity assumption holds, then the only thing that an Occam algorithm does is restrict our hypothesis space to include only C hypotheses. There is no particular reason to choose the C simplest hypotheses: Blumer et al. (1987)’s proof that Occam algorithms are polynomial learners relies only on the fact that the algorithms choose C hypotheses; their complexity is irrelevant. The similarity assumption would mean that any set of C randomly chosen hypotheses could be used as a hypothesis class. In fact, why not choose the C most complex hypotheses in \mathcal{H}_n ?

Thus, Herrmann defines an “Anti-Occam” algorithm which restricts the hypothesis class to include only the C most complex hypotheses in a given \mathcal{H}_n . His similarity assumption means he need not worry about whether the target concept is in this effective hypothesis space. Similarity guarantees that there is at least one hypothesis which is consistent with the sample. Herrmann then shows, that such an “Anti-Occam” algorithm would require the same sample size as an Occam algorithm. The two algorithms would then perform equally well given the same data sample; but one favours simple hypotheses and the other favours complexity. Thus, Herrmann claims, there is nothing special about Occam algorithms or simplicity: the existence of Occam algorithms does not provide any justification for Occam’s razor.

3.3 Encodings and Implications

Herrmann (2020)’s penultimate section details additional concerns he has with the use of Occam algorithms as epistemic justification for Occam’s razor. Firstly, he points out that the

complexity, i.e. description length, of any given hypothesis depends on the language or encoding used to label it. Thus, the measure of complexity that is used in the Occam algorithm is representation dependent; a different choice of language could result in an entirely different set of hypotheses in the Occam algorithm’s restricted hypothesis space. How can Occam algorithms justify Occam’s razor when they have no objective definition of simplicity?

Secondly, Herrmann (2020) criticises the direction of implication. The fact that Occam algorithms are PAC learners is not a particularly strong justification of anything philosophically interesting: it would be a much more noteworthy result if PAC learnability of a hypothesis class was somehow shown to imply the existence of an Occam algorithm for the class.

4 A Response to Herrmann

4.1 No Such Thing as an “Anti-Occam” Algorithm

Herrmann (2020) rightly points out that Blumer et al. (1987)’s polynomial bounds arise from the fact that, if our encoding system is binary, the algorithm only considers the $C = 2^{\ell(f)^c} m^\alpha$ simplest hypotheses in \mathcal{H} . He then defines his “Anti-Occam” algorithm as one that instead considers the C “most complex” hypotheses in each \mathcal{H}_n . In Appendix E, we show that such “Anti-Occam” algorithms are not well-posed (e.g. because there may exist infinitely many representations of a hypothesis class and so no ‘most complex’ hypothesis) and prove that they cannot polynomially learn as many hypothesis classes as Occam algorithms.

However, the weaknesses of “Anti-Occam” algorithms are not fatal to Herrmann’s general argument: Many earlier works have also critiqued the relevance of Occam algorithms for Occam’s razor (Domingos, 1999; Helmbold and Warmuth, 1992; Montanez, 2017; Schaffer,

1993; Shalizi, 2021) by showing that the individual hypotheses could be simple or complex, what matters is that the cardinality of the set is bounded and grows polynomially (see Appendix B).

4.2 Properties of the Hypothesis Class

Herrmann (2020)’s second attack on Occam algorithms is to claim that they require an implicit similarity assumption for hypotheses in \mathcal{H} . This argument fails when one considers how bounds of the Occam algorithm work. Firstly, it acts on a sample of a function f that has maximum complexity $\ell(f)$. Secondly, it outputs a function that has maximum complexity $\ell(f)^c m^\alpha$, with $c \geq 1$ and $0 \leq \alpha < 1$. If $\ell(f)^c m^\alpha \geq \ell(f)$, then our new “effective hypothesis space” will include f , and the realizability assumption is satisfied. The worst case (most restricted hypothesis space) scenario, given the bounds on c and α , is that our effective hypothesis space has complexity $\ell(f)$. Thus, even in the worst case, our effective hypothesis space is always guaranteed to include f and therefore satisfy the realizability assumption: no need for similarity. In fact, as we show in Appendix B, it is non-Occam algorithms which require either a similarity assumption or additional knowledge about the hypothesis class in order to learn.

4.3 Representation Dependence

The fact that which hypotheses are considered by the Occam algorithm depends on choice of encoding seems to suggest that the definition of simplicity is arbitrary: different encoding languages will disagree on which concepts are simple or complex. This does not mean that we cannot gain insight from this property. In fact, it appears that learnability itself may be a

property of the choice of representation. We describe an illustrative example of the importance of choice of representation for learning Boolean functions in Appendix C. This illustrates how the choice of representation is important for learnability, and the descriptive power or richness of a representation can determine its efficacy in learning applications (Board and Pitt, 1990). The representation dependence of simplicity in Occam algorithms reflects the relation between learnability and choice of representation: this is not a bug, but rather the only possible outcome.

5 Simplicity and Learning Theory

5.1 No Free Lunch and Learning with Minimal Information

If we satisfy realizability and end up with a finite, polynomially growing hypothesis class, then we can restrict our hypothesis space in any way we like. We could create a non-Occam algorithm by considering C randomly distributed hypotheses in the hypothesis class, *as long as we can ensure the realizability assumption is met*. But, as discussed in Appendix B, the problem remains: how can we ensure that this assumption is met, if we don't know anything about f ? After all, we don't know the function we are trying to learn before we learn it.

