Optimal Teaching Curricula with Compositional Simplicity Priors

Manuel Garcia-Piqueras (✉)[1][0000−0001−8088−8393] and José Hernández-Orallo[2][0000−0001−9746−7632]

[1] Math. Dept., Universidad de Castilla-La Mancha, Albacete, Spain
manuel.gpiqueras@uclm.es
[2] VRAIN, Universitat Politècnica de València, València, Spain
jorallo@upv.es

Abstract. Machine teaching under strong simplicity priors can teach any concept in universal languages. Remarkably, recent experiments suggest that the teaching sets are shorter than the concept description itself. This raises many important questions about the complexity of concepts and their teaching size, especially when concepts are taught incrementally. In this paper we put a bound to these surprising experimental findings and reconnect *teaching size* and concept complexity: complex concepts do require large teaching sets. Also, we analyse teaching curricula, and find a new *interposition* phenomenon: the teaching size of a concept can increase because examples are *captured* by simpler concepts built on previously acquired knowledge. We provide a procedure that not only avoids interposition but builds an *optimal curriculum*. These results indicate novel curriculum design strategies for humans and machines.

Keywords: Machine teaching · Interposition · Kolmogorov complexity.

1 Introduction

A *teacher* instructing a series of concepts to a *learner* using examples would ideally design a curriculum such that the whole teaching session is shortest. For one concept, the field of *machine teaching* has analysed the efficiency of the teacher, the learner or both, for different representation languages and teaching settings [42,5,16,28,37]. For more than one concept, however, we need to consider different sequences of concepts, or *curricula*, to make learning more effective. While there has been extensive experimental work in curriculum learning [36], the theoretical analysis is not abundant and limited to continuous models [26,12,40]. It is not well understood how curriculum learning can be optimised when concepts are *compositional*, with the underlying representation mechanisms being rich languages, even Turing-complete. Also, in a curriculum learning situation where a *teacher* chooses the examples sequentially, it is surprising that the connection with machine teaching has not been made explicit at a general conceptual level, with only a specific minimax approach for gradient-based representations [41,10,11]. In other words, to our knowledge, a theoretical framework has not

yet been articulated for curriculum learning in machine teaching, or *curriculum teaching*, when dealing with universal languages, as a counterpart to incremental inductive inference based on simplicity [34,35].

While the teaching dimension has been the traditional metric for determining how easy it is to teach a concept [42], the *teaching size* [38] is a new metric that is more reasonably related to how easy it is to teach an infinite compositional concept class. It is also more appropriate to understand 'prompting' of language models as a kind of teaching, where users need to think of the shortest prompts that make a language model such as BERT, GPT-2 or GPT-3 achieve a task by few-shot learning [6,27,4]. However, as far as we know, the following issues are not clear yet: (1) *What is the relationship between the Kolmogorov complexity of a concept and how difficult it is to be taught under the teaching size paradigm?* and (2) *Is there a way to extend machine teaching, and teaching size in particular, to consider the notion of* optimal teaching curricula*?*

Theorem 1 addresses the first question and shows that concepts with *high* complexity are *difficult to teach*, putting a limit to the surprising experimental finding recently reported in [38], where teaching a concept by examples was usually more economical (in total number of bits) than showing the shortest program for the concept. This connection suggests that the second question may rely on a strong relation between incremental learning using simplicity priors and curriculum teaching. For instance, consider the concepts c_+ for addition, $c_\times$ for multiplication, $c_\wedge$ for exponentiation and $c_{\not 0}$ for the removal of zeros (Fig. 1). If the concept of c_+ is useful to allow for a shorter description of $c_\times$, is it also reasonable to expect that c_+ would also be useful to *teach* $c_\times$ from examples? Or even $c_\wedge$? In general, is the conditional algorithmic complexity $K(c_2|c_1)$ related to the minimal size of the examples needed to teach c_2 after having acquired c_1?

