JMLR: Workshop and Conference Proceedings 1–25, 2018 Algorithmic Learning Theory 2018

On the Help of Bounded Shot Verifiers, Comparators and Standardisers for Learnability in Inductive Inference

Ziyuan Gao MATGAOZ@NUS.EDU.SG
Department of Mathematics
National University of Singapore
Singapore 119076, Republic of Singapore
Sanjay Jain SANJAY@COMP.NUS.EDU.SG
School of Computing
National University of Singapore
Singapore 117417, Republic of Singapore
Frank Stephan FSTEPHAN@COMP.NUS.EDU.SG
Department of Mathematics
National University of Singapore
Singapore 119076, Republic of Singapore
Thomas Zeugmann THOMAS@IST.HOKUDAI.AC.JP
Division of Computer Science
Hokkaido University
N-14, W-9, Sapporo 060-0814, Japan

Editor: Mehryar Mohri and Karthik Sridharan

Abstract

The present paper deals with the inductive inference of recursively enumerable languages from positive data (also called text). It introduces the learning models of *verifiability* and *comparability*. The input to a verifier is an index e and a text of the target language L, and the learner has to *verify* whether or not the index e input is correct for the target language L. A comparator receives two indices of languages from the target class $\mathcal{L}$ as input and has to decide in the limit whether or not these indices generate the same language. Furthermore, *standardisability* is studied, where a *standardiser* receives an index j of some target language L from the class $\mathcal{L}$, and for every $L \in \mathcal{L}$ there must be an index e such that e generates L and the standardiser has to map every index j for L to e. Additionally, the common learning models of *explanatory learning*, *conservative explanatory learning*, and *behaviourally correct learning* are considered. For almost all learning models mentioned above it is also appropriate to consider the number of times a learner changes its mind. In particular, if no mind change occurs then we obtain the *finite* variant of the models considered. Occasionally, also learning with the help of an oracle is taken into consideration.

The main goal of this paper is to figure out to what extent verifiability, comparability, and standardisability are helpful for the inductive inference of classes of recursively enumerable languages. Here we also distinguish between *indexed families*, *one-one enumerable classes*, and *recursively enumerable classes*. Our results are manyfold, and an almost complete picture is obtained. In particular, for indexed families and recursively enumerable classes finite comparability, finite standardisability, and finite verifiability always imply finite learnability. If at least one mind change is allowed, then there are differences, i.e., for indexed families, comparability or verifiability imply conservative explanatory learning, but standardisability does not; still explanatory learning can be achieved.

1. Introduction

The process of hypothesising a general rule from "eventually" complete positive data is called inductive inference. Philosophy of science has studied inductive inference during the last centuries. Some of the principles developed are very much alive in *algorithmic learning theory*. Computer scientists widely use their insight into the theory of computability to obtain a better and deeper understanding of processes performing inductive generalisations.

The present paper mainly deals with formal language learning, a field in which many interesting and sometimes surprising results have been elaborated within the last decades (see Jain et al., 1999; Zeugmann and Zilles, 2008, and the references therein). Inductive inference of formal languages may be characterised as the study of systems that map evidence on a language into *hypotheses* about it. Of special interest is the investigation of scenarios in which the sequence of hypotheses *stabilises* to an *accurate* and *finite* description (a grammar) of the target language. If stabilisation is formalised *syntactically* then we obtain *explanatory learning* (see Case and Smith, 1983), which is also called *learning in the limit* (see Gold, 1967). Replacing syntactically by *semantically* results in *behaviourally correct learning* (see Feldman, 1972; Bārzdiņš, 1974, 1977; Case and Smith, 1983; Case, 1999). A further special case is *finite learning*, where the learner outputs essentially just one hypothesis which has to be a correct one. In the learning models described so far, evidence is provided by successively growing initial segments of any infinite sequence of strings that eventually contains all strings of the target language L and none of the strings outside of L (such a sequence is called a *text* for L or a sequence of *positive data* for L).

The hypotheses output by the learner are natural numbers (also called *indices*). The set of all admissible hypotheses is called the *hypothesis space*. In the present paper the hypothesis space is any fixed *acceptable numbering* $W_0, W_1, W_2, \ldots$ of all recursively enumerable languages, which are identified with the recursively enumerable subsets of the natural numbers, and any index is interpreted as a grammar. We are then in general interested in the learnability of classes $\mathcal{L}$ of languages; i.e., we ask whether or not there is one learner that can learn every language $L \in \mathcal{L}$.

Furthermore, in the present paper we consider learning scenarios in which evidence may also be provided in the form of indices. We introduce the model of *verifiability*, where the evidence is any index e and a text of the target language L, and the learner has to *verify* whether or not the index input is correct for the target language; i.e., whether or not $W_e = L$. Second, we also define *comparability*, where the learner receives two indices of languages from the target class $\mathcal{L}$ as input and has to decide in the limit whether or not these indices generate the same language. Third, we study *standardisability* in the sense that evidence is provided by any index j of a target language L from the class $\mathcal{L}$. Then for every $L \in \mathcal{L}$ there has to exist an index e such that $W_e = L$ and the *standardiser* has to map every index j of L to e. This mapping may be performed in the limit or just by outputting a single guess which then must be e (called *finite standardisability*). Standardisability has been considered previously in the literature (see, e.g., Kinber, 1975; Freivald and Wiehagen, 1979; Freivalds et al., 1984; Jain and Sharma, 1994).

For almost all the learning models described above it is also meaningful to take a closer look at *the number of times a learner changes its mind*. For the technical details we refer the reader to Definition 3.

The main problem studied in the present paper is the question to what extent verifiability, comparability, and standardisability are useful for the inductive inference of classes of recursively enumerable languages. For example, we are interested in learning whether or not a verifiable

class is also explanatorily learnable, and if it is, whether or not the number of mind changes is preserved. Table 1 provides a summary of the results obtained, where the last three columns give the best obtained results for learning an Indexed, One–One R.E. or R.E. class which is n-shot comparable/standardisable/verifiable as provided in the first two columns. (Here n-shot means that the comparator/standardiser/verifier makes at most $(n-1)$-mind changes — that is it gets n-shots at getting the correct answer.) In the table, $\mathrm{Ex}[K]$ denotes Ex-learning using oracle K, and a ? beside a criterion denotes an open problem at this point. An entry Not I denotes that the corresponding comparator/standardizer/verifier notion does not imply I-learnability.

Notation	Shots	Indexed	One-One R.E.	R.E.
Comparator	1	Fin (Thm 11)	Fin (Thm 11)	Fin (Thm 11)
	2	ConsvEx (Thm 21)	Ex (Thm 17), ConsvEx?	Not Ex(Thm 19), ConsvEx$[K]$ (Thm 22)
	3	Ex (Thm 28), Not ConsvEx (Thm 32)	ConsvEx$[K]$ (Thm 29), Not Ex (Thm 26), BC?	Not BC (Prop 30)
	4 and More	Not BC (Thm 33)	Not BC (Thm 33)	Not BC (Thm 33)
Standardiser	1	Fin (Cor 12)	Fin (Cor 12)	Fin (Cor 12)
	2	Ex (Thm 15), Not ConsvEx (Thm 18)	Ex (Thm 15), Not ConsvEx (Thm 18)	Not BC (Prop 30)
	3 and More	Not BC (Thm 33)	Not BC (Thm 33)	Not BC (Thm 33)
Verifier	1	Fin (Prop 10)	Fin (Prop 10)	Fin (Prop 10)
	2	ConsvEx (Thm 20)	ConsvEx (Thm 20)	ConsvEx (Thm 20)
	3 and More	Ex (Prop 24), Not ConsvEx (Thm 32)	Ex (Prop 24), Not ConsvEx (Thm 32)	Ex (Prop 24), Not ConsvEx (Thm 32)

Table 1: Summary of main results

2. Notation

The notation and terminology from recursion theory adopted in this paper follows Rogers (1987). For background on inductive inference we refer the reader to Jain et al. (1999) and Zeugmann and Zilles (2008).

We use $\mathbb{N} = \{0, 1, 2, \ldots\}$ to denote the set of all natural numbers. The set of all *partial recursive functions* and of all *recursive functions* of one, and two arguments over $\mathbb{N}$ is denoted by $\mathcal{P}$, $\mathcal{P}^2$, $\mathcal{R}$,

and $\mathcal{R}^2$, respectively. We write $\mathcal{R}_{\{0,1\}}$ to denote the set of all $\{0,1\}$-valued recursive functions. Let $\varphi_0, \varphi_1, \varphi_2, \ldots$ denote a fixed acceptable programming system of all partial recursive functions and let $W_0, W_1, W_2, \ldots$ be a *acceptable numbering of all recursively enumerable sets* (abbr. r.e. sets) of natural numbers, where W_e is the domain of φ_e for all $e \in \mathbb{N}$. Let $e, x \in \mathbb{N}$; if $\varphi_e(x)$ is defined then we write $\varphi_e(x)\downarrow$ and also say that $\varphi_e(x)$ *converges*. Otherwise, $\varphi_e(x)$ is said to *diverge* (abbr. $\varphi_e(x)\uparrow$). Furthermore, if the computation of $\varphi_e(x)$ halts within s steps of computation then we write $\varphi_{e,s}(x)\downarrow = \varphi_{e(x)}$; otherwise $\varphi_{e,s}(x)\uparrow$. For all $e, s \in \mathbb{N}$ the set $W_{e,s}$ is defined as the domain of $\varphi_{e,s}$. The symbol K denotes the *diagonal halting problem*, i.e., $K = \{e : e \in \mathbb{N},\ \varphi_e(e)\downarrow\}$. We use $C(x)$ to denote the plain Kolmogorov complexity of x (see Li and Vitányi, 2008, chap. 2).

Given any set S, we use S^* to denote the set of all finite sequences of elements from S. By $D_0, D_1, D_2, \ldots$ we denote any fixed *canonical indexing of all finite sets* of natural numbers. We recall that Cantor's pairing function $\langle \cdot, \cdot \rangle \colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ is given by $\langle x, y \rangle = \frac{1}{2}(x+y)(x+y+1)+y$ for all $x, y \in \mathbb{N}$. For any two sets A and B, we define $A \oplus B = \{2x : x \in A\} \cup \{2y+1 : y \in B\}$. Analogously, one defines $A \oplus B \oplus C = \{3x : x \in A\} \cup \{3y+1 : y \in B\} \cup \{3z+2 : z \in C\}$.

We continue with some technical notations needed for our definitions of variants of learnability. In the following we always assume that $\# \notin \mathbb{N}$. Furthermore, for $\sigma \in (\mathbb{N} \cup \{\#\})^*$ and $n \in \mathbb{N}$ we write $\sigma(n)$ to denote the element in the nth position of σ. Additionally, $\sigma[n]$ denotes the sequence $\sigma(0), \sigma(1), \ldots, \sigma(n-1)$. Given a number $a \in \mathbb{N}$ and some fixed $n \in \mathbb{N}$, $n \geq 1$, we denote by a^n the finite sequence $a, \ldots, a$, where a occurs exactly n times. Moreover, we identify a^0 with the empty string λ. For any finite sequence σ we use $|\sigma|$ to denote the length of σ. The concatenation of two sequences σ and τ is denoted by $\sigma \circ \tau$; for convenience, and whenever there is no possibility of confusion, this is occasionally denoted by $\sigma\tau$.