The 'no free lunch' theorem of machine learning states that, averaged uniformly over all possible f , all learning algorithms will perform equally poorly and therefore no better than random guessing (Schaffer, 1993; Wolpert, 1992). Averaged uniformly over all possible f , an Occam's razor-like bias towards simplicity will do no better than a random learner either. Any restriction of the hypothesis class (or even choice of hypothesis class) amounts to a form of inductive bias (Sterkenburg and Grunwald, 2021). To learn, we must therefore have some knowledge of the problem which allows us to choose a suitable inductive bias, restricting the

hypothesis space. Such a restriction of the hypothesis class will necessarily be one which meets the realizability assumption, which we have previously stated is required for learning.

In general cases where one does not know which inductive bias to use to restrict the hypothesis class, Occam algorithms provide a useful way of restricting the hypothesis space that requires minimal amounts of a priori information. The natural zero lower bound on complexity (complexity cannot be negative), combined with the structure of the effective hypothesis space (which is necessarily stratified by complexity), means that one can trivially satisfy realizability by ratcheting up the complexity until the target hypothesis is included in the effective hypothesis space. Appendix D provides a more technical analysis of this process for Occam algorithms using the continuous Vapnik-Chervonenkis (VC) dimension as a complexity measure.

In the context of Occam algorithms, the preference for simplicity is simply a pragmatic parameterisation of the hypothesis space that provides a trivial lower bound and makes it easy to satisfy the realizability assumption. In principle, any other discrete parameter with a trivial lower bound could be used; description length is just a convenient one for functions with binary encoding. Because the number of functions grows exponentially with complexity (Li and Vitányi, 2008), this way of partitioning is an especially efficient way of organizing the hypothesis space so that it remains relatively small and still satisfies realizability. Naturally, if there is additional information about f , then the hypothesis space can be restricted further to make learning easier. But in the absence of other such information, Occam algorithms offer a minimum amount of restriction required for learning.

5.2 The Role of Simplicity in Learning

Blumer et al. (1987) showed that, if \mathcal{H} is learnable by an Occam algorithm, it is polynomially learnable in general. In 1989, the same research team published another paper, this time defining Occam algorithms using a VC dimension (see Appendix D). They again showed that learnability by such an Occam algorithm implied polynomial learnability (Blumer et al., 1989). They left open the question of whether the converse was true: whether any \mathcal{H} that is polynomially learnable can be learnt by an Occam algorithm.

Board and Pitt (1990) showed a partial converse to Blumer et al. (1987). For concepts that are strongly closed under exception lists (a property which is beyond the scope of this paper, but includes many Boolean formulae), they showed that polynomial learnability is, in fact, equivalent to learnability by Occam algorithms. In the VC-dimension model of Occam algorithms, their results are even more general, requiring only closure under exception lists, as opposed to strong closure (Board and Pitt, 1990; Keinath-Esmail, 2023). Their theorem suggests that there is an intrinsic link between learnability of many concept classes and the ability to weakly approximate a solution to the minimum consistent hypothesis problem (see Appendix A.3). This is exactly the result that Herrmann claims is missing but would strengthen arguments in favour of the epistemic value of Occam’s Razor.

However, we do not believe that the results of Board and Pitt (1990); Keinath-Esmail (2023) provide any epistemic justification of Occam’s Razor. They do not show any inherent benefit of using an Occam algorithm over another PAC learner for a given PAC-learnable hypothesis class; only that it is possible to construct an Occam algorithm for such a class. Instead, these results only justify the pragmatic value of Occam algorithms: they exist for, and thus can be used to learn, the PAC-learnable hypothesis classes analysed in Board and Pitt

(1990). In learning situations where little is known about the hypothesis class, it may therefore be wise to attempt to learn it using an Occam algorithm. An Occam algorithm may not be the most *efficient* way of learning these concept classes, but it is *sufficient*.

This apparent connection between the learnability of a concept class and the ability to learn it using an Occam algorithm raises deep questions about the role of simplicity in learning. When Pearl (1978) examined the justification behind preferring simple hypotheses in learning, he wrote that “[n]o logical argument can possibly connect simplicity with credibility”, arguing that simplicity was incidental to his analysis and that the only requirement for learning is to suitably restrict the hypothesis class. The ‘no free lunch’ theorem formalizes this notion: if preferring simplicity is an inductive bias, then it cannot perform equally well across all learning applications.

Extensions and idealisations of Occam algorithms to use representation-independent measures of complexity such as Kolmogorov complexity Li and Vitányi (2008) have been developed to attempt to further clarify the connections between simplicity and learning (Li, Tromp, and Vitanyi, 2003; Li and Vitányi, 2008). However, it remains frustratingly unclear which general properties make a hypothesis class learnable, how to choose a suitable representation for such a class, and what role simplicity plays.