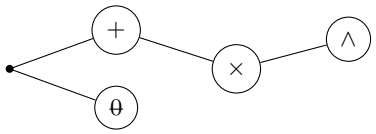

Fig. 1. Curriculum teaching for a set of concepts.

Our perspective studies the sequence of learning a set of concepts, instead of learning a sequence of instances under the same concept. In the general case, we define a teaching curriculum as a set of partial alternative sequences, such as the top and bottom branches in Fig. 1. The order between branches is irrelevant, but the order of concepts inside each branch is crucial. This tree structure is proposed as future work in [26]. Given a set of concepts, is there a curriculum that minimises the overall teaching size?

Our second group of contributions turns around this new concept of teaching curriculum. We provide a definition of *conditional teaching size*, given some other concepts already taught, $TS(c|c_1, \ldots, c_n)$. We show that, in general, $K(c_1|c_2) < K(c_2|c_1)$, for conditional Kolmogorov complexities, does not imply $TS(c_1|c_2) < TS(c_2|c_1)$, and vice versa. Furthermore, given a concept c, it is not true that $TS(c|B) \leq TS(c)$, $\forall B$. We find a new *interposition* phenomenon: acquired con-

cepts may increase the teaching size of new concepts. We give conditions to avoid or provoke interposition. Theorems 3 and 4 are key results in this direction, providing an explicit range where interposition might happen. Finally, we present an effective procedure, $\mathbb{I}$-*search*, to design *optimal* curricula, minimising overall teaching size, for a given set of concepts.

2 Notation and background

Let us consider a machine M and a universal (i.e., Turing complete) language L. We assume that L is formed by a finite set of instructions in an alphabet Υ, each of them been coded with the same number of bits. Hence, each program p in language L can simply be represented as a string in $\Sigma = \{0,1\}^*$, whose length is denoted by $\dot{\ell}(p)$ (in number of instructions) and denoted by $\ell(p)$ (in bits). There is a total order, $\prec$, over programs in language L defined by two criteria: (i) length and (ii) lexicographic order over Υ only applied when two programs have equal size. Programs map binary strings in Σ to $\Sigma \cup \bot$, denoted by $p(\mathtt{i}) = \mathtt{o}$, with $p(\mathtt{i}) = \bot$ representing that p does not halt for $\mathtt{i}$. Two programs are equivalent if they compute the same function.

We say that c is an L-concept if it is a total or partial function $c : \Sigma \to \Sigma \cup \bot$ computed by at least a program in language L. The class of concepts defined by all programs in L is denoted by C_L; $[p]_L$ denotes the equivalence class of program p. Given $c \in C_L$, we denote $[c]_L$ as the equivalence class of programs in L that compute the function defined by c. Examples are just pairs of strings, and their space is the infinite set $X = \{\langle \mathtt{i}, \mathtt{o}\rangle : \langle \mathtt{i}, \mathtt{o}\rangle \in \Sigma \times (\Sigma \cup \bot)\}$. A *witness* can be any finite example subset of X, of the form $S = \{\langle \mathtt{i}_1, \mathtt{o}_1\rangle, \ldots, \langle \mathtt{i}_k, \mathtt{o}_k\rangle\}$. In order to calculate the *size* of these sets, we consider self-delimiting codes. Let δ be the number of bits needed to encode S, using certain prefix code. For instance, if we consider Elias coding [7], the string 01010010001001000101 (size = 20) expresses the example set $\{\langle 1, 010\rangle, \langle 0, 1\rangle\}$ unambiguously. The size of an example set is the size of its encoding (e.g., $\delta(\{\langle 1, 010\rangle, \langle 0, 1\rangle\} = 20$ in Elias coding). For output strings, the natural number to be encoded is increased by 1, to accommodate for $\bot$. We also define a total order $\lessdot$ on X, i.e., $\forall S, S'$ such that $S \lessdot S'$ then $\delta(S) \leq \delta(S')$ with any preference (e.g., lexicographic) for equal size.