3. Learnability, Standardisability, Comparability and Verifiability

Let $\mathcal{L}$ be a class of r.e. languages. Throughout this paper, the mode of data presentation is that of a *text*. A text is any infinite sequence of natural numbers and the $\#$ symbol, where the symbol $\#$ indicates a pause in the data presentation. More formally, a *text* T_L for a language $L \in \mathcal{L}$ is any total mapping $T_L : \mathbb{N} \to \mathbb{N} \cup \{\#\}$ such that $L = \mathrm{range}(T_L) \setminus \{\#\}$. We use $\mathrm{content}(T)$ to denote the set $\mathrm{range}(T) \setminus \{\#\}$, i.e., the content of a text T contains only the natural numbers appearing in T. Furthermore, for every $n \in \mathbb{N}$ we use $T[n]$ to denote the finite sequence $T(0), \ldots, T(n-1)$, i.e., the *initial segment* of length n of T. Analogously, for a finite sequence $\sigma \in (\mathbb{N} \cup \{\#\})^*$ we use $\mathrm{content}(\sigma)$ to denote the set of all numbers in the range of σ.

3.1. Learning Criteria

The main learning criteria studied in this paper are *explanatory learning* (also called *learning in the limit*) introduced by Gold (1967) and *behaviourally correct learning*, which goes back to Feldman (1972), who called it *matching in the limit*. The name behaviourally correct learning was coined by Case and Smith (1983). It was also studied by Bārzdiņš (1974, 1977) and Case and Lynes (1982). Furthermore, we shall also consider *finite learning* (see Gold, 1967). In the following definitions, an *inductive inference machine* (abbr. IIM) M is a recursive function mapping $(\mathbb{N} \cup \{\#\})^*$ into $\mathbb{N} \cup \{?\}$; the $?$ symbol permits M to abstain from conjecturing at any stage.

Definition 1 *Let $\mathcal{L}$ be any class of r.e. languages.*

(1) An IIM M explanatorily (Ex) learns $\mathcal{L}$ *if, for every* L *in* $\mathcal{L}$ *and each text* T_L *for* L*, there is a number* n *for which* $L = W_{M(T_L[n])}$ *and, for every* $j \geq n$*,* $M(T_L[j]) = M(T_L[n])$.

(2) An IIM M behaviourally correctly (BC) learns $\mathcal{L}$ *if, for every* L *in* $\mathcal{L}$ *and each text* T_L *for* L*, there is a number* n *for which* $L = W_{M(T_L[j])}$ *whenever* $j \geq n$.

(3) An IIM M finitely (FinEx) learns $\mathcal{L}$ *if, for every* L *in* $\mathcal{L}$ *and each text* T_L *for* L*, there is a number* n *for which* $L = W_{M(T_L[n])}$ *and for every* $m < n$*,* $M(T_L[m]) = ?$ *and for every* $j \geq n$*,* $M(T_L[j]) = M(T_L[n])$.

(4) An IIM M $(n+1)$-shot learns $\mathcal{L}$ *if it Ex learns* $\mathcal{L}$ *and for every text* T *for an* $L \in \mathcal{L}$ *there are at most* n *distinct values* k *for which* $M(T[k]) \neq ?$ *and* $M(T[k]) \neq M(T[k+1])$.

Note that for finite learning the IIM itself indicates that it has successfully finished its learning task, since the first hypothesis output, which is different from ?, is correct.

We shall also consider the learning constraint of *conservativeness* (see Angluin, 1980). A learner M is said to be *conservative* if the following condition is satisfied: If M on input $T[n]$ makes the guess j_n and then makes a different guess $j_{n+k} \neq j_n$ at some subsequent step on input $T[n+k]$, where $k \geq 1$, and $j_n \neq ? \neq j_{n+k}$ then $\text{content}(T[n+k]) \not\subseteq W_{j_n}$. By ConsvEx and ConsvBC we denote the learning criteria obtained by combining conservativeness with Ex learnability and BC learnability, respectively.

In some cases we consider learners using oracles. In this case the learning criterion I, when the learners are allowed use of oracle A, is denoted by $I[A]$. For every learning criterion I considered in the present paper, there exists a recursive enumeration $M_0, M_1, \ldots$ of the learning machines such that if a class is I-learnable, then some M_i I-learns the class. We fix one such enumeration of learning machines. A class $\mathcal{L}$ is said to be *uniformly r.e.* (or just *r.e.*) if there is an r.e. set $S \subseteq \mathbb{N}$ such that $\mathcal{L} = \{W_i : i \in S\}$. A class is said to be *1–1 r.e.*, if the r.e. set S as above additionally satisfies the condition that for $i, j \in S$, $W_i = W_j$ iff $i = j$. An r.e. class as above is said to be *uniformly recursive* or an *indexed family* if there exists a recursive function $f \in \mathcal{R}_{\{0,1\}}$ such that for all $i \in S$ and $x \in \mathbb{N}$, $f(i,x) = 1$ iff $x \in W_i$.

Angluin (1980) considered an important condition for learnability of classes based on *tell-tale sets* as defined below.

Definition 2 *Assume* $\mathcal{L}$ *is a class of languages. A set* S *is said to be a* tell-tale *with respect to* $\mathcal{L}$ *for a language* $L \in \mathcal{L}$ *if* $S \subseteq L$*,* S *is finite and for all* $L' \in \mathcal{L}$*,* $S \subseteq L' \subseteq L$ *implies* $L = L'$*. The class* $\mathcal{L}$ *satisfies the tell-tale property if every* $L \in \mathcal{L}$ *has a tell-tale with respect to* $\mathcal{L}$.

Every behaviourally correctly learnable class of languages satisfies the tell-tale property (see Angluin, 1980; Baliga et al., 1999). A *characteristic sample* (Lange and Zeugmann, 1992; Mukouchi, 1992) of a language L (with respect to a class $\mathcal{L}$) is a finite subset S of L such that for all $L' \in \mathcal{L}$, $L \neq L'$ implies $S \not\subseteq L'$. A class $\mathcal{L}$ of languages is said to be *inclusion-free* if there are no two languages A and B in the class $\mathcal{L}$ such that $A \subset B$.

3.2. Criteria for Verifiability, Comparability and Standardisability

In this paper, we consider a learning scenario in which the learner may not be required to output a correct index for the target language (in the limit), but only has to (iii) *verify* whether or not a given index is correct, or (i) to decide whether or not any two given r.e. indices of sets in the target class are

indices for the same set. We shall also study classes of r.e. languages that can be *finitely standardised* in the sense that for a target class $\mathcal{L}$, there exists a partial-recursive function f such that for any given language L in $\mathcal{L}$, there is an e with $W_e = L$ for which f outputs e when fed with *any* r.e. index for L. *Mutatis mutandis*, replacing the function f by a limiting recursive function yields *standardisation* (in the limit) of languages (see (ii) below). Finite standardisation was introduced by Freivald and Wiehagen (1979), while standardisation goes back to Kinber (1975). Their motivation was to study whether a given index of the object to be learnt is more useful than the graph of a target function or a text of the target language. It was further investigated by Freivalds et al. (1984) for function learning, and by Jain and Sharma (1994) for the inductive inference of r.e. languages.

Definition 3 *Let $\mathcal{L}$ be any class of r.e. languages.*

(i) The class $\mathcal{L}$ is said to be $(n+1)$-shot comparable *if there is a partial-recursive function $F\colon \mathbb{N}\times\mathbb{N}\times\mathbb{N}\to\{yes, no, ?\}$ (called an $(n+1)$-*shot comparator for $\mathcal{L}$*) such that for all $i, j \in \mathbb{N}$ with $W_i, W_j \in \mathcal{L}$,*

(1) $G(i,j) := \lim_{k\to\infty} F(i,j,k)$ exists and $G(i,j) \in \{yes, no\}$;

(2) $G(i,j) = yes$ if $W_i = W_j$ and $G(i,j) = no$ if $W_i \neq W_j$;

(3) there are at most n distinct values of k for which $F(i,j,k) \in \{yes, no\}$, and $F(i,j,k) \neq F(i,j,k+1)$.

Intuitively, this means that F "changes its value (or mind)" at most n times.[1]

A 1-shot comparable class will also be called finitely comparable.

(ii) The class $\mathcal{L}$ is said to be $(n+1)$-shot standardisable *if there is a partial-recursive function $F\colon \mathbb{N}\times\mathbb{N}\to\mathbb{N}\cup\{?\}$ (called an $(n+1)$-*shot standardiser for $\mathcal{L}$*) such that*

(1) for all i with $W_i \in \mathcal{L}$, $G(i) := \lim_{k\to\infty} F(i,k)$ exists, $G(i) \in \mathbb{N}$ and $W_{G(i)} = W_i$;

(2) for all $i, j \in \mathbb{N}$ with $W_i, W_j \in \mathcal{L}$, $G(i) = G(j)$ iff $W_i = W_j$;

(3) there are at most n distinct values of k for which $F(i,k) \in \mathbb{N}$ and $F(i,k) \neq F(i,k+1)$.

Intuitively, this means that F "changes its value" at most n times.

A 1-shot standardisable class will also be called finitely standardisable.

(iii) M is said to verify *$\mathcal{L}$ if on any input given by an index e for a language in $\mathcal{L}$ and text T for some perhaps different language in $\mathcal{L}$, M converges to "yes" in the case that e is an index for the language of the text and "no" if that is not the case; M is called a* verifier for $\mathcal{L}$. *One can similarly define finite verifiability and $(n+1)$-shot verifiability.*

4. Results

In this section all results are presented. They are grouped with respect to the number of shots a verifier, comparator, and standardiser is allowed to make. We then compare the power of these models to the more standard models of learnability.

1. In order to minimise notation, we shall often omit the third argument in the definition of F and simply write "F changes its value at most n times", where F is a given $(n+1)$-shot comparator for a class; the meaning of this statement will be clear from the context. A similar remark applies to any $(n+1)$-shot standardiser.

4.1. Comparison of Learnability With 1-Shot Verifiability, 1-Shot Comparability and 1-Shot Standardisability

We start this section with some examples that compare the power of verifiers with finite learnability and explanatory learnability. Taking into account that the collection of all finitely learnable classes is a proper subset of the collection of all explanatorily learnable classes, these examples show that the power of verifiers is incomparable to the power of explanatory learners.

Example 4 *The class $\mathcal{K}$ consisting of K and all singleton languages $\{x\}$ with $x \notin K$ is finitely learnable but not finitely verifiable.*

Note that the second part of the proof in Example 4 (cf. Section 5.1) also shows that $\mathcal{K}$ is a uniformly r.e. class. We continue with further properties of the r.e. class $\mathcal{K}$, which, as we have shown in Example 4, is finitely learnable but not finitely verifiable.