Interestingly, many practical learning scenarios involve data with significant underlying structure, implying some form of (Kolmogorov) simplicity since that structure can be used to compress the data. Consider the deep neural networks (DNNs) that underlie modern modern AI. These exhibit an inherent inductive bias toward (Kolmogorov) simple functions. Interestingly, this bias is not a strict application of Occam’s razor that selects the absolute simplest function but rather a softer, exponential bias favouring simpler functions which counterbalances the exponential growth in the number of functions with increasing

complexity Mingard et al. (2024). For the generic example of learning Boolean functions, this bias means that DNNs are only capable of effectively learning simple Boolean functions which represent an exponentially small subset of all possible Boolean functions. By contrast, DNNs perform poorly when trying to learn the vast majority of Boolean functions, which are more complex Mingard et al. (2024). On average over all Boolean functions, they do no better than random chance, as expected by no-free-lunch theorems. Such results suggest that the remarkable success of modern AI systems is predicated on their inductive bias toward learning (Kolmogorov) simple hypotheses. The key question for AI, therefore, is not primarily an epistemic one—since a bias toward simplicity is necessary to achieve strong performance on structured datasets—but an ontological one: why is the data that we care about so simple?

It should also be noted that for the problem of learning Boolean functions, the minimal-consistent hypothesis problem is NP hard, and Occam algorithms show that softening this constraint can lead to important improvements in learnability. It would be interesting to explore whether something analogous holds in some applications of Occam’s razor: that pushing for the absolute simplest explanation may not be efficacious.

6 The dangers of decorative prose

Learning theorists who invoke Occam’s razor tend to fall into two camps. The first consists of scientists who regard their technical work as having genuine philosophical significance. For example, a substantial body of literature uses principles from algorithmic information theory—particularly Solomonoff induction—to engage with longstanding philosophical debates on induction more generally, and Occam’s razor more specifically Li, Tromp, and Vitányi (2003); Li and Vitányi (2008); Rathmanner and Hutter (2011). This connection is not

surprising given that Solomonoff induction was directly inspired by Carnap's programme of inductive logic. However, recent philosophical scrutiny has cast doubt on the viability of these approaches, on grounds that are both subtle and nontrivial. For example, Sterkenburg (2016) has argued that these accounts presuppose a hidden inductive assumption of effectiveness, undermining claims of justifying Occam's razor. Neth (2023) has pointed out the incompatibility between the need to generate practical predictions, which requires reliance on computable approximations of Solomonoff induction, and the problem of language dependence, which demands an appeal to its uncomputable idealisation.

A second, and much larger camp, may still invoke Occam's razor as useful shorthand for the broad brushstrokes of their focus: technical links between complexity and learnability, simplicity-based algorithms, and learning strategies. But, they do not regard their technical work as having a significant bearing on key philosophical questions.

Sometimes, distinguishing which camp an author is in can be difficult. A learning theorist may simply be badly informed about philosophy. However, a more subtle source of ambiguity arises from the broader institutional pressures shaping scientific communication. The ever-increasing pressure to justify the economic and practical relevance of academic work—its “impact”—stimulates the inclusion of speculative passages about potential applications in scientific publications and grant proposals. We will refer to such passages as “decorative prose”.

Decorative prose is performative rather than substantive; its primary function is rhetorical, to signal openness to applications, to frame research within broader societal narratives, or—especially in grant writing—to inveigle non-specialist evaluators of the work's prospective utility. Much like the decorative wrapping of a gift, such prose is meant to enhance the presentation but is not integral to the substance of the work.

Both readers of scientific literature and specialist reviewers of grant proposals are generally attuned to this performative dynamic. There exists a tacit understanding that decorative prose is not to be scrutinized with the same epistemic rigour as the technical material it accompanies. This implicit contract allows for a certain latitude in making speculative claims while maintaining the integrity of the core scientific discourse.

In this case, Blumer et al. (1987) included passages that, at face value, suggest they think their work is directly relevant to an epistemic version of Occam’s razor. However, they are self-consciously building on Pearl (1978)’s classic work which emphasises that simplicity itself has no relation to credibility and that this version of Occam’s razor offers nothing but a restriction on the hypothesis class. Indeed, one of the authors in a later paper describes Occam algorithms as considering a “small” hypothesis class, without mentioning simplicity at all (Helmbold and Warmuth, 1992). All this suggests that the passages in Blumer et al. (1987) about Occam’s razor are best understood as decorative prose, intended for rhetorical effect rather than substance.

By contrast, in the context of philosophy, claims about Occam’s razor would be held to much more exacting standards. The absence of explicit markers signalling the nature of decorative prose poses a vexing challenge for interdisciplinary work. While specialists can effortlessly distinguish such rhetoric from substantive content, outsiders may struggle to make the same distinction.

7 Conclusions

Occam algorithms are primarily a technical advance within the broader field of PAC learning, offering an approximation to the minimum consistent hypothesis problem for Boolean

functions Board and Pitt (1990). Despite their evocative name and the decorative prose used in the original paper, it is unlikely they were also intended to carry philosophical implications for Occam's razor. Nevertheless, there may still be philosophical mileage in Occam algorithms. For example, the fact that they use softer constraints on simplicity and exploit the ease of stratifying hypotheses by simplicity may provide insight into the epistemology of learning. Such questions merit further investigation by philosophers.

A PAC Learning and Occam Algorithms

A.1 PAC Learning

Probably approximately correct (PAC) learning is an attempt to define the minimum properties of what can be considered a learning algorithm. The PAC model, as well as various restrictions and relaxations thereof, are now standard fare in the field of computational learning theory and form the foundation of our understanding of machine learning. This formalism is used to describe both algorithms (is this algorithm a PAC learner for this concept class?) and concept classes (is this concept class learnable by some PAC learner?).