A concept c defines a unique subset of the example space X and we call any element in that subset a *positive* example. A concept c satisfies example set S, denoted by $c \vDash S$, if S is a subset of the positive examples of c. For instance, a witness set for the concept c_{θ} (*remove zeros*) is $\{\langle 10011, 111\rangle, \langle 001, 1\rangle\}$. Example sets cannot have different outputs for equal inputs: $\{\langle 1, 00\rangle, \langle 1, 01\rangle\}$ is not valid.

A program p is compatible with $S = \{\langle \mathtt{i}_j, \mathtt{o}_j\rangle\}_{j=1}^{k} \subset X$, denoted by $p \vDash S$, if $p_S(\mathtt{i}_j) = \mathtt{o}_j$ for every $j \in \{1, \ldots, k\}$. For a finite example set S, there is always a program, denoted by $\ddot{p_S}$, that implements a conditional hard-coded structure of if-then-elses (trie) specifically designed for S. If we know the number of bits of input $\mathtt{i}$ and the set of examples in S, the number of comparisons using a trie-data structure is linearly time-bounded. Namely, for any $\ddot{p_S}$, there exists a constant, ρ, such that $\rho \cdot \min\{\ell(\mathtt{i}), \ell(\mathtt{i}_{max})\} + \ell(\mathtt{o}_{max})$ is an upper bound of time steps for

each input $\mathtt{i}$, where $\ell(\mathtt{i}_{max})$, $\ell(\mathtt{o}_{max})$ are the lengths of the longest input string and output string in S, respectively. In general, for any program that employs a trie-data structure for S, there exists a time-bound linear function, denoted by $\lambda_L(\mathtt{i}, S)$, that represents an upper bound in time steps on every input $\mathtt{i}$.

Complexity functions $\mathtt{f} : \mathbb{N} \to \mathbb{N}$ act as time bounds. We say that a program p is $\mathtt{f}$-compatible with the example set $S = \{\langle \mathtt{i}_j, \mathtt{o}_j \rangle\}_{j=1}^{k} \subset X$, denoted by $p \vDash_{\mathtt{f}} S$, if $p(\mathtt{i}_j) = \mathtt{o}_j$ within $\max\{\mathtt{f}(\ell(\mathtt{i}_j)), \lambda_L(\mathtt{i}_j, S)\}$ time steps (time-bound) for each $j \in \{1, \ldots, k\}$. In other words, within time bound, for each pair $\langle \mathtt{i}, \mathtt{o} \rangle \in S$ the program p on input $\mathtt{i}$: (1) outputs $\mathtt{o}$ when $\mathtt{o} \neq \perp$ or (2) does not halt when $\mathtt{o} = \perp$. Note that: (i) For any complexity function $\mathtt{f}$ and any example set S, there is always[3], a program $\mathtt{f}$-compatible with S, (ii) there may be programs p such that $p \nvDash_{\mathtt{f}} S \wedge p \vDash S$, if $\mathtt{f}$ and S do not guarantee enough time bound and (iii) larger complexity functions distinguish more programs.

3 Absolute teaching size and complexity

Now we can study how a non-incremental teacher-learner setting works and the relationship between teaching size and Kolmogorov complexity.

Following the K-dimension [2,3], seen as preference-based teaching using simplicity priors [8,15], we assume that the learner is determined to find the shortest program (according to the prior $\prec$). Namely, the learner Φ returns the first program, in order $\prec$, for an example set S and a complexity function $\mathtt{f}$ as follows:

$$\Phi^{\mathtt{f}}_{\ell}(S) = \arg\min_{p}{}^{\prec} \{\ell(p) : p \vDash_{\mathtt{f}} S\}$$

Note that the $\mathtt{f}$-bounded Kolmogorov complexity of an example set S, $K^{\mathtt{f}}(S)$, is the length of the program returned by the learner $K^{\mathtt{f}}(S) = \ell(\Phi^{\mathtt{f}}_{\ell}(S))$. We say that S is a *witness set* of concept c for learner Φ if S is a finite example set such that $p = \Phi^{\mathtt{f}}_{\ell}(S)$ and $p \in [c]_L$.