Example 5 *The uniformly r.e. class $\mathcal{K}$ is neither finitely comparable nor finitely standardisable.*

Remark Example 5 shows in particular that finite learnability does *not* imply finite standardisability if learning of r.e. languages is considered. This contrasts the corresponding result for learning classes of recursive functions, where every finitely learnable function class is also finitely standardisable (see Freivald and Wiehagen, 1979).

Example 6 *Consider a class $\mathcal{L}$ of languages which contains for every $e \in \mathbb{N}$ exactly one language $L_e \subseteq \{\langle e, 0\rangle, \langle e, 1\rangle, \ldots\}$ which is not empty and not behaviourally correctly learnt by the learner M_e. To choose such an L_e let $X_i = \{\langle e, x\rangle : x \in W_i\}$. Now $\{X_i : i \in \mathbb{N}\}$ is not behaviourally correctly learnable by learner M_e as otherwise the class of all r.e. sets would be behaviourally correctly learnable, contradicting Case and Lynes (1982). Thus, one can choose L_e to be some nonempty X_i which is not behaviourally correctly learnable by M_e.*

The class $\mathcal{L}$ is then verifiable by an algorithm which on input of an index d and a text T enumerates the set W_d and tracks the text until a member (i, s) is enumerated into W_d and a non-pause datum (j, t) is found in the text. Then the index d is for the language of T iff $i = j$, as the verifier needs only to work on members of the class. However, by choice, $\mathcal{L}$ is not behaviourally correctly learnable.

Example 7 *Assume that each language in $\mathcal{L}$ contains exactly one even element and that no two languages in $\mathcal{L}$ contain the same even element. Then $\mathcal{L}$ is finitely verifiable and finitely comparable. However, one can choose the sets in $\mathcal{L}$ such that the e-th behaviourally correct learner does not converge to the right index on the text for the language in $\mathcal{L}$ with the even element $2e$ and similarly one can also fool the standardiser and thus this class $\mathcal{L}$ can be chosen such that it is neither behaviourally correctly learnable nor standardisable in the limit.*

In order to establish a connection between learnability and finite standardisability as well as finite comparability, we first make a general observation. Assume that a class $\mathcal{L}$ has an n-shot standardiser. Then the class is $(2n-1)$-shot comparable. This can be seen as follows: Let d and e be two indices such that both W_d and W_e are in $\mathcal{L}$. Then the algorithm runs two instances of the standardiser in parallel on the two inputs d and e, respectively, and waits until each of them has produced an output. From then on, if the current outputs of the two instances are equal, then the algorithm outputs "yes",

and if the two outputs are different, then it outputs "no". Note that the two instances must produce at least one output each, since both W_d and W_e are in $\mathcal{L}$. Once this is done, every mind change of the comparator requires that at least one of the standardisers makes another shot and so one can bound the number of shots of the comparator by $1 + 2 \cdot (n-1) = 2n - 1$. Hence, we have the following proposition:

Proposition 8 *Every n-shot standardisable class is $(2n-1)$-shot comparable.*

We next show that 2-shot comparable classes and finitely standardisable classes have a rather restrictive structure.

Proposition 9 *Any class which is 2-shot comparable must be inclusion-free. Thus, any class which is finitely standardisable must be inclusion-free.*

Proof If the class contains two sets A and B with $A \subset B$, and has a 2-shot comparator F, then using the double recursion theorem (see Smullyan, 1961), one can construct two grammars i and j such that $W_i = W_j = A$, if the comparator never outputs "yes" on input (i, j). If the comparator outputs "yes" on input (i, j) at some point, and then never outputs "no" after that, then $W_i = A$ and $W_j = B$. Otherwise, $W_i = W_j = B$. So it follows that the comparator is wrong on input (i, j). ■

The next three results show that with respect to uniformly r.e. classes, finite verifiability, finite standardisability and finite comparability are so restrictive that each of them already implies finite learnability.

Proposition 10 *Let $\mathcal{L}$ be a uniformly r.e. class. Then we have the following: If $\mathcal{L}$ is finitely verifiable, then $\mathcal{L}$ is also finitely learnable.*

Proof If a class $\mathcal{L}$ has a recursively enumerable list $e_0, e_1, e_2, \ldots$ of indices, then a finite verifier can be turned into a finite learner by dovetailing the enumeration of the indices $e_0, e_1, \ldots$ and by simulating the verifier on e_0 versus T, e_1 versus T, ... until one of them outputs "yes". The finite learner then conjectures the first index e_k, where the simulation gives the answer "yes". ■

Theorem 11 *Every uniformly r.e. and finitely comparable class $\mathcal{L}$ is finitely learnable.*

The next corollary follows from Proposition 8 and Theorem 11.

Corollary 12 *Every uniformly r.e. and finitely standardisable class $\mathcal{L}$ is finitely learnable.*

Remark A slight modification of the proof of Theorem 11 (cf. Section 5.3) gives that whenever a uniformly r.e. class $\mathcal{L}$ is finitely comparable then $\mathcal{L}$ is also finitely standardisable and finitely verifiable: The verifier just checks whether on input of a text T and an index e the above constructed learner M outputs on the text T an index d with $F(d, e)$ having the value "yes"; the standardiser outputs for an index d the first index e_k in the given recursive enumeration, where $F(d, e_k)$ has the value "yes".

We continue with further results concerning finite standardisation, finite verifiability, finite learning, and explanatory learning, when the classes considered may not be recursively enumerable. We formulate the following example in terms of function learning. Note that function learning is just a special case of language learning, since learning functions is in general equivalent to learning their graphs as sets from text.

Example 13 *This example is for function learning, for ease of notation. The example can be easily modified for language learning by considering graphs of functions rather than the functions themselves. We define* $f_0(x) := 0$ *for all* $x \in \mathbb{N}$*, and for* $n \geq 1$ *we set* $f_n(x) := 0$ *for all* $x \in \mathbb{N} \setminus \{n\}$ *and* $f_n(n) := 1$*. Furthermore, we use* $\min_\varphi f$ *to denote the least index* i *such that* $\varphi_i = f$*. Next, consider the class* $\mathcal{F} := \{f_n \mid n \in \mathbb{N},\ n \leq \min_\varphi f_n\}$*. Then* $\mathcal{F}$ *is finitely standardisable but neither finitely learnable nor finitely verifiable. Note that* $\mathcal{F}$ *is not an r.e. class.*

Finite standardisability of $\mathcal{F}$ *as well as that* $\mathcal{F}$ *is not finitely learnable has been shown by Freivald and Wiehagen (1979).*

In order to see that $\mathcal{F}$ *is not finitely verifiable let* $e \in \mathbb{N}$ *be an index for* f_0*, and let* T *be a text for* f_0 *such that* $T = ((0, f_0(0)), (1, f_0(1)), (2, f_0(2)) \ldots)$*. Then, on input* e *and* T*, a finite verifier would have to eventually output "yes", since otherwise it could not verify that* T *is a text for the function* f_0*. Let this happen when the verifier has seen* $T[m]$*. Consequently, for every* $n > m$ *and a text* T' *in the same order* $(0, f_n(0)), (1, f_n(1)), (2, f_n(2)) \ldots$ *for* f_n *it must, on input* e *and* T'*, also output "yes", a contradiction to the fact that* T' *is not a text for* f_0*.*

Examples 4, 5, 6 and 13 show both that the collection of finitely learnable classes is incomparable to the collections of finitely standardisable classes and to the collection of finitely verifiable classes.

Next, we ask whether or not finite standardisability is of any help for explanatory learning, when considering possibly non r.e. classes. This question deserves attention, since it deals with the problem of information presentation versus the mode of convergence. The affirmative answer is given below.

Theorem 14 *If a class* $\mathcal{L}$ *is finitely-standardisable then* $\mathcal{L}$ *is explanatorily learnable.*

4.2. Comparing Learnability With 2-Shot Verifiability, 2-Shot Comparability and 2-Shot Standardisability

We begin by showing that 2-shot standardisability of one-one r.e. classes implies explanatory learnability.

Theorem 15 *If a class has a 2-shot standardiser and has a one-one r.e. numbering, then it is explanatorily learnable.*

Example 16 *Consider the class of graphs of the functions with only finitely many non-zero values. This class is 2-shot comparable and 2-shot verifiable and has a one-one enumeration. This holds as a comparator, on input* i *and* j*, can start with "yes", and then change its mind to "no" when a convergent error is found, that is, there exist* x, y, z *with* $y \neq z$ *such that* $(x, y) \in W_i$ *and* $(x, z) \in W_j$*. Similarly, a 2-shot verifier can be constructed. Furthermore, the class is clearly one-one enumerable.*

However, this class cannot be learnt by any confident learner even if it uses an oracle as the graph of any finite function has an extension in the class. Thus there is no upper bound on the number of shots of the learner.

Here a confident learner *(see Osherson et al., 1986) is a learner which converges to some hypothesis on all texts, even for texts for languages not in the class being learnt.*

An analogue of Algorithm 1 (see the proof of Theorem 15) shows that any 2-shot comparable r.e. class with a one-one numbering is explanatorily learnable.

Theorem 17 *If a class $\mathcal{L}$ has a 2-shot comparator and a one-one r.e. numbering, then $\mathcal{L}$ is explanatorily learnable.*

Recall that a conservative learner revises its conjecture only if the content of the present text segment is not contained in the learner's original conjecture. Intuitively speaking, a conservative learner performs *exclusively* justified mind changes. In particular, a conservative learner can never *overgeneralise*; i.e., it can never guess a proper superset of the target language (see, e.g., Angluin, 1980; Zeugmann et al., 1995; Jain et al., 1999).

Theorem 18 *There exists a 2-shot standardisable indexed family which is not conservatively learnable.*

Although 1-shot comparable uniformly r.e. classes are always finitely learnable (cf. Theorem 11), we next show that 2-shot comparable r.e. classes may not be learnable in the limit (cf. Theorem 19).

Theorem 19 *There is an r.e. 2-shot comparable class which is not explanatorily learnable.*

Theorem 20 *If a uniformly r.e. class $\mathcal{L}$ is 2-shot verifiable, then it is conservative explanatorily learnable.*

As any class which is 2-shot comparable is inclusion-free (cf. Proposition 9), it is easy to show the next result.

Theorem 21 *Every indexed family which is 2-shot comparable is conservative explanatorily learnable.*

Furthermore, as the membership problem for indices of r.e. sets can be solved using K as an oracle, we obtain the following theorem:

Theorem 22 *Every uniformly r.e. class $\mathcal{L}$ which is 2-shot comparable is ConsvEx$[K]$-learnable.*

4.3. Comparison of Learnability With 3 Or More Shot Verifiability, 3 Or More Shot Comparability and 3 Or More Shot Standardisability

In this section, we compare learnability with verifiability, comparability and standardisability when more than 2-shots are allowed. We begin with a simple example to show that the power of comparability increases with the number of shots allowed.