A hypothesis class is PAC learnable if, for every target hypothesis f of complexity at most $\ell(f)$, there is some algorithm that takes sample size $m_{\mathcal{H}}(\epsilon, \delta, n, \ell(f))$ such that the algorithm will, with probability $1 - \delta$, produce a hypothesis h that is approximately equal to f (within some error tolerance ϵ). Then an algorithm is a PAC learner for a given hypothesis class if, given an input sample $m_{\mathcal{H}}(\epsilon, \delta, n, \ell(f))$, the algorithm will, with probability $1 - \delta$, produce a hypothesis h that is approximately equal to f (within the error tolerance ϵ).

Following Blumer et al. (1987); Kearns and Vazirani (1994), we write this out more

formally as follows :

Definition A.1 (PAC Learning). A hypothesis class \mathcal{H} is PAC learnable if there exists a function $m_{\mathcal{H}}(\varepsilon, \delta, n, \ell(f)) : (0, 1)^2 \times \mathbb{N}^2 \rightarrow \mathbb{N}$, a learning algorithm with the following property: for all $0 < \varepsilon, \delta < 1$, for all $n \in \mathbb{N}$, for all probability distributions \mathcal{P} on X , and for all $f \in \mathcal{H}$ with complexity at most $\ell(f)$, then when running the learning algorithm on $m \geq m_{\mathcal{H}}(\varepsilon, \delta, n, \ell(f))$ examples generated by \mathcal{P} on X and labelled by f , the algorithm returns hypothesis $h \in \mathcal{H}$ such that with probability of at least $1 - \delta$, $\mathcal{P}(h \Delta f) \leq \varepsilon$, where Δ is the set (or symmetric) difference operation.

The PAC definition has been extended to allow the initial concept class from which the target f is drawn to be distinct from the hypothesis class from which the hypothesis h is chosen. We will discuss the implications of this extension in later sections.

PAC learnability makes no assumptions about computational feasibility: it is a purely statistical definition (Kearns and Vazirani, 1994). If we want to guarantee that our algorithm is computationally feasible, we must restrict it to run in “polynomial time”: in this case, that means requiring that it runs in time “polynomial in the length of the sample”, and that $m_{\mathcal{H}}$ is polynomial in $1/\varepsilon$, $1/\delta$, n , and $\ell(f)$ (Blumer et al., 1987, 1989; Kearns and Vazirani, 1994). This property is called “polynomial PAC learning/learnability”, sometimes shortened simply to “polynomial learning/learnability”.

A.2 Occam Algorithms

Following Blumer et al. (1987) an Occam algorithm is defined as follows:

Definition A.2 (Occam Algorithm). An Occam algorithm for \mathcal{H} with constant parameters $c \geq 1$ and $0 \leq \alpha < 1$ is a learning algorithm that, when given a sample of size $m_{\mathcal{H}}$ of any

function $f \in \mathcal{H}$ with complexity at most $\ell(f)$:

1. produces a consistent hypothesis of complexity at most $(\ell(f))^c m^\alpha$
2. runs in time polynomial in the sample length

It can be shown that, if there is an Occam algorithm for a concept class, it will run using a sample size that is polynomial in $1/\epsilon$, $1/\delta$, and $\ell(f)$. Thus, if there is an Occam algorithm for a given concept class \mathcal{H} , that concept class is polynomially learnable (Blumer et al., 1987, 1989).

Kearns and Vazirani (1994) use a similar definition for an Occam algorithm but set the bound $c \geq 0$, and require the output hypothesis to have maximum complexity $(n\ell(f))^c m^\alpha$. In addition, they allow the initial concept class from which f is drawn to be different from the hypothesis class from which h is chosen, which is more general than Blumer et al. (1987)'s algorithm.

Herrmann (2020) uses a maximum complexity bound that is dependent on n and m but not $\ell(f)$: this is only equivalent to the above definitions if our representation scheme is chosen in such a way that $\ell(f)$ is polynomial in n . In addition, he assumes f and h are drawn from the same \mathcal{H}_n . These choices restrict the possible representations and hypothesis classes the algorithm can consider but are not fatal to his argument.

A.3 Motivation for Occam Algorithms

It is important to be aware of the reason Occam algorithms were developed in the first place. Blumer et al. (1987) were motivated in their creation of Occam algorithms by some of the limitations of learning algorithms at the time. It is widely thought that, for many concept

classes, a good learning strategy is to find the “minimum consistent hypothesis”: the simplest hypothesis that is consistent with the sample (Blumer et al., 1987, 1989). However, for certain concept classes, such as deterministic finite automata (DFAs) or some Boolean functions, finding a minimum consistent hypothesis is NP-hard (Blumer et al., 1987; Kearns and Valiant, 1994; Pitt and Warmuth, 1993): while it may be relatively easy to confirm (in polynomial time) that one has a valid hypothesis, it is computationally impossible to verify (in polynomial time) that one has found the minimally complex hypothesis.

Thus, the idea behind Occam algorithms was primarily to simply use the same general methodology – trying to limit the algorithm to simple hypotheses – while relaxing the “minimum” constraint to prevent NP-hardness. This relaxation is why we have a maximum complexity that is polynomial in $\ell(f)$ and sublinear in m . A minimum consistent hypothesis algorithm would have maximum complexity $\ell(f)$ (note that this is an Occam algorithm with $c = 1$ and $\alpha = 0$; the worst case scenario). Occam algorithms are essentially a weak approximation of the solution to the minimum consistent hypothesis problem Board and Pitt (1990).