The teacher selects the *simplest* witness set that allows the learner to identify the concept, according to set size (δ) and associated total order $\preccurlyeq$, as follows:

$$\Omega^{\mathtt{f}}_{\ell}(c) = \arg\min_{S}{}^{\preccurlyeq} \left\{\delta(S) : \Phi^{\mathtt{f}}_{\ell}(S) \in [c]_L)\right\}$$

The $K^{\mathtt{f}}$-teaching size of a concept c is $TS^{\mathtt{f}}_{\ell}(c) = \delta(\Omega^{\mathtt{f}}_{\ell}(c))$.

Every program the teacher picks defines a concept c. The teacher-learner protocol is computable for any complexity function $\mathtt{f}$ and able to create pairs (p_c, w_c), where p_c defines a concept c and w_c is a witness set of c. We can think of these pairs as if they were inserted sequentially in the so-called $\mathtt{f}$-*Teaching Book* ordered by w_c, with no repeated programs or witness sets. For example, if we consider the concept $a \in C_L$ for swapping ones and zeros in a binary string, there will be a pair (p_a, w_a) in the $\mathtt{f}$-Teaching Book, e.g., containing a witness set like $w_a = \{\langle 10, 01 \rangle, \langle 110, 001 \rangle\}$ that the teacher would provide with which the learner

[3] Note that this $\ddot{p}_S$ is ensured by the max with time costs.

would output p_a, a program that swaps 1 and 0. Theorem 1 in [38] shows that *for any concept $c \in C_L$, there exists a complexity function* f *such that there is a pair (p_c, w_c) in the* f*-Teaching Book*. The teaching size makes more sense than the traditional teaching dimension (the smallest cardinality of a witness set for the concept) because some concepts could be taught by very few examples, but some of them could be extremely large. Also, the use of size instead of cardinality allows us to connect teaching size and Kolmogorov complexity, as we do next.

Our first result[4] shows an equipoise between teaching size and data compression, an extra support for machine teaching; the compressing performance of the learner and the minimisation of the teaching size go in parallel.

Proposition 1. Let f be a complexity function and Φ^{f}_{ℓ} the learner. There exist two constants $k_1, k_2 \in \mathbb{N}$, such that for any given pair $(w, p) \in \mathsf{f}$-Teaching Book we have that:[5]

$$K(p) \leq \delta(w) + k_1 \text{ and } K(w) \leq \ell(p) + k_2 \tag{1}$$

Proposition 1 is a key result ensuring that the size difference between programs and witness sets is bounded: a short witness set would not correspond with an arbitrarily complex concept and vice versa. This puts a limit to the surprising empirical observation in [38], where the size of the witness sets in bits was usually smaller than the size of the shortest program for that set, i.e., in terms of information it was usually cheaper to teach by example than sending the shortest description for a concept.

There is another close relationship between the Kolmogorov complexity of a concept and its teaching size. First we need to define the complexity of a concept through the *first program* of a concept in language L.

$$p^*_c = \arg\min_p{}^{\prec} \{\ell(p) : p \in [c]_L\}$$

For every concept $c \in C_L$, we will simply refer to the Kolmogorov complexity of a concept c with respect to the universal language L as $K_L(c) = \ell(p^*_c)$. Now,

Theorem 1. Let L be a universal language, M be a universal machine and k_M be a constant that denotes the length of a program for Φ in M.[6] For any concept $c \in C_L$, there exists a complexity function f, such that $K_L(c) \leq TS^{\mathsf{f}}_{\ell}(c) + k_M$.

This gives an upper bound (the teaching size) for the Kolmogorov complexity of a concept. On the other hand, this theorem implies that concepts with *high* complexity are *difficult to teach* in this setting. The surprising observation found in [38] of some concepts having shorter TS than K has a limit.

[4] The proofs can be found in [9]

[5] We use the standard definition of K using a monotone universal machine U [19] (we will drop U when the result is valid for any U), applied to binary strings (where programs and example sets are encoded as explained in the previous section). With K^f we refer to a non-universal version where the descriptional machine is the learner.