Example 23 *For each $n \geq 1$, let FINS$_n$ be the class of all sets of cardinality at most n. Then FINS$_n$ is $(2n+1)$-shot comparable but not $2n$-shot comparable.*

The following result is due to the learning algorithm, which for a recursive list of indices $e_0, e_1, \ldots$ of a class converges to the first index e_k where the verifier is, on the text T, converging to "yes" while the verifier converges, for all previous indices $e_0, \ldots, e_{k-1}$, to "no".

Proposition 24 *If an r.e. class is verifiable in the limit then it is also explanatorily learnable.*

The next lemma states that any uniformly r.e. class of infinite sets that is n-shot comparable can be expanded to a 1–1 r.e. class that is $(n+1)$-shot comparable. This will allow us to extend Theorem 19 (albeit in a weaker form) to 1–1 r.e. families.

Lemma 25 *Assume $\mathcal{L} = \{W_{f(i)} : i \in \mathbb{N}\}$ is an r.e. class of infinite languages, each of which is contained in $\{\langle 1, x\rangle : x \in \mathbb{N}\}$, where $f \in \mathcal{R}$. Suppose that $\mathcal{L}$ is n-shot comparable. Then there is a 1–1 r.e. class $\mathcal{L}'$ of infinite languages such that $\mathcal{L} \subseteq \mathcal{L}'$ and $\mathcal{L}'$ is $(n+1)$-shot comparable.*

We now deduce the following theorem from Lemma 25 and Theorem 19:

Theorem 26 *There is a 1–1 r.e. 3-shot comparable class which is not explanatorily learnable.*

Proof One can cylindrify the 2-shot comparable and uniformly r.e. class $\{L_{e,x} : e, x \in \mathbb{N}\}$ constructed in the proof of Theorem 19; i.e. we define $U_{e,x} = \{\langle 1, \langle y, z\rangle\rangle : y \in L_{e,x} \wedge z \in \mathbb{N}\}$ for all $e, x \in \mathbb{N}$, to obtain a uniformly r.e. class $\mathcal{L}$ of infinite languages which is contained in $\{\langle 1, z\rangle : z \in \mathbb{N}\}$. Note that $\mathcal{L}$, like the uniformly r.e. class in the proof of Theorem 19, is not explanatorily learnable. Lemma 25 then gives a 1–1 r.e. class $\mathcal{L}'$ such that $\mathcal{L}' \supseteq \mathcal{L}$, $\mathcal{L}'$ is 3-shot comparable and $\mathcal{L}'$ is not explanatorily learnable. ■

We have seen that for indexed families, 1-shot comparability implies finite learnability (Theorem 11) while 2-shot comparability implies conservatively explanatory learnability (Theorem 21). The next theorem completes this fairly neat hierarchy of learnability notions implied by n-shot comparability for indexed families: 3-shot comparability, while insufficient for guaranteeing conservative learnability (see Theorem 32), still implies explanatory learnability. We first show that 3-shot comparable 1-1 r.e. classes have uniformly r.e. families of finite tell-tale sets.

Proposition 27 *Assume $\mathcal{L} = \{U_0, U_1, \ldots\}$ is a 1–1 r.e. class with $U_i \neq U_j$ for i, j with $i \neq j$. Suppose $\mathcal{L}$ is 3-shot comparable. Then one can effectively (in i) enumerate tell-tale sets for U_i with respect to $\mathcal{L}$.*

The following theorem now follows from the characterisation of explanatory learnability using tell-tale sets by Angluin (1980) (and the fact that every indexed family containing infinitely many distinct sets has a 1–1 r.e. numbering).

Theorem 28 *Any 3-shot comparable indexed family is explanatorily learnable.*

When given an oracle for K, a 1–1 r.e. family behaves like an indexed family for the purposes of the earlier proof of Proposition 27. Furthermore, using an oracle for K and an analogue of the construction of $W_{g(e,d)}, W_{h(e,d)}$ in Proposition 27, one can determine for each member L of a 1–1 r.e. family $\mathcal{L} = \{U_0, U_1, U_2, \ldots\}$ a finite tell-tale set for L w.r.t. $\mathcal{L}$: first, test whether or not there exists a j' for which $X_{i,j'}$ (as constructed in the proof of Proposition 27) is nonempty; if no such j' exists, then the empty set is a tell-tale; otherwise, search for the first j such that $X_{i,j}$, a tell-tale for L w.r.t. $\mathcal{L}$, is defined. Thus, on any input σ, a learner equipped with an oracle for K may test whether or not a potential hypothesis U_i contains $\mathrm{content}(\sigma)$ as well as whether a tell-tale for U_i found earlier is contained in $\mathrm{content}(\sigma)$, thereby ensuring that it is conservative.

Consequently, we have the following result:

Theorem 29 *Any* 3*-shot comparable 1–1 r.e. family is ConsvEx*$[K]$ *learnable.*

In order to show further results we need the following:

Proposition 30 *The r.e. class* $\mathcal{L}$ *consisting of* $\mathbb{N}$ *and all sets* $\mathbb{N} \setminus \{x\}$ *with* $C(x) \geq \log(x)$ *has the following properties:*

(i) The class is not behaviourally correctly learnable;

(ii) the class is 2*-shot standardisable but not finitely standardisable;*

(iii) the class is 3*-shot comparable but not* 2*-shot comparable;*

(iv) the class is not verifiable.

Proposition 31 *If a class is* n*-shot learnable then it has a* $2n$*-shot comparator.*

Next we show that Theorem 20 and Theorem 21 do not generalise to 3-shot verification/comparable classes.

Theorem 32 *For indexed families* 3*-shot comparability or* 3*-shot verifiability do not imply conservative learnability.*

At this point it is only natural to ask what holds for 4-shot comparators. The answer is provided by the following theorem, where U denotes a fixed universal Turing machine used to define the Kolmogorov complexity.

Theorem 33 *There is an indexed family* $\mathcal{L}$ *which is* 4*-shot comparable and* 3*-shot standardisable but not behaviourally correct learnable.*

Consequently, we have shown all results presented at the end of the Introduction in Table 1.

Acknowledgments

Sanjay Jain was supported in part by NUS grant C252-000-087-001. Furthermore, Ziyuan Gao, Sanjay Jain and Frank Stephan have been supported in part by the Singapore Ministry of Education Academic Research Fund Tier 2 grant MOE2016-T2-1-019 / R146-000-234-112. This research was performed partially while the fourth author was visiting the Institute of Mathematical Sciences at the National University of Singapore in September 2017. His visit was partially supported by the Institute. Thomas Zeugmann was also supported in part by the Japan Society for the Promotion of Science, and the Grand-in-Aid for Challenging Exploratory Research No. 15K12022.

We thank the anonymous referees of the conference for several useful comments.

References

Dana Angluin. Inductive inference of formal languages from positive data. *Information and Control*, 45(2):117–135, 1980.

Ganesh Baliga, John Case, and Sanjay Jain. The synthesis of language learners. *Information and Computation*, 152(1):16–43, 1999.

Janis M. Bārzdiņš. Two theorems on the limiting synthesis of functions. In Janis M. Bārzdiņš, editor, *Theory of Algorithms and Programs I*, volume 210 of *Proceedings of the Latvian State University*, pages 82–88. Latvian State University, Riga, 1974. (in Russian).

Janis M. Bārzdiņš. Inductive inference of automata, functions and programs. In *American Mathematical Society Translations*, pages 107–122, 1977. Appeared originally in the Proc. of the 20-th International Congress of Mathematicians 1974, Volume 2, pp. 455–460, (in Russian).

John Case. Periodicity in generations of automata. *Mathematical Systems Theory*, 8(1):15–32, 1974.

John Case. The power of vacillation in language learning. *SIAM Journal on Computing*, 28(6): 1941–1969, 1999.

John Case and Christopher Lynes. Machine inductive inference and language identification. In Mogens Nielsen and Erik Meineche Schmidt, editors, *Automata, Languages and Programming, Ninth Colloquium, Aarhus, Denmark, July 12-16, 1982, Proceedings*, volume 140 of *Lecture Notes in Computer Science*, pages 107–115. Springer-Verlag, 1982.

John Case and Carl Smith. Comparison of identification criteria for machine inductive inference. *Theoretical Computer Science*, 25(2):193–220, 1983.

Jerome Feldman. Some decidability results on grammatical inference and complexity. *Information and Control*, 20(3):244–262, 1972.

R. V. Freivald and R. Wiehagen. Inductive inference with additional information. *Elektronische Informationsverarbeitung und Kybernetik*, 15(4):179–185, 1979.

Rūsiņš Freivalds, Efim B. Kinber, and Rolf Wiehagen. Connections between identifying functionals, standardizing operations, and computable numberings. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 30(9-11):145–164, 1984.

Ziyuan Gao, Frank Stephan, and Sandra Zilles. Partial learning of recursively enumerable languages. *Theoretical Computer Science*, 620:15–32, 2016.

E. Mark Gold. Language identification in the limit. *Information and Control*, 10(5):447–474, 1967.

Sanjay Jain and Arun Sharma. Characterizing language identification by standardizing operations. *Journal of Computer and System Sciences*, 49(1):96–107, 1994.

Sanjay Jain, Daniel Osherson, James S. Royer, and Arun Sharma. *Systems that Learn: An Introduction to Learning Theory, second edition*. MIT Press, Cambridge, Massachusetts, 1999.

Efim B. Kinber. On comparison of limit identification and standardization of general recursive functions. In Janis M. Bārzdiņš, editor, *Theory of Algorithms and Programs II*, volume 233 of *Proceedings of the Latvian State University*, pages 45–56. Latvian State University, Riga, 1975. (in Russian).

Steffen Lange and Thomas Zeugmann. Types of monotonic language learning and their characterization. In David Haussler, editor, *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory, July 27-29, 1992, Pittsburgh, Pennsylvania*, pages 377–390, New York, NY, 1992. ACM Press.

Ming Li and Paul Vitányi. *An Introduction to Kolmogorov Complexity and its Applications*. Springer, third edition, 2008.

Yasuhito Mukouchi. Characterization of finite identification. In K.P. Jantke, editor, *Analogical and Inductive Inference, International Workshop AII '92, Dagstuhl Castle, Germany, October 1992, Proceedings*, volume 642 of *Lecture Notes in Artificial Intelligence*, pages 260–267, 1992.

Daniel N. Osherson, Michael Stob, and Scott Weinstein. *Systems That Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists*. The MIT Press, Cambridge, Massachusetts, 1986.

Hartley Rogers, Jr. *Theory of Recursive Functions and Effective Computability*. The MIT Press, Cambridge, Massachusetts, 1987.

Raymond M. Smullyan. *Theory of Formal Systems*, volume 47 of *Annals of Mathematics Studies*. Princeton University Press, Princeton, New Jersey, USA, 1961.

Thomas Zeugmann and Sandra Zilles. Learning recursive functions: A survey. *Theoretical Computer Science*, 397(1-3):4–56, 2008.