B Non-Occam Algorithms

Consider a learning algorithm (in a representation scheme for \mathcal{H}_n which is countably infinite), which considers a hypothesis space containing C hypotheses which are not necessarily the C simplest hypotheses: a non-Occam algorithm (the set of non-Occam algorithms includes the set of “Anti-Occam” algorithms). Is this algorithm as good (efficient) a PAC learner for \mathcal{H} as an Occam algorithm?

Maybe. Certainly, the restriction on the hypothesis space means the combinatoric

arguments used to prove that Occam algorithms are PAC learners will equally apply to non-Occam algorithms, as long as they sufficiently restrict the effective hypothesis space. But the other requirement for PAC learning, which is not always explicitly mentioned, is that the restricted hypothesis space being considered must contain either the target function itself or a suitable approximation: this assumption is the aforementioned *realizability* assumption.

Non-Occam algorithms will be PAC learners on \mathcal{H} if and only if they meet the realizability assumption. But meeting the realizability assumption is a crucial issue for such algorithms. It is not guaranteed. One must have some a priori knowledge of the hypothesis space or the target function to choose a restricted hypothesis space that includes either the target function or a suitable approximation.

An illustrative example: consider a sample of m data points and an infinite space of binary classification hypotheses encoded numerically in, say, lexicographic order. Assume we know, somehow, that the true concept's encoding is the n th Ramanujan prime R_n multiplied by the first nonzero digit of $\zeta(n+2)$ where ζ is the Riemann zeta function, for some unknown n . Then we could create an algorithm that considers each hypothesis encoded by R_n times the first nonzero digit of $\zeta(n+2)$ for all $R_n \leq m$. Then for a function with error ϵ , the chance of being consistent with all the data is $(1 - \epsilon)^m$. There will be at most $m/\log(m)$ Ramanujan primes less than m , so with a union bound, we find our algorithm's chance of success $1 - \delta \leq (m(1 - \epsilon)^m)/\log(m)$. Thus, we can achieve arbitrarily small errors with polynomial samples, so we have a PAC algorithm.

But without the a priori knowledge regarding the relevance Ramanujan primes and the Riemann zeta function for the target function, where would our learning algorithm be? Considering all 2^m hypotheses that could describe the sample would be an exponentially difficult task: we must somehow restrict the hypothesis space to allow for PAC learning. A

priori knowledge is crucial, providing an inductive bias which allows us to restrict our hypothesis space sufficiently to be polynomially learnable.

Absent such knowledge, a non-Occam algorithm would require some sort of “similarity assumption” stating that the hypothesis class contains functions that are similar enough to the target function that any restricted hypothesis space would meet the realizability assumption. The issue remains that such a similarity assumption relies on knowledge of the target function and introduces an inductive bias that restricts which hypothesis classes can be learnt.

C Boolean Functions and Representation Dependence

Boolean functions offer a clear example of how the choice of representation affects learning. Briefly, a Boolean function of a set A is a subset of A which can be obtained by a finite number of set intersection, union, and complement operations. Such functions may be represented in a variety of ways, such as truth tables (which explicitly write out the value of the function for each possible value of each argument), decision trees, and decision lists. Two forms of particular interest are disjunctive normal forms (DNFs), which are written as a union of intersections of arguments and their complements, and conjunctive normal forms (CNFs), which are written as an intersection of unions of arguments and their complements. We will restrict each of these forms to contain only k terms: thus, we call them k -DNFs and k -CNFs (Kearns and Vazirani, 1994).

It has been shown that, given a sample of length m produced by a particular k -DNF, the task of finding a k -DNF that is consistent with this sample (in other words, learning a k -DNF using a hypothesis space of k -DNFs) is equivalent to the graph three-colouring problem, which is known to be NP-complete (Kearns and Vazirani, 1994). However, it is possible to construct an

algorithm which finds a k -CNF consistent with the aforementioned sample. Thus, k -DNFs are not learnable by themselves, but they are learnable by k -CNFs (Kearns and Vazirani, 1994). Of note, both k -DNFs and k -CNFs are simply representations of Boolean functions. k -CNFs are a superset of k -DNFs: that is, every k -DNF can be written as a k -CNF, but the reverse is not true. This case of learning Boolean functions provides an example of a strong representation dependence.

D Occam Algorithms and the VC Dimension

In their 1989 paper, Blumer et al. extend the idea of Occam algorithms to hypothesis classes in which the complexity is determined by the Vapnik-Chervonenkis (VC) dimension of the class, as opposed to simply the length of a hypothesis. The VC dimension is independent of representation. Board and Pitt (1990) showed that any polynomially learnable algorithm is learnable by an Occam algorithm in the VC dimension model.

We present a useful theorem from Blumer et al. (1989):

Theorem D.1. *Let H be any well-behaved hypothesis space of finite VC dimension d contained in 2^X , P be any probability distribution on X and the target concept c be any Borel set contained in X . Then for any $0 < \epsilon, \delta < 1$, given*

$$m \geq \max \left(\frac{4}{\epsilon} \log \frac{2}{\delta}, \frac{8d}{\epsilon} \log \frac{13}{\epsilon} \right)$$

independent random examples of c drawn according to P , with probability at least $1 - \delta$, every hypothesis in H that is consistent with all of these examples has error at most ϵ .