[6] For any universal Turing machine M, a finite program can be built coding an interpreter for Φ in M and taking w_c as input. The length of this 'glued' program does not depend on the concept c but on the machine M to glue things together and how many bits of the program instructions are required to code Φ, i.e., $K_M(\Phi)$.

4 Conditional teaching size

In this section we introduce the notion of conditional teaching size and the *curriculum teaching problem*. We now assume that the learner can reuse any already learnt concept to *compose* other concepts. The curriculum teaching problem is to determine the optimal *sequential* way of teaching a set of concepts $Q = \{c_1, c_2, \ldots, c_n\}$, in terms of minimum total teaching size. Let $TS(c_i|c_j, c_k \ldots)$ be the conditional teaching size of concept c_i, given the set of concepts $\{c_j, c_k \ldots\}$ previously distinguished by the learner. The challenge is to minimise $TS(c_1) + TS(c_2|c_1) + TS(c_3|c_1, c_2) + \ldots$.

In this new setting we need a definition of $TS(c_i|c_j)$ that considers that (1) a concept c has infinitely many programs that generate it, so which one the learner has identified may be important, and (2) the learner must have some *memory*, where that program is stored. Interestingly, if we assume that memory is implemented by storing the identified programs in a library, where the learner can only make calls to —but not reuse its parts—, then it is irrelevant which program has been used to capture concept c, since the learner only reuses the functional *behaviour* of the program[7].

4.1 Conditional teaching size and minimal curriculum

We define a library $B = \{p_1, \ldots, p_k\}$, as a set of programs in the universal language used by the learner. Let $|B| = k$ the number of *primitives*. We assume that Υ always includes an instruction @ for making static[8] library calls. We use @i to denote the instruction that calls the primitive that is indexed as i in the library. If $|B| = 1$, then @ needs no index. Accordingly, the length of a call to the library is $\ell(@\mathsf{i}) = \ell(@) + \log_2(|B|) = \log_2(|\Upsilon|) + \log_2(|B|)$ bits.

Let p, p' be programs in the universal language L and B a library. We say that a program p *contains a call to* p' when @i is a substring of p and i is the index of $p' \in B$. L_B denotes a language L that implements static calls to a library B. Even with static calls, the flow of the program may never reach @ for an input. Interestingly, we can avoid this undecidable question when dealing with programs in the teaching book by considering @ as the last instruction regarding lexicographical order.

Lemma 1. Let f be a complexity function and B a library. For any $(w, p) \in$ f-Teaching Book, if p has a call to B then p effectively reaches @ and executes a primitive on at least one input of w.

Let us use $\dot{p}$ to denote program @i, where i is the index of p in the library.

[7] If the learner uses a complexity function f, then we may have that a particular program p_1 identifies c_1 and c_1 is very useful for c_2, but p_1 is too slow to be used in any reasonably efficient program for c_2, so becoming useless incrementally. Computational time has also been considered in other machine teaching frameworks [21,43].

[8] There is no loss of generality here, since every program that uses dynamic calls can be rewritten only using static calls [1].

Lemma 2. Let B be a library. The language L_B satisfies: $\dot{p} \prec p',\ \forall p'$ such that $p' \notin [p]_L \wedge p'$ has a call to p.

Now, we are able to redefine the learner, Φ^{f}_{ℓ}, and the time-bounded Kolmogorov complexity for a given library.