Thomas Zeugmann, Steffen Lange, and Shyam Kapur. Characterizations of monotonic and dual monotonic language learning. *Information and Computation*, 120(2):155–173, 1995.

5. Appendix

This appendix contains the proofs not presented in the main part.

5.1. Proof of Example 4

Proof A finite verification algorithm on an index e for K and a text $x, x, x, x, \ldots$ would have to eventually output "no" iff $x \notin K$ and that would, combined with the enumeration procedure of K, lead to a decision procedure of K.

Let $\psi(x,y) = 0$, if $y = x$; $\psi(x,y) = \varphi_x(x) * \varphi_y(y)$, otherwise. Now, if $x \notin K$, then $\psi(x,y)$ diverges for all $y \neq x$, and thus $\mathrm{domain}(\psi(x,\cdot)) = \{x\}$. If $x \in K$, then for all $y \neq x$, $\psi(x,y)$ converges iff $y \in K$, and thus $\mathrm{domain}(\psi(x,\cdot)) = \{x\} \cup K = K$.

Now, the finite learner outputs the canonical index of domain of $\psi(x,\cdot)$ in the acceptable numbering $W_0, W_1, W_2, \ldots$ on input of any text T, if the first non-# symbol in the text T is x (that is, for some i, $T(i) = x$, and $T(i') = \#$ for all $i' < i$). It is easy to verify that the above learner finitely learns the class $\mathcal{K}$. ■

5.2. Proof of Example 5

Proof The proof is in both cases indirect. Suppose the converse, and let $k \in \mathbb{N}$ be any fixed index of K, i.e., $W_k = K$. Then one can design an algorithm deciding K. This algorithm uses the numbering ψ constructed in the second part of the proof of Example 4. Since the numbering of the sets $W_0, W_1, W_2, \ldots$ is acceptable, there is a recursive function $c \in \mathcal{R}$ such that $\psi_x = \varphi_{c(x)}$ for all $x \in \mathbb{N}$. Hence $W_{c(x)}$ is equal to K iff $x \in K$ and equal to $\{x\}$ iff $x \notin K$.

Consequently, for every $x \in \mathbb{N}$ one runs the finite comparator on input $c(x)$ and k. Note that by construction, $W_k, W_{c(x)} \in \mathcal{K}$ for all $x \in \mathbb{N}$, and thus the finite comparator must be defined on all these inputs. Also, it must return "yes" iff $W_{c(x)} = K$ and "no" otherwise; a contradiction to the undecidability of the set K.

For the finite standardiser the algorithm works *mutatis mutandis*. First, we run it on input k to find out to which index k is finitely standardised, say to s. In order to decide K one executes the finite standardiser on input $c(x)$ for any given $x \in \mathbb{N}$. If it returns s, then $x \in K$ and otherwise $x \notin K$. ■

5.3. Proof of Theorem 11

Proof Suppose $\mathcal{L}$ is a finitely comparable class as witnessed by comparator F. First, by Proposition 9, the class $\mathcal{L}$ must be inclusion-free.

Second if $\mathcal{L}$ is a uniformly r.e. class with an r.e. list $e_0, e_1, e_2, \ldots$ of indices covering the class and if $\mathcal{L}$ is inclusion-free and finitely comparable, then one constructs for each k a set $W_{h(k)}$ which is related to e_k as outlined below.

For each e_k, there is an index $h(k)$ such that $W_{h(k)}$ starts enumerating the elements enumerated into W_{e_k} until $F(e_k, h(k))$ terminates and outputs "yes". When this has happened, let $D_{h'(k)}$ denote the set, given by canonical index $h'(k)$, of elements enumerated so far. Now the algorithm searches for an index e_ℓ such that $D_{h'(k)} \subseteq W_{e_\ell}$ and $F(e_k, e_\ell)$ has the value "no". If this search terminates then $W_{h(k)} = W_{e_\ell}$ else $W_{h(k)} = D_{h'(k)}$.

Third, this information can now be used to make the following finite learner M: M reads elements of the text T until a k is found such that all members of $D_{h'(k)}$ have been enumerated into T and then M conjectures e_k for this k.

Fourth, for the verification of correctness, note that for each index e_k, the function $F(e_k, h(k))$ must output "yes" as otherwise $W_{e_k} = W_{h(k)}$ and both $e_k, h(k)$ are indices for the same set in the class without F indicating this. Thus also $h'(k)$ is defined for all k. Now, for each e_k, $D_{h'(k)} \subseteq W_{e_k}$ and W_{e_k} is the only superset of $D_{h'(k)}$ in the r.e. class. The reason is that in the search for e_ℓ after the definition of $D_{h'(k)}$, the search cannot terminate for any k as otherwise either $F(e_k, h(k))$ or $F(e_k, e_\ell)$ for the ℓ found in the search are wrong; when $F(e_k, h(k))$ is right then $W_{e_k} = W_{e_\ell}$ and $F(e_k, e_\ell)$ had wrongly said that they are different. So W_{e_k} is the only superset in the class of $D_{h'(k)}$, although there might be several indices for W_{e_k} in the r.e. enumeration $e_0, e_1, \ldots$ of the indices. From the above it is easy to see that the learning algorithm is correct. ■

5.4. Proof of Theorem 14

Proof Assume that F is a finitely standardising function for $\mathcal{L}$, and let $S = \{e : e \in \mathbb{N},\ F(e) = e\}$.

Using the operator recursion theorem (see Case, 1974), let g be a recursive function such that

$$W_{g(e',e)} = \begin{cases} W_{e'}\,, & \text{if } F(g(e',e)) \text{ is undefined or not equal to } e'\,; \\ W_{e',t(e',e)}\,, & \text{if } F(g(e',e)) = e' \text{ in exactly } t(e',e) \text{ steps} \\ & \text{but } W_{e',t(e',e)} \not\subseteq W_e\,; \\ W_e\,, & \text{otherwise}\,. \end{cases}$$

As in the definition of $W_{g(e',e)}$, above let $t(e',e)$ denote the time needed for $F(g(e',e))$ to converge. Note that if $e, e' \in S$, $e \neq e'$, and $F(g(e',e)) = e'$, then either W_e does not contain $W_{e',t(e',e)}$ or W_e is not in $\mathcal{L}$ (as F outputs two different values on the indices e and $g(e',e)$ for it).

Assume that T is the input text for a language in $\mathcal{L}$. Let $e_0, e_1, \ldots,$ be a 1–1 enumeration of S. On input $T[n]$, the learner outputs e_i for the least $i \leq n$ such that e_i is not eliminated. Here e_i is eliminated if there exists a $j \neq i$, $j \leq n$ such that $F(g(e_j, e_i)) = e_j$ within n steps and $W_{e_j,t(e_j,e_i)} \subseteq \text{content}(T[n])$ — note that in this case e_i is not the index for input language as either $W_{e_j,t(e_j,e_i)}$ and thus $\text{content}(T[n])$ is not contained in W_{e_i} or $F(e_i) \neq F(g(e_j, e_i))$ but $W_{g(e_j,e_i)} = W_{e_i}$; if no such i exists, then the output of the learner does not matter and can be anything.

Assume $W_{e_r} \in \mathcal{L}$, and T is a text for it. Then, eventually, all e_i, $i < r$ will be eliminated (that is not output by the learner above, on input $T[n]$ for large enough n) as (i) $F(g(e_r, e_i)) = e_r$ (otherwise, $W_{g(e_r,e_i)} = W_{e_r}$, but $F(g(e_r, e_i)) \neq F(e_r)$, contradicting the finite-standardisability of $\mathcal{L}$ by F), and (ii) $W_{e_r,t(e_r,e_i)} \subseteq W_{e_r} = \text{content}(T)$. Furthermore, e_r is never eliminated as for all e_i, $i \neq r$, $F(g(e_i, e_r)) \neq e_i$ or $W_{e_i,t(e_i,e_r)} \not\subseteq W_{e_r}$ (as otherwise $W_{g(e_i,e_r)} = W_{e_r}$, and therefore, $F(g(e_i, e_r)) = e_r \neq e_i$ by standardisability).

It follows that the above learner on T converges to e_r. Thus, the above learner Ex-learns $\mathcal{L}$. ■

5.5. Proof of Theorem 15

Proof Suppose F is a 2-shot standardiser for $\mathcal{L} = \{W_{h(0)}, W_{h(1)}, \ldots\}$, where h is a recursive function such that $W_{h(i)} \neq W_{h(j)}$ for i, j with $i \neq j$.

Below, we shall have that grammars $h(j)$ would be either in group 1 or group 2. They are in group 1, if $F(h(j))$ outputs at most one distinct grammar, and in group 2, if $F(h(j))$ outputs two distinct grammars. Let $F^1(h(j))$ denote the first and $F^2(h(j))$ denote the second grammar (if any) output by F on $h(j)$.

Algorithm 1: The construction of $W_{g_1(i,j)}$ and $W_{g_2(i,j)}$, for i, j with $i \neq j$, along with the definitions of $S_{1,i,j}$, $R_{1,i,j}$, $S_{2,i,j}$, $R_{2,i,j}$

i. $W_{g_1(i,j)}$ starts enumerating $W_{h(i)}$ until it observes that the current values of $F(g_1(i,j))$ and of $F(h(i))$ are the same and not in $\{F^1(h(j)), ?\}$.

   Then, suppose that $W_{g_1(i,j)}$ enumerated up to now is $S_{1,i,j}$.

   For defining g_2, correspondingly, we use $F^2(h(j))$ and $S_{2,i,j}$.

ii. Wait until $W_{h(j)}$ contains $S_{1,i,j}$.

iii. Enumerate $W_{h(j)}$ until $F(g_1(i,j))$ becomes equal to $F^1(h(j))$. Note that this means $F(g_1(i,j))$ must have output a second value. Let $R_{1,i,j}$ denote the $W_{g_1(i,j)}$ enumerated up to now.

   Note that if the algorithm reaches this step, the above must eventually succeed. Go to Step iv.

   For defining g_2, correspondingly, we use $F^2(h(j))$, $S_{2,i,j}$ and $R_{2,i,j}$.

iv. Now, $W_{g_1(i,j)}$ searches for a $k \neq j$ such that $W_{h(k)}$ contains $R_{1,i,j}$, and then it switches to enumerating $W_{h(k)}$. Similarly, $W_{g_2(i,j)}$ searches for a $k \neq j$ such that $W_{h(k)}$ contains $R_{2,i,j}$, and then it switches to enumerating $W_{h(k)}$.

   Note that the above should never happen (except when g_1's assumption about $F(h(j))$ outputting only one grammar is wrong).

   Thus, $R_{1,i,j}$ (or $R_{2,i,j}$) would be a characteristic sample for $W_{h(j)}$.