This theorem, which can be found as Theorem A2.2 in Blumer et al. (1989), is key to the

development of VC dimension-based Occam algorithms. As soon as we show that there is an upper bound on complexity of the hypothesis our algorithm returns, Theorem 1 guarantees that we will have the errors we desire, and a sample size m that is polynomial in our errors and complexity measure d .

It is known that any hypothesis class with a finite VC dimension is PAC-learnable; however, Occam algorithms as defined in Blumer et al. (1989) can be used to polynomially learn hypothesis classes with an infinite VC dimension. It's true that, considering all possible hypotheses in a class with infinite dimension, the class would not be PAC learnable. The way Occam algorithms allow us to learn these classes is by restricting the hypothesis space, and allowing this new *effective* hypothesis space to grow only polynomially in some complexity measure s .

It is worth noting that Blumer et al. rely on this 'natural' measure of complexity s and its properties for much of their work. For an infinite hypothesis class C , they require that the VC dimension of C_s (the set of concepts in C with size less than s) grow polynomially in s .

Secondly, Occam algorithms rely on the ability to guarantee that the hypothesis h output by the algorithm will have a size s polynomial in that of the target concept. This step is arguably what makes Occam algorithms go: if we have this guarantee, then we can slot in Theorem 1 and be on our way. When we describe simplicity as a useful parameterisation of the hypothesis space, this is what we are referring to. The parameter s and our ability to place certain limits on the maximum s of the hypothesis space is what makes proof of Occam algorithms go.

Choice of suitable s that meets these requirements can be a limiting factor in creating Occam algorithms. For many hypothesis classes whose learnability is unknown or which are suspected not to be polynomially learnable, there is no obvious 'natural' complexity measure we can use that is polynomially related to the VC dimension.

E “Anti-Occam” Algorithms

We first explain why “Anti-Occam” algorithms are not well posed, showing that they cannot polynomially learn the same hypothesis classes that Occam algorithms can. Motivated by discussions with Daniel Herrmann, we then include some potential modifications to anti-Occam algorithms, showing that in all cases the anti-Occam model is applicable to fewer learning problems than Occam algorithms. Thus, the two models cannot be considered equivalent, or even equally pragmatically useful.

E.1 Against “Anti-Occam” Algorithms

The existence of such an “Anti-Occam” algorithm is predicated on the idea that each \mathcal{H}_n is finite, and thus there is an upper limit to the complexity of the hypothesis. Then we can choose the C most complex hypotheses. Remember, however, that complexity is in fact a property of our *representation* of \mathcal{H}_n , rather than the underlying hypotheses. Our representations may be countably infinite, even if our hypothesis class is not, and it is these representations that the algorithm is considering and learning.

Herrmann (2020)’s “Anti-Occam” algorithm, then, can only exist if we are guaranteed a finite number of representations in our hypothesis class. After all, how can an infinite hypothesis class (and thus infinitely complex hypothesis class) have a “most complex” hypothesis? It can’t. Thus, the existence of Herrmann’s “Anti-Occam” algorithm is a feature of the subset of representation schemes he considers, rather than a strategy for learning any hypothesis class under any fixed representation. The feasibility of ensuring the hypothesis contains a finite number of representations is examined further in Appendix E.2. Appendix E.4 explores the feasibility of creating an “Anti-Occam” algorithm that includes a built-in

realizability assumption: we find that such an algorithm is not as broadly applicable as Occam algorithms.

E.2 Can We Make “Anti-Occam” Hypothesis Classes Finite?

Above, we make the point that “Anti-Occam” algorithms cannot exist because there may be an infinite number of representations of our hypothesis class \mathcal{H}_n , so there cannot be a ‘most complex’ representation (and hence hypothesis). We argue that Herrmann’s “Anti-Occam” algorithms can therefore only exist for certain representation schemes, in which the number of representations is finite: for infinite representations, we instead analyze the feasibility of general ‘non-Occam’ learning algorithms.

We know that there are a finite number of hypotheses in \mathcal{H}_n , and it is the number of representations that is potentially infinite. Thus, we may ask: is there a way to restrict this infinite number of representations such that each distinct hypothesis has one and only one representation? If such a restriction method were possible, it would be possible to create an “Anti-Occam” algorithm that matches Herrmann (2020)’s definition, learning the most complex hypothesis in each \mathcal{H}_n , even in the case of infinite representation schemes (as opposed to making do with our ‘non-Occam’ algorithms).

It might seem like we could stipulate that for each hypothesis, we represent it using the shortest representation that corresponds to the hypothesis, and disregard all longer representations of the same hypothesis: let us call this the ‘minimum representation length’ stipulation. This would be equivalent to trying to solve the ‘minimum consistent hypothesis’ problem, which we know is NP-hard for some hypothesis classes, such as DNFs (Blumer et al., 1987). Thus, the ‘minimum representation length’ stipulation will not be computationally

feasible for all hypothesis classes.

There may be other ways to restrict each hypothesis to having only one representation, but for such a method to be feasible, it would have to be shown that it is in P for all hypothesis classes that can be learnt by Occam algorithms. Failing such a method, there is no guarantee that a hypothesis class can be restricted to having a finite number of representations in it. Without such a restriction, it is not possible to have an “Anti-Occam” algorithm which is able to learn the same hypothesis classes as Occam algorithms.