Definition 1. Let f be a complexity function, B a library and S an example set. The learner Φ calculates the *first program* for S in language L_B:

$$\Phi^{\mathsf{f}}_{\ell}(S|B) = \arg\min_{p \in L_B}{}^{\prec} \{\ell(p) : p \vDash_{\mathsf{f}} S\}$$

The f-bounded Kolmogorov complexity of S, denoted by $K^{\mathsf{f}}(S|B)$, is the length of the program returned by the learner: $K^{\mathsf{f}}(S|B) = \ell\left(\Phi^{\mathsf{f}}_{\ell}(S|B)\right)$. The extension of the teacher, denoted by $\Omega^{\mathsf{f}}_{\ell}(c|B)$, also selects the shortest witness set that makes the learner distinguish the concept:

$$\Omega^{\mathsf{f}}_{\ell}(c|B) = \arg\min_{S}{}^{\preccurlyeq} \left\{\delta(S) : \Phi^{\mathsf{f}}_{\ell}(S|B) \in [c]_{L_B})\right\}$$

And the definition of the K^{f}-teaching size of a concept c is $TS^{\mathsf{f}}_{\ell}(c|B) = \delta(\Omega^{\mathsf{f}}_{\ell}(c|B))$.

We can also extend Theorem 1 in [38].

Corollary 1. Let L be a universal language and B a library. For any concept c in C_{L_B}, there is a complexity function f so that the f-Teaching Book will contain some (p_c, w_c) with $p_c \in [c]_{L_B}$ and $TS^{\mathsf{f}}_{\ell}(c|B) = \delta(w_c)$.

Sometimes we will refer to the original L OR the augmented L_B depending on whether we see it conditional to B or not. We are now in position to give a formal definition of the conditional teaching size given a set of concepts.

Definition 2. Let $a \in C_L$, $\{c_i\}_{i=1}^n \subset C_L$ and let $p_i = \Phi(\Omega^{\mathsf{f}}_{\ell}(c_i))$, for each $i = 1, \ldots, n$. Let $B = \{p_i\}_{i=1}^n$. We define the conditional teaching size of concept a given the concepts $\{c_i\}_{i=1}^n$, denoted by $TS^{\mathsf{f}}_{\ell}(a|c_1, \ldots, c_n)$, as

$$TS^{\mathsf{f}}_{\ell}(a|c_1, \ldots, c_n) = TS^{\mathsf{f}}_{\ell}(a|B)$$

The programs that identify the concepts are in the same f-Teaching Book.

We now give a definition of *curriculum*. Given a set of concepts, a curriculum is a set of disjoint sequences covering all the concepts. Our notion of curriculum is more general than just a simple sequence. If some branches are unrelated, a curriculum should not specify which branch comes first, and are considered independent 'lessons'. We will see how this flexibility is handled by the algorithm that finds the optimal curriculum in section 5. For instance, Fig. 2 shows how a set of concepts $\{a, b, c, d, e, f, g\}$ is partitioned into three branches: $\{a \to b \to c \to d, e \to f, g\}$, where $a \to b$ means that b must come after a in the curriculum. For each *branch*, there is no background knowledge or library at the beginning. The library grows as the teacher-learner protocol progresses in each branch.

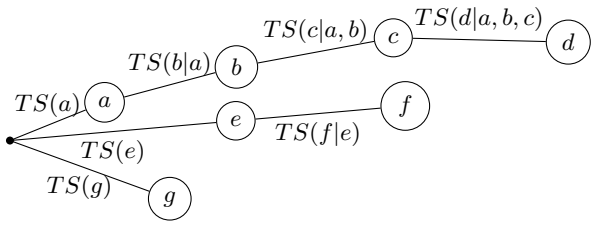

Fig. 2. Curriculum $\{a \to b \to c \to d, e \to f, g\}$ for a set of concepts $\{a, b, c, d, e, f, g\}$.

Definition 3. Let $Q = \{c_i\}_{i=1}^{n}$ a set of n labelled concepts. A curriculum $\pi = \{\sigma_1, \sigma_2, \cdots, \sigma_m\}$ is a full partition of Q where each of the m subsets $\sigma_j \subset Q$ has a total order, becoming a sequence.

We denote $\overline{Q}$ as the set of all the curricula in Q. The order in which the subsets are chosen does not matter, but the order each subset is traversed does. For example, the curriculum $\pi = \{a \to b \to c \to d, e \to f, g\}$ can have many paths, such as $abcdedfg$ or $gabcdef$. But note that π is different from $\pi' = \{b \to a \to c \to d, f \to e, g\}$. It is easy to check that, for any Q with n concepts, the number of different curricula is $|\overline{Q}| = n! \cdot \left(\sum_{k=0}^{n-1} \binom{n-1}{k} \cdot \frac{1}{(k+1)!}\right)$.