Let $g_1(i,j)$ (similarly $g_2(i,j)$), $i \neq j$, be recursive functions defined using the operator recursion theorem (see Case, 1974) such that $W_{g_1(i,j)}$ and $W_{g_2(i,j)}$ are as in Algorithm 1. Note that $g_1(i,j)$ and $g_2(i,j)$ (for i, j with $i \neq j$) are similar, except that they work on grammars $h(j)$ from group 1 or 2, respectively, thus in Algorithm 1, $g_2(i,j)$ would correspondingly start only after $F(h(j))$ has output the second grammar. Intuitively, $g_1(i,j)$ and $g_2(i,j)$ simulate F and along the way define sets $S_{1,i,j}$, $R_{1,i,j}$, $S_{2,i,j}$, $R_{2,i,j}$. These are the sets which the learner looks for in the input language to output its conjectures. It will be argued below that this allows for explanatory learning.

A grammar $h(j)$ starts in group 1 and then later may move to group 2. Within each group, it starts in active list, moves to cancelled list, and then to "correct grammar" list when the characteristic sample as in Algorithm 1 is observed. Note that a grammar which is in cancelled list/correct grammar for group 1 may move to group 2 at some point (where it starts as active grammar).

A grammar $h(j)$ in active list in group 1 (group 2) gets cancelled if there exists $i \neq j$ such that $S_{1,i,j}$ (respectively, $S_{2,i,j}$) gets defined and is contained in the input.

A grammar $h(j)$ in group 1 (group 2) becomes "correct grammar" when the input contains $R_{1,i,j}$ (respectively $R_{2,i,j}$) for some $i \neq j$.

The learner outputs the least grammar which is not in cancelled list (irrespective of which group it belongs to).

Note that if $h(j)$ is the correct grammar for input, then eventually for all $i < j$, $S_{1,j,i}$ (or $S_{2,j,i}$, in case $F(h(i))$ outputs two grammars) would get defined, and thus $h(i)$ would go to cancelled list of that group. It will not move to "correct grammar" list of the eventual group $h(i)$ is in, as otherwise, $g_1(j,i)$ or $g_2(j,i)$ would then be able to follow $h(j)$, forcing F to make three outputs on $g_1(j,i)$ or $g_2(j,i)$ based on which group $h(i)$ belongs to.

If $h(j)$ gets into cancelled list of group 1 (or group 2) due to some $h(i)$, $i \neq j$, (that is $S_{1,i,j}$ or $S_{2,i,j}$ getting defined), then eventually it moves to the correct grammar list in the corresponding group as $g_1(i,j)$ (or $g_2(i,j)$ respectively) would eventually have Step ii and iii succeed as $S_{1,i,j}$ was in the input text and thus in $W_{h(j)}$.

Thus, eventually $h(j)$ is the grammar which is output by the learner. ■

5.6. Proof of Theorem 17

Proof We follow the notation in the proof of Theorem 15. Let $\{W_{h(0)}, W_{h(1)}, \ldots\}$ be a one-one r.e. numbering of a class $\mathcal{L}$ and let F be a 2-shot comparator for $\mathcal{L}$. Given any $i, j \in \mathbb{N}$ with $i \neq j$, define a recursive function g using the operator recursion theorem (see Case, 1974) so that $W_{g(i,j)}$ is as in Algorithm 2.

Algorithm 2: Construction of $W_{g(i,j)}$, for i, j with $i \neq j$, along with definition of $S_{i,j}$ and $R_{i,j}$

i. Enumerate $W_{h(i)}$ into $W_{g(i,j)}$ until $F(g(i,j), h(j)) = no$. Let $S_{i,j}$ be the set of elements enumerated into $W_{g(i,j)}$ up to the point where $F(g(i,j), h(j))$ outputs "no".

ii. Wait until $W_{h(j)}$ contains $S_{i,j}$ (if $S_{i,j} \not\subseteq W_{h(j)}$, then this step does not terminate).

iii. Enumerate $W_{h(j)}$ into $W_{g(i,j)}$ until $F(g(i,j), h(j))$ outputs "yes" (after the above "no"). Let $R_{i,j}$ be the set of elements enumerated into $W_{g(i,j)}$ up to this point.

iv. Wait until $W_{h(k)}$ contains $R_{i,j}$ for some $k \neq j$; $W_{h(k)}$ is then enumerated into $W_{g(i,j)}$.

(Note that this step should never succeed, because otherwise $g(i,j)$ would be an r.e. index for $W_{h(k)}$ such that $F(g(i,j), h(j))$ changes its value at least twice, from "no" to "yes" and then to "no" again.)

An Ex learner M for $\mathcal{L}$ runs Algorithm 2 and does the following: Each index may be assigned one of two possible states, "cancelled" or "correct". An index j is assigned the state "cancelled" if there is an $i \neq j$ such that $S_{i,j}$ is contained in the range of the input text but j is not in state "correct"; it is assigned the state "correct" if, for some $i \neq j$, $R_{i,j}$ is contained in the range of the input text. Furthermore, the state of an index may change at any stage (or it may not be assigned any state at all). At each stage, M outputs the least index j such that j is not in a "cancelled" state (note that in particular, j may not have been assigned any state up to the current stage).

Suppose that M is fed a text T for $W_{h(j)}$. As argued in the proof of Theorem 15, if the index j is assigned the state "cancelled" at any stage, then it will eventually be assigned the state "correct" and its state will henceforth never change. On the other hand, for every $i < j$, i will eventually be

assigned the state "cancelled", since $S_{j,i}$ is contained in the range of the text T. Also, it will never be assigned the state "correct" after being in the state "cancelled", since otherwise there would be some $j' \neq i$ such that Algorithm 2 on input (j', i) goes through Step iv, resulting in $g(j', i)$ becoming an index for $W_{h(k)}$ with $k \neq i$, and $F(g(j', i), h(i))$ changing its value at least twice, a contradiction. ■

5.7. Proof of Theorem 18

Proof For each e, let y_e be the first $y > 3$ found, if any, in some standard search, such that for some σ with $\text{content}(\sigma) = \{e\} \oplus \{x : x \leq y\} \oplus \emptyset$, $W_{M_e(\sigma)} \supset \{e\} \oplus \{x : x \leq 4^y\} \oplus \emptyset$.

Consider the class $\mathcal{L}$ consisting of

(i) $L_e = \{e\} \oplus \mathbb{N} \oplus \emptyset$;

(ii) if y_e is defined then, for z with $y_e < z < 4^{y_e}$, the sets

- $X_{e,z}$, if $C(z) \geq y_e$,
- $Y_{e,z}$, if $C(z) < y_e$,

also belong to $\mathcal{L}$, where

$X_{e,z} = \{e\} \oplus \{x : x \leq y_e\} \cup \{z\} \oplus \emptyset$, and

$Y_{e,z} = \{e\} \oplus \{x : x \leq y_e\} \cup \{z\} \oplus \{2^z(2t+1)\}$, where t is the time needed to find y_e and that $C(z) < y_e$.

Note that $\mathcal{L}$ is an indexed family. Moreover, it is not conservatively learnable as if a learner M_e conservatively learns L_e, then using a locking sequence argument, y_e is defined. Furthermore, for some z, $y_e < z < 4^{y_e}$, $C(z) \geq y_e$. Now, the learner cannot conservatively learn $X_{e,z}$.

Next it is shown that $\mathcal{L}$ is 2-standardisable. Note that if $W_{e'}$ is an index for $X_{e,z}$, where y_e is defined, then $y_e \leq C(z) \leq \log(e')$+constant. Thus, for standardising it can be assumed, without loss of generality, that $y_e \leq e'$, as standardising for the e' where $e' - \log(e')$ is smaller than the constant above can be done by patching.

Now, the standardiser on input e' first searches for an e such that $W_{e'}$ contains $3e$. Then,

- if $W_{e'}$ contains $3(4^{e'}) + 3 + 1$, then the standardiser outputs the canonical index for L_e and never changes its mind after that;
- if $W_{e'}$ contains $3(2^z(2t+1)) + 2$, for some z, t, then it outputs the canonical index for $Y_{e,z}$ and never changes its mind after that;
- if y_e is defined and $W_{e'} = X_{e,z}$ for some z, then it outputs the canonical index for $X_{e,z}$.

It is easy to see that the above standardiser is a 2-shot standardiser and standardises $\mathcal{L}$. ■

5.8. Proof of Theorem 19

Proof The class consists of languages $L_{e,x}$ which are defined as follows:

(i) $2e$ is the unique even element of $L_{e,x}$;

(ii) for each e there is an "event-horizon" $t_{e,s}$ which moves up in stages (that is, $t_{e,s}$ is monotonically non-decreasing in s) and converges to a limit value $t_e \in \mathbb{N} \cup \{\infty\}$;

(iii) if $2x+1 \leq t_e$ or $C(x) < \log(x)$ then $L_{e,x}$ contains besides $2e$ all odd numbers below t_e and all odd numbers $2y+1$ with $C(y) < \log(y)$; and

(iv) if $2x+1 > t_e$ and $C(x) \geq \log(x)$ then $L_{e,x}$ contains $2e$, $2x+1$ and all odd numbers below t_e.

For the ease of notation, let z be a fixed element with $C(z) < \log(z)$. In the case that $t_e = \infty$, for all x, $L_{e,x} = \{e\} \oplus \mathbb{N}$.

Now one defines the movements of the t_e through a diagonalisation of the explanatory learner M_e on a text constructed as the limit of sequences $\sigma_0, \sigma_1, \ldots$; at the beginning, σ_0 is just the single entry $2e$ and $t_{e,0} = 0$. At stage $s > 0$, one assumes that only numbers below s are enumerated into sets $L_{e,x}$. Now one searches for an x with $t_{e,s} \leq x < (s^2-1)/2$, such that M_e makes a mind change somewhere on the way from $M_e(\sigma_s)$ towards $M_e(\sigma_s\,(2x+1)^s)$. If this x is found then one sets $t_{e,s+1} = s^2$ and σ_{s+1} to be the extension of $\sigma_s(2x+1)^s$ which contains all the data $\{2x+1 : 2x+1 < s^2\} \cup \{2e\}$; else one defines $t_{e,s} = t_{e,s-1}$ and $\sigma_s = \sigma_{s-1}$.

If $t_e = \infty$ then this construction produces a text for $L_{e,z}$ on which M_e does not converge and M_e does not learn $L_{e,z}$. If $t_e < \infty$ then M_e learns at most one of the $L_{e,x}$ with $2x+1 > t_e$ and $C(x) \geq \log(x)$, as on each of the texts $\sigma_{t_e}\,(2x+1)^\infty$ for these $L_{e,x}$ the learner converges to the same index $M_e(\sigma_{t_e})$. Thus the class does not have any learner which learns it in the limit. The class is also r.e., since one can take a canonical enumeration of the $L_{e,x}$ with $e, x \in \mathbb{N}$.