E.3 Structure of the Hypothesis Space

The way Blumer et al. (1987) set up their domain and hypothesis class, there is no stratification based on n . X includes strings of any/all lengths, and their hypothesis class \mathcal{H} is the class of $\{0, 1\}$ -valued functions representing regular languages over X . That is, a Blumer hypothesis could conceivably take as input strings of varying lengths, whereas under Herrmann (2020)’s definition, such a function would require a union of multiple hypotheses, each from a different \mathcal{H}_n . Blumer et al. (1987) also state an assumption that any Boolean function of v variables will return a standard value representing ‘undefined’ on any input string which does not consist of exactly v bits.

Now, let us grant the ‘minimum representation length’ stipulation from Appendix E.2. It is possible to choose the most complex ones, and write that as an anti-Occam algorithm. But in Blumer et al.’s hypothesis space, such a construction is not possible: the lack of stratification by n means that one cannot separate the hypothesis space into an infinite number of finite partitions, as Herrmann (2020) do. Thus, choosing the ‘most complex’ hypothesis is not possible (the most complex hypothesis is infinitely long and the computer will never finish

reading it).

The hypothesis spaces as defined by Blumer et al. (1987) and Herrmann (2020) are equivalent if we allow unions of hypotheses from different \mathcal{H}_n under Herrmann’s definition (this is why we decided to use the same definition of \mathcal{H} and \mathcal{H}_n as Herrmann (2020) did). However, what appears to be an anti-Occam algorithm under Herrmann’s definition does not restrict the hypothesis space to the ‘most complex’ hypotheses as defined by Blumer et al. (1987), but simply to a finite set of hypotheses about which there is little else of note.

In other words, the possibility of the existence of “Anti-Occam” algorithms is dependent on this particular construction of the hypothesis space; in other constructions, the algorithm still works as a learning algorithm but is not ‘anti-Occam’ per se.

E.4 Can We Make “Anti-Occam” Algorithms Satisfy Realizability?

Let us take an algorithm, on this stratified hypothesis space (made up of a union of various \mathcal{H}_n as we describe in Appendix E.3) and granting the ‘minimum length representation’ stipulation from Appendix E.2. Let us assume that somehow, we know that the target hypothesis has length at least $\ell(h)$. We can choose as our effective hypothesis space as all hypotheses in \mathcal{H}_n whose representation has length greater than $\ell(h)$: this is not quite the same as an Occam algorithm (which may output hypotheses slightly longer than the maximum complexity of the target) but let us proceed anyhow. With this construction of an anti-Occam algorithm, we satisfy the realizability assumption and have the smallest-possible “Anti-Occam” hypothesis class.

There are, in the Boolean case, 2^{2^n} hypotheses in \mathcal{H}_n . Our effective hypothesis class thus has $2^{2^n} - 2^{\ell(h)}$ hypotheses in it, so $r = 2^{2^n} - 2^{\ell(h)}$. As described in Blumer et al. (1987), we need sample size m such that $r(1 - \epsilon)^m = \delta$. Rearranging, $m \geq \frac{-\ln(r) + \ln(1/\delta)}{\ln(1 - \epsilon)}$.

For polynomial learning, we require m to be polynomial in ϵ, δ, n , and $\ell(h)$. However, our $\ln(r)$ term, $\ln(2^{2n} - 2^{\ell(h)})$, is superpolynomial in n . Thus, we cannot polynomially learn this hypothesis class.

“Anti-Occam” algorithms constructed to satisfy the realizability assumption in this way will only be able to learn hypothesis classes in which the cardinality of \mathcal{H}_n grows exponentially with n . This is a much more restricted set of hypothesis classes than those which Occam algorithms are able to learn. Alternatively, “Anti-Occam” algorithms would require the knowledge of further ways to restrict their effective hypothesis space (i.e. more information about the learning problem) in order to be able to learn such classes polynomially.

References

- Aaronson, Scott. 2013. “Why philosophers should care about computational complexity.” *arXiv preprint*, arXiv:1108.1791v3. <https://doi.org/10.48550/arXiv.1108.1791>.
- Blumer, Anselm, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. 1987. “Occam’s Razor.” *Information processing letters* 24(6): 377–80. [https://doi.org/10.1016/0020-0190\(87\)90114-1](https://doi.org/10.1016/0020-0190(87)90114-1).
- Blumer, Anselm, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. 1989. “Learnability and the Vapnik-Chervonenkis dimension.” *Journal of the ACM (JACM)* 36(4): 929–965. <https://doi.org/10.1145/76359.76371>.
- Board, Raymond, and Leonard Pitt. 1990. “On the necessity of Occam algorithms.” In *Proceedings of the Twenty-second Annual ACM Symposium on Theory of Computing*, ed.