In what follows we will consider that all concepts are in the original f-Teaching Book, so they can be taught independently. This is not an important constraint, given Theorem 1 in [38] and Corollary 1. With this we ensure the same f for all of them. Now we can define the teaching size of a curriculum:

Definition 4. *Let* f *be a complexity function and let* Q *be a set of concepts that appear in the original* f*-Teaching Book. Let* $\pi = \{\sigma_1, \sigma_2, \cdots, \sigma_m\}$ *a curriculum in* Q*. We define the teaching size of each sequence* $\sigma = \{c_1, c_2, ..., c_k\}$ *as* $TS_\ell^{\mathsf{f}}(\sigma) = TS_\ell^{\mathsf{f}}(c_1) + \sum_{j=2}^{k} TS_\ell^{\mathsf{f}}(c_j|c_1, \ldots, c_{j-1})$*. The overall teaching size of* π *is just* $TS_\ell^{\mathsf{f}}(\pi) = \sum_{i=1}^{m} TS_\ell^{\mathsf{f}}(\sigma_i)$*.*

We say that a curriculum in Q is minimal, denoted by π^*, if no other has less overall teaching size.

4.2 Interposition and non-monotonicity

We now show a teaching phenomenon called *interposition*: new acquired concepts may lead to an increase in teaching size. The phenomenon might not even preserve the relationship established between two concepts, in terms of conditional Kolmogorov complexity, when considering conditional teaching size.

Definition 5. We say that B is an *interposed* library for concept c if $TS(c|B) > TS(c)$; if $B = \{p'\}$ we say that p' is an interposed program for c.

Proposition 2. For any $(w_c, p_c) \in$ f-Teaching Book, such that $@ \prec p_c$, there is an interposed library for concept c.

The above proposition means that virtually every concept (represented in the teaching book by a program of more than one instruction) may be interposed by

a primitive that makes the witness set lead to another concept. Not only may some concepts be useless for the concepts yet to come in the curriculum, but that they may even be harmful. This will have important implications when we look for minimal curricula in the following section.

This contrasts with conditional Kolmogorov complexity, where for every a and b we have that $K(a|b) \leq K(a)$. Given this, we can study the monotonicity between concept complexity and teaching size. Namely, *is there any relationship between* $K(a|b) \leq K(b|a)$ *and* $TS(a|b) \leq TS(b|a)$*?* We now show that, for any universal language, the inequalities aforementioned have, in general, different directions. First, we give the following definition.

Definition 6. Let $c \in C_L$ and let B be a library. We define the Kolmogorov conditional complexity of a concept c given a library B as $K_{L_B}(c) = \ell(p_c^*)$ where p_c^* is calculated using L_B. We use the notation $K(c|B) = K_{L_B}(c)$.

We now extend the conditional Kolmogorov complexity to a set of concepts through programs that identify the concepts given in the same f-Teaching Book:

Definition 7. Let $a \in C_L$, the set $\{c_i\}_{i=1}^n \subset C_L$ and $p_i = \Phi(\Omega_\ell^{\mathsf{f}}(c_i))$, for each $i = 1, \ldots, n$. Let $B = \{p_i\}_{i=1}^n$. We define the Kolmogorov complexity of concept a given the concepts $\{c_i\}_{i=1}^n$, denoted by $K(a|c_1, \ldots, c_n)$, as

$$K(a|c_1, \ldots, c_n) = K(a|B)$$

In words, the conditional complexity of a concept given a set of concepts is equal to the conditional complexity of the concept given the canonical programs for those concepts as extracted from the original teaching book.

We now show the non-monotonicity between K and TS:

Theorem 2. There exist two concepts a, $b \in C_L$ and a complexity function, f, such that $K(a|b) < K(b|a)$ and $TS_\ell^{\mathsf{f}}(a|b) > TS_\ell^{\mathsf{f}}(b|a)$.