Note that if $W_{e'} = L_{e,x} \neq L_{e,z}$, then, $C(x) \leq 2\log(e') + 2\log(e) + \log s + const$, where s is maximal such that $t_{e,s} \neq t_{e,s+1}$. As $2x+1 > t_e \geq s^2$, $\log(t_e) \leq C(x) + 2 \leq 2\log(e') + 2\log(e) + 0.5\log(t_e) + const$, or $0.5\log(t_e) \leq 2\log(e') + 2\log(e) + const$. Thus, one can compute for each index e' with $W_{e'}$ containing a number $2e$, an upper bound $f(e,e')$ such that whenever $t_e > f(e,e')$ then either $W_{e'} = L_{e,z}$ or $W_{e'}$ is not in the class.

To see that the class is 2-shot comparable, consider any indices e', e'' of languages in the class; if the languages are not in the class, then the comparator can do whatever it wants, including remaining undefined. On e', e'', the comparator waits until each of them enumerates an even number; if e', e'' are indices for languages in the class, then these must show up and be unique. If these even numbers are different, the comparator just guesses "no" and does not change the mind. If they are the same number $2e$ then the comparator simulates $W_{e'}$ and $W_{e''}$ until a stage $s > f(e,e') + f(e,e'')$ is reached such either $t_{e,s} > f(e,e') + f(e,e'')$ or x', x'' have been found such that $t_{e,s} < (2x'+1)$, $t_{e,s} < (2x''+1)$, $2x'+1 \in W_{e'}$ and $2x''+1 \in W_{e''}$. In the case that $t_{e,s} > f(e,e') + f(e,e'')$, the comparator conjectures "yes", as both sets are equal to $L_{e,z}$. In the case that x', x'' are found, the comparator checks whether $x' = x''$. Now if $x' = x''$ then the comparator conjectures "yes", else the comparator conjectures "no"; furthermore, the comparator makes in the latter case a mind change from "no" to "yes" iff either $t_{e,\infty} > f(e,e') + f(e,e'')$ or both $C(x') < \log(x')$ and $C(x'') < \log(x'')$. These conditions imply that both sets are equal to $L_{e,z}$; for the verification also note that t_e is either $t_{e,s}$ or larger than $f(e,e') + f(e,e'')$, so that whenever $t_{e,s}$ becomes updated to a larger number, this is immediately above $f(e,e') + f(e,e'')$. ■

5.9. Proof of Theorem 20

Proof Let $\mathcal{L} = \{W_{f(0)}, W_{f(1)}, W_{f(2)}, \ldots\}$ be a uniformly r.e. class that has a 2-shot verifier h, where f is a given recursive function. By Gao et al. (2016, Observation 35), it suffices to show that $\mathcal{L}$ has an Ex learner N such that for every $L \in \mathcal{L}$, N does not conjecture a superset of L on any segment of any given text for L.

Given a text T, define a learner M as follows: on input $T[n]$, M outputs $f(i)$ for the least $i \leq n$ such that h outputs "yes" on $(f(i), T[n])$; if no such i exists, then M just outputs its prior conjecture (or an r.e. index for the empty set if $n = 0$). Suppose T is a text for some $W_{f(e)} \in \mathcal{L}$. Let ℓ be the least index such that $W_{f(\ell)} = W_{f(e)}$. Then for all sufficiently large m, h will output "no" on pairs $(f(\ell'), T[m])$ for each $\ell' < \ell$, while h will output "yes" on $(f(\ell), T[m])$. As a 2-shot verifiable class is also 2-shot comparable and thus inclusion-free by Proposition 9, M will never output a proper superset on input of languages from the class. ■

5.10. Proof of Theorem 22

Proof By Proposition 9 we know that any class which is 2-shot comparable must be inclusion-free. Let $e_0, e_1, \ldots$ be any fixed enumeration of the class $\mathcal{L}$. Consequently, the algorithm, which has access to the K-oracle always takes the enumeration $e_0, e_1, \ldots$ of the indices of the class $\mathcal{L}$ and conjectures the first e_n such that either n is the number of data items seen so far or W_{e_n} contains all the data observed. The first case is only included to avoid partialness of the K-recursive learner.

In order to see that the learner is conservative we use that $\mathcal{L}$ is inclusion-free. Hence, on each wrong hypothesis e_n the learner will eventually see a counterexample, that is, an element outside W_{e_n} and drop it eventually. ■

5.11. Proof of Example 23

Proof A $(2n+1)$-shot comparator F works as follows: on input (i, j), F outputs "yes" at the s^{th} computational step if $W_{i,s} = W_{j,s}$ and "no" otherwise. Then if $|W_i| \leq n$ and $|W_j| \leq n$, F changes its value at most $2n$ times and its last value is correct, i.e., F is a $(2n+1)$-shot comparator.

To see that FINS_n is not $2n$-shot comparable, by way of a contradiction let us assume that such a $2n$-shot comparator G does exist. In particular, the comparator G on any input (i, j) such that $|W_i| \leq n$ and $|W_j| \leq n$ changes its value at most $2n - 1$ times. By the recursion theorem (see Rogers, 1987, chap. 11), there are r.e. sets W_i and W_j for which either (1) $W_i = W_j = \emptyset$ and G never outputs "yes" on input (i, j), or (2) there exists a least $m \leq n$ such that $W_i = \{x : 0 \leq x \leq m - 1\}$ and $W_j = W_i \setminus \{m - 1\}$, and G on input (i, j) does not output "no" in the limit, or (3) there is a least $m' \leq n$ such that $W_i = W_j = \{x : 0 \leq x \leq m' - 1\}$ and G on input (i, j) does not output "yes" in the limit. This contradicts the fact that G is a $2n$-shot comparator for FINS_n. ■

5.12. Proof of Lemma 25

Proof Let $\mathcal{L}$ be as given in the lemma. Let $h(i, t)$ be a recursive function such that $\lim_{t\to\infty} h(i, t) = 1$ iff for all $j < i$, $W_{f(j)} \neq W_{f(i)}$. Note that such a recursive h can be easily constructed. Also note that in case $W_{f(j)} = W_{f(i)}$ for some $j < i$, then $\lim_{t\to\infty} h(i, t)$ may not exist. Let $U_{i,t} = W_{f(i)}$ if t is minimal such that $h(i, t') = 1$ for all $t' \geq t$. Otherwise, $U_{i,t} = \{\langle 1, x\rangle : x < \langle i, t, z\rangle\} \cup$

$\{\langle 0, \langle i, t, z\rangle\rangle\} \cup \{\langle 2, x\rangle : x \in \mathbb{N}\}$, where z is the least $t' \geq t$ such that $h(i, t') = 0$ (the element $\langle 0, \langle i, t, z\rangle\rangle$ ensures that $U_{i,t} \neq U_{j,s}$ whenever $\langle i, t\rangle \neq \langle j, s\rangle$). Clearly, $\mathcal{L}' = \{U_{i,t} : i, t \in \mathbb{N}\}$ is a 1–1 r.e. class which is $(n+1)$-shot comparable. The comparator essentially uses the comparator for $\{W_{f(i)} : i \in \mathbb{N}\}$ until it finds that one of the languages contains $\langle 0, y\rangle$, for some y. Then it waits for either $\langle 0, y'\rangle$ or some element $\langle 1, z\rangle$ with $z \geq y$, to appear in the other language, and then outputs the correct comparison as "yes" iff $\langle 0, y'\rangle$ appears in the other language with $y' = y$. ■

5.13. Proof of Proposition 27

Proof Assume that F is a 3-shot comparator for $\mathcal{L} = \{U_0, U_1, \ldots\}$.

Let g, h be recursive functions, obtained via the operator recursion theorem (see Case, 1974), such that $W_{g(e,d)}$ and $W_{h(e,d)}$ are defined as follows for $e \neq d$:

1. $W_{g(e,d)}$ and $W_{h(e,d)}$ follow U_d, until it is observed that $F(g(e,d), h(e,d)) =$ "yes". Assume that $W_{h(e,d)}$, enumerated until now, is $S_{e,d}$.

2. $W_{g(e,d)}$ continues to follow U_d and $W_{h(e,d)}$ waits until it is observed that $S_{e,d} \subseteq U_e$. Then, $W_{h(e,d)}$ starts following U_e until it is observed that $F(g(e,d), h(e,d)) =$ "no" (after the above observed "yes"). Assume that $W_{g(e,d)}$ enumerated until now is $R_{e,d}$.

3. $W_{h(e,d)}$ continues to follow U_e and $W_{g(e,d)}$ waits until it is observed that $R_{e,d} \subseteq U_e$. Then, $W_{g(e,d)}$ starts following U_e until it is observed that $F(g(e,d), h(e,d)) =$ "yes" (after the above observed "no"). Assume that $W_{g(e,d)}$ enumerated until now is $X_{e,d}$.

4. $W_{h(e,d)}$ continues to follow U_e and $W_{g(e,d)}$ waits until it is observed that $X_{e,d} \subseteq U_k$, for some $k \neq e$. Then, $W_{g(e,d)}$ starts following U_k. Note that the above search should never succeed, as otherwise, $F(h(e,d), g(e,d))$ needs to output "no" in the limit, but it has no more mind changes available.

   Thus, $X_{e,d}$ (if defined) is a subset of U_e but not contained in any U_k, where $k \neq e$.

Note also that if $U_d \subseteq U_e$, then the above process must reach Step 4, though it would not succeed in finding k. This happens as $S_{e,d}$ is always defined due to the comparator $F(g(e,d), h(e,d))$ needing to eventually output "yes" when both $g(e,d)$ and $h(e,d)$ are following U_d. Since we have $S_{e,d} \subseteq U_d \subseteq U_e$, Step 2 will eventually succeed in finding this, and thus $W_{h(e,d)}$ would start following U_e. Hence eventually $F(g(e,d), h(e,d))$ needs to output "no" and the procedure will reach Step 3. Here again $R_{e,d} \subseteq U_d \subseteq U_e$, and thus $W_{g(e,d)}$ would also start following U_e and thus eventually $F(g(e,d), h(e,d))$ will output "yes", and the process will reach Step 4.

Thus, we have the following:

(a) if U_e has no proper subset in $\mathcal{L}$, then $\emptyset$ is a tell-tale set for U_e;

(b) if U_e has a proper subset in $\mathcal{L}$, then there exists a d such that $X_{e,d}$ as above gets defined;

(c) for any e, d, if $X_{e,d}$ get defined then $X_{e,d}$ is a tell-tale set for U_e with respect to $\mathcal{L}$.

It thus follows that one can enumerate tell-tale sets for members of $\mathcal{L}$: on input e, the tell-tale set enumerator initially just enumerates $\emptyset$, and searches for a d such that $X_{e,d}$ is defined by the above process. It then enumerates $X_{e,d}$. ■

5.14. Proof of Example 30

Proof The non-learnability follows from the fact that the set $\mathbb{N}$ does not have any tell-tale set with respect to this class (see Angluin, 1980; Baliga et al., 1999). This also implies that the class is not verifiable. The positive results are obtained by the following fact: From an index e of $\mathbb{N} \setminus \{x\}$ and an upper bound on x, one can compute x by enumerating the set W_e until all elements below the upper bound but x have shown up, as the upper bounds can have very small Kolmogorov complexity compared to x, this means that all sufficiently large x satisfy that either $W_e \neq \mathbb{N} \setminus \{x\}$ or $C(x) < \log(x)$. Hence there exists a recursive function f such that whenever W_e is of the form $\mathbb{N} \setminus \{x\}$ with $C(x) \geq \log(x)$ then we have $x < f(e)$. Thus a 2-shot standardiser would first enumerate the elements of W_e below $f(e)$ until all but one x have shown up. Then the learner outputs an index $g(x)$ computed from x for the set $\mathbb{N} \setminus \{x\}$. Once this element x is also enumerated into W_e, the standardiser revises the hypothesis and outputs a fixed index for $\mathbb{N}$.