- Harriet Ortiz, 54–63. New York:Association for Computing Machinery.
[https://doi.org/10.1016/0304-3975\(92\)90367-0](https://doi.org/10.1016/0304-3975(92)90367-0).
- Chen, Eddy K. 2023. “The simplicity of physical laws.” *arXiv:2210.08143*.
<https://doi.org/10.1111/nous.12542>.
- Domingos, Pedro. 1999. “The role of Occam’s razor in knowledge discovery.” *Data mining and knowledge discovery* 3:409–25. <https://doi.org/10.1023/A:1009868929893>.
- Hancock, Thomas, Tao Jiang, Ming Li, and John Tromp. 2005. “Lower bounds on learning decision lists and trees.” In *STACS 95: 12th Annual Symposium on Theoretical Aspects of Computer Science Munich, Germany, March 2–4, 1995 Proceedings*, ed. Ernst Mayr, and Claude Puech, 527–38. Berlin:Springer.
https://doi.org/10.1007/3-540-59042-0_102.
- Harman, Gilbert, and Sanjeev Kulkarni. 2007. *Reliable Reasoning: Induction and Statistical Learning Theory*. MIT Press. <https://doi.org/10.7551/mitpress/5876.001.0001>.
- Haussler, David, Michael Kearns, Nick Littlestone, and Manfred K. Warmuth. 1991. “Equivalence of models for polynomial learnability.” *Information and Computation* 95(2): 129–61. [https://doi.org/10.1016/0890-5401\(91\)90042-Z](https://doi.org/10.1016/0890-5401(91)90042-Z).
- Helmbold, David P., and Manfred K. Warmuth. 1992. “Some weak learning results.” In *Proceedings of the fifth annual workshop on Computational learning theory, July, 1992*, ed. David Haussler, 399–412. New York:Association for Computing Machinery.
<https://doi.org/10.1145/130385.130428>.

- Herrmann, Daniel. 2020. “PAC Learning and Occam’s Razor: Probably Approximately Incorrect.” *Philosophy of Science* 87(4): 685–703. <https://doi.org/10.1086/709786>.
- Rathmanner, Samuel, and Marcus Hutter (2011). “A philosophical treatise of universal induction.” *Entropy* 13(6): 1076–1136. <https://doi.org/10.3390/e13061076>.
- Kearns, Michael, and Leslie Valiant. 1994. “Cryptographic limitations on learning boolean formulae and finite automata.” *Journal of the ACM (JACM)* 41(1): 67–95. <https://doi.org/10.1145/174644.174647>.
- Kearns, Michael, and Umesh Vazirani. 1994. *An introduction to computational learning theory*. MIT press. <https://doi.org/10.7551/mitpress/3897.001.0001>.
- Keinath-Esmail, Zaman. 2023. “On the equivalence of Occam algorithms.” *arXiv preprint*, arXiv:2308.15906. <https://doi.org/10.48550/arXiv.2308.05906>.
- Li, Ming, and Paul Vitányi. 2008. *An introduction to Kolmogorov complexity and its applications*. Springer. <https://doi.org/10.1007/978-0-387-49820-1>.
- Li, Ming, John Tromp, and Paul Vitányi. 2003. “Sharpening Occam’s razor.” *Information Processing Letters* 85(5): 267–74. [https://doi.org/10.1016/S0020-0190\(02\)00427-1](https://doi.org/10.1016/S0020-0190(02)00427-1).
- Littlestone, Nick, and Manfred K. Warmuth. 1986. “Relating data compression and learnability.” *Unpublished manuscript*. <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=e5ec4059989296a8d1e2d8cad880352617462d3b>.
- Chris Mingard, Henry Rees, Guillermo Valle-Pérez, and Ard A. Louis. 2025. “Deep neural

- networks have an inbuilt Occam's razor." *Nature Communications* 16, 220.
<https://doi.org/10.1038/s41467-024-54813-x>.
- Montanez, George. 2017. "Why machine learning works." PhD diss., Carnegie Mellon University. <http://reports-archive.adm.cs.cmu.edu/anon/anon/ml2017/CMU-ML-17-100.pdf>.
- Pearl, Judea. (1978). "On the connection between the complexity and credibility of inferred models." *International Journal of General System* 4(4): 255–64.
<https://doi.org/10.1080/03081077808960690>.
- Neth, Sven (2023). "A dilemma for Solomonoff prediction." *Philosophy of Science* 90(2): 288–306. <https://doi.org/10.1017/psa.2023.21>.
- Pitt, Leonard, and Leslie Valiant (1988). "Computational limits on learning from examples." *Journal of the ACM (JACM)* 35(4): 965–84. <https://doi.org/10.1145/48014.63140>.
- Pitt, Leonard, and Manfred K. Warmuth. 1993. "The minimum consistent DFA problem cannot be approximated within any polynomial." *Journal of the ACM (JACM)* 40(1): 95–142.
<https://doi.org/10.1145/138027.138042>.
- Schaffer, Cullen. 1993. "Overfitting avoidance as bias." *Machine learning* 10:153–78.
<https://doi.org/10.1023/A:1022653209073>.
- Shalizi, Cosma. "Occam-style bounds for long programs." Last modified 23 September, 2021.
<http://bactra.org/notebooks/occam-bounds-for-long-programs.html>.
- Sober, Elliott. 2015. *Ockham's Razors: A User's Manual*. Cambridge: Cambridge University Press. <https://doi.org/10.1017/CB09781107705937>.

Sterkenburg, Tom F. (2016). “Solomonoff prediction and Occam’s razor.” *Philosophy of Science* 83(4): 459–479. <https://doi.org/10.1086/687860>.

Sterkenburg, Tom, and Peter Grunwald. 2021. “The no-free-lunch theorems of supervised learning.” *Synthese* 199:9979–10015. <https://doi.org/10.1007/s11229-021-03233-1>.

Valiant, Leslie. 1984. “A theory of the learnable.” *Communications of the ACM* 27(11): 1134–42. <https://doi.org/10.1145/1968.1972>.

Wolpert, David. 1992. “On the connection between in-sample testing and generalization error.” *Complex Systems* 6(1): 47.

F Acknowledgements

We thank Daniel A. Herrmann for feedback on our manuscript.

G Declarations

None to declare.

H Funding Information

None to declare.