However, the class is not finitely standardisable, as it is not inclusion-free; for the same reason it can also not be 2-shot comparable. The 3-shot comparability follows from the general implication that every 2-shot standardisable class is 3-shot comparable. ■

5.15. Proof of Proposition 31

Proof Suppose $\mathcal{L}$ has an n-shot learner M. Without loss of generality assume that M does not use more than n-shots on any text, even for texts for languages outside $\mathcal{L}$. On input (d, e), a $2n$-shot comparator F for $\mathcal{L}$ builds $\sigma_1, \sigma_2, \ldots, \sigma_n$ as follows (some of these σ_i may not be defined). Initially, it searches for σ_1 with $\text{content}(\sigma_1) \subseteq W_d \cup W_e$ such that $M(\sigma_1) \neq ?$. Then, inductively, after defining σ_i, it searches for a σ_{i+1} such that σ_{i+1} is an extension of σ_i, $\text{content}(\sigma_{i+1}) \subseteq W_d \cup W_e$ and $M(\sigma_i) \neq M(\sigma_{i+1}) \neq ?$. The comparator F outputs ? until σ_1 gets defined. After that, at any stage s, it considers the last σ_i that is defined, and outputs "yes" iff $\sigma_i \subseteq W_{d,s} \cap W_{e,s}$; otherwise, it outputs "no". As the learner M is a n-shot learner, the comparator is a $2n$-shot comparator, as it possibly starts with "no" output and then perhaps changes to "yes" output for each σ_i. Now assume both W_d, W_e are in $\mathcal{L}$. If $W_d = W_e$, then clearly all σ_i will satisfy that $\text{content}(\sigma_i) \subseteq W_d \cap W_e$ and thus the comparator will converge to "yes". If $W_d \neq W_e$, then for the last σ_i that gets defined, $\text{content}(\sigma_i) \not\subseteq W_d \cap W_e$, as otherwise, the learner converges on texts for W_d or W_e which extend σ_i, to the same conjecture, contradicting that M explanatorily learns both. ■

5.16. Proof of Theorem 32

Proof This can be seen as follows: For all $e, s \in \mathbb{N}$, we define $L_{e,0} := \{e\} \oplus \mathbb{N}$, and for $s > 0$

$$L_{e,s} := \begin{cases} \{e\} \oplus D\,, & \text{if there is a first step } t \leq s \text{ at which some } \sigma \in \mathbb{N}^* \text{ is found such that there} \\ & \text{is a } D \subseteq \mathbb{N} \text{ with } \text{content}(\sigma) = \{e\} \oplus D \text{ and } W_{M_{e,t}(\sigma)} \supset \text{content}(\sigma)\,; \\ \{e\} \oplus \mathbb{N}\,, & \text{otherwise.} \end{cases}$$

Here, we assume without loss of generality that at any step, at most one σ is found in the first clause above. We set $\mathcal{L} := \{L_{e,s} : e, s \in \mathbb{N}\}$ and note that $\mathcal{L}$ has a uniformly recursive numbering. Furthermore, if M_e were a behaviourally correct learner of $\mathcal{L}$, then, since M_e must learn $\{e\} \oplus \mathbb{N} \in \mathcal{L}$, there must exist some $\sigma \in \mathbb{N}^*$ and $D \subset \mathbb{N}$ with $\text{content}(\sigma) = \{e\} \oplus D$ such that M_e overgeneralises on input σ. Hence $\mathcal{L}$ does not have a conservative behaviourally correct learner.

On the other hand, the class $\mathcal{L}$ is 3-shot comparable. Given indices i and j, a 3-shot comparator F simulates W_i and W_j, and it outputs ? until W_i and W_j each enumerates an even element. If the first even element enumerated by W_i is different to that enumerated by W_j, then F outputs "no". Suppose the first even element enumerated by both W_i and W_j is $2e$. Then F outputs "yes" until it finds an s such that for some finite set $D \subset \mathbb{N}$, $L_{e,s} = \{e\} \oplus D$ and exactly one of the sets W_i, W_j enumerates some $2y+1 \notin \{2x+1 : x \in D\}$; it will then output "no" until W_i and W_j each enumerates some odd number not belonging to $\{2x+1 : x \in D\}$; $F(i,j)$ will now output "yes". Note that if $W_i, W_j \in \mathcal{L}$, then $F(i,j)$ will change its value at most twice; moreover, $F(i,j)$ will converge to the correct value.

Similar proof as above also shows that $\mathcal{L}$ is 3-shot verifiable. ■

5.17. Proof of Theorem 33

Proof Let $\mathcal{L} = \{L_0, L_1, L_2, \ldots, \}$ be a class of r.e. sets such that

$$L_0 := \{2y : y \in \mathbb{N}\} \text{ and for all } x \in \mathbb{N}\,,$$

$$L_{x+1} := \begin{cases} (\{2y : y \in \mathbb{N} \wedge y \leq t\} \setminus \{2x\}) \cup \{2t+1\}\,, & \text{if } t \text{ is the first step at which some} \\ & p < \log(x) \text{ is found such that} \\ & U(p,\varepsilon) = x\,; \\ \{2y : y \in \mathbb{N} \wedge y \neq x\}\,, & \text{if no such } p \text{ is found}\,. \end{cases}$$

Note that $\mathcal{L}$ has a uniformly recursive numbering. Furthermore, one can define a 4-shot comparator F for $\mathcal{L}$ as follows: First, as in Proposition 30, let f be a recursive function such that whenever $W_e = \{2y : y \in \mathbb{N} \wedge y \neq x\}$ for some x with $C(x) \geq \log(x)$, $x < f(e)$. Given $d, e \in \mathbb{N}$, simulate W_d and W_e. At every step, F performs the instructions in the case (among the four cases below) with the highest priority that applies; Case i has higher priority than Case j iff $i < j$.

Case 1: One of the sets, say W_d, enumerates an odd number $2y+1$. Simulate W_d until all even numbers but one (say $2x$) below $2y+1$ appear in W_d. Simulate W_e until the first of the following cases applies:

- **Case 1.1:** Either $2x$ or some even number larger than $2y$ appears in W_e, or an odd number different from $2y+1$ occurs in W_e. Output "no".
- **Case 1.2:** The set W_e enumerates all elements of $\{2z : 0 \leq z \leq y \wedge z \neq x\} \cup \{2y+1\}$. Output "yes".

Case 2: All the even numbers below $f(d)+f(e)$ have been enumerated into both W_d and W_e. Output "yes".

Case 3: All the even numbers below $f(d)+f(e)$ have been enumerated into exactly one of W_d and W_e. Output "no".

Case 4: There is exactly one even number $x_1 < f(d)+f(e)$ that has not been enumerated into W_d, and there is exactly one even number $x_2 < f(d)+f(e)$ that has not been enumerated into W_e. If $x_1 = x_2$, output "yes". If $x_1 \neq x_2$, output "no".

Now it is verified that F is indeed a 4-shot comparator for $\mathcal{L}$. Suppose $W_d, W_e \in \mathcal{L}$. Consider the following case distinction:

Case i: At least one of W_d, W_e contains an odd number. Then Case 1 will almost always apply, and either (a) both W_d and W_e are equal to $\{2z : 0 \leq z \leq y \wedge z \neq x\} \cup \{2y+1\}$ for some $y, x \in \mathbb{N}$, so that F will output "yes" in the limit, or (ii) one of W_d, W_e is equal to $\{2z : 0 \leq z \leq y \wedge z \neq x\} \cup \{2y+1\}$ for some $y, x \in \mathbb{N}$ while the other contains $2x$ or some odd number different from $2y+1$ or an even number larger than $2y$, so that F will output "no" in the limit.

Case ii: $W_d = \{2z : z \in \mathbb{N}\} \setminus \{2x_1\}$ and $W_e = \{2z : z \in \mathbb{N}\} \setminus \{2x_2\}$ for some $x_1, x_2 \in \mathbb{N}$. Then Case 4 will almost always apply, and if $x_1 = x_2$, then F will output "yes" in the limit; if $x_1 \neq x_2$, then F will output "no" in the limit.

Case iii: Both sets W_d and W_e contain only even numbers, and at least one of W_d, W_e is equal to $\{2z : z \in \mathbb{N}\}$. If both W_d and W_e are equal to $\{2z : z \in \mathbb{N}\}$, then Case 2 will almost always apply, so that F will output "yes" in the limit. If exactly one of W_d and W_e is equal to $\{2z : z \in \mathbb{N}\}$, then Case 3 will almost always apply, so that F will output "no" in the limit.

Furthermore, note that $F(d, e)$ changes its value between Steps t_1 and t_2 (where $t_2 > t_1$) only if there are distinct i, j with $j < i$ such that F performs the instructions in Case i at Step t_1' and then performs the instructions in Case j at Step t_2' for some t_1', t_2' with $t_1 \leq t_1' < t_2' \leq t_2$; in particular, if Case 1 applies at some step, then $F(d, e)$ will not change its value at any subsequent step. Thus F changes its value at most thrice on input (d, e), and it is therefore a 4-shot comparator for $\mathcal{L}$.

A 3-shot standardiser G for $\mathcal{L}$ can be defined similarly. Given any index e, G outputs ? until the first of the following cases applies:

Case a: The set W_e enumerates all even numbers but one (say $2x_1$) below $f(e)$. Then G keeps outputting a canonical index for $\{2z : z \in \mathbb{N} \wedge z \neq x_1\}$. If W_e enumerates $2x_1$ at any later step, then G switches to outputting a canonical index for $\{2z : z \in \mathbb{N}\}$. If W_e enumerates an odd number at any step, then G follows the instructions in Case b.

Case b: The set W_e enumerates an odd number $2y+1$. Then G waits until W_e enumerates all even numbers but one (say $2x_2$) below $2y+1$; it will then output a canonical index for $\{2z : z \leq y \wedge z \neq x_2\} \cup \{2y+1\}$ in the limit.

That G is indeed a 3-shot standardiser for $\mathcal{L}$ can be verified using ideas very similar to those in the earlier proof that F is a 4-shot comparator for $\mathcal{L}$.

Moreover, $\mathcal{L}$ is not behaviourally correctly learnable because L_0 does not have a finite tell-tale (see Baliga et al., 1999, Corollary 3): for every set D of even numbers, there is some $x > \max(D)$ with $C(x) \geq \log(x)$, so that $D \subset L_{x+1} \subset L_0$. ■