When considering conditional teaching size for curriculum learning, we need general conditions to avoid interposition. For instance, an important reduction of program size in language L_B usually minimises the risk of interposition.

Corollary 2. Let $(w_c, p_c) \in \mathsf{f}$-Teaching Book, with $p_c \in [c]_L$. If there exists a library B and a witness set w, verifying the following conditions (1) $\delta(w) < \delta(w_c)$ and (2) the first program $p'_c \in [c]_{L_B}$, using order $\prec$, such that $p'_c \vDash_{\mathsf{f}} w$, precedes any other program p in language L_B, satisfying $p \vDash_{\mathsf{f}} w$, then $TS_\ell^{\mathsf{f}}(c|B) < TS_\ell^{\mathsf{f}}(c)$.

These conditions to avoid interposition are strong, since we shall elucidate, e.g., whether a program is the shortest one, using a time complexity bound f.

5 Minimal curriculum: Interposition range and $\mathbb{I}$-search

One key reason why interposition is hard to avoid is the existence of programs (and concepts) with *parallel behaviour*, i.e., programs with equal inputs-outputs

up to large sizes of the inputs, e.g., one implementing the even function, and the other doing the same except for the input 2^{300}. However, in practice, the concepts we use in the break-out for a curriculum do not have this problem. For instance, we can use addition to teach multiplication. They coincide in a few cases, $2+2=4$ and $2\times 2=4$, but they clearly differ in many other short inputs.

Thus, let a, b be distinct concepts such that $\exists(w_a, p_a), (w_b, p_b) \in \mathsf{f}-$Teaching Book, with p_a, p_b in L verifying $w_a \nvDash_{\mathsf{f}} p_b$ and $w_b \nvDash_{\mathsf{f}} p_a$. Assume that we use w_a first and the learner outputs p_a, and adds it to $B = \{p_a\}$. With this increased L_B, if we give w_b to the learner, it does not output p_a since $p_a \nvDash_{\mathsf{f}} w_b$. However, there might still be interposition. For instance, suppose that L has four instructions: x, y, z and t. Let $B = \{\mathsf{xx}\}$ and suppose that $p_b = \mathsf{zytxz}$ is f-compatible with w_b. Suppose that there exists $p = \mathsf{xxytxx}$, expressed as $p = \mathsf{@yt@}$ in L_B, such that $p \vDash_{\mathsf{f}} w_b$. Program p would interpose to p_b. It would be important to know about such programs p, i.e., the ones that precede p_b in L_B and are posterior in L.

5.1 Interposition range: $\mathbb{I}$-sets

Firstly, we define the set of *interposed programs*.

Definition 8. Let w be a witness set and B be a library. Let p be a program in language L_B such that $p \vDash_{\mathsf{f}} w$. We define the $\mathbb{I}$-set of *interposed programs* in language L_B for p and w as $\mathbb{I}^{\mathsf{f}}_{w}(p|B) = \{q \text{ in } L_B : q \vDash_{\mathsf{f}} w \text{ and } q \prec p\}$.

We now show how large the $\mathbb{I}$-sets can be. To do that, we use the size of a program when its library calls are *unfolded*, i.e., given a program p and a library B, we use $\circ(p)$ to denote the program that is equivalent to p (as it worked in L_B) , where each primitive call @ has been replaced by the instructions of the called primitive in B.

Given an $\mathbb{I}$-set, we call *size-range*, denoted as $[i_{min}, i_{max}]$, to the range of $i = \dot{\ell}(\circ(q))$, $\forall q \in \mathbb{I}$-set. The *call-range*, denoted as $[j_{min}, j_{max}]$, is the range of the number of library calls, j, $\forall q \in \mathbb{I}$-set. We call *s/c-ranges* to both ranges; interposition occurs within them. The following theorem gives the *s/c-ranges* explicitly and provides a bound for the cardinality of the $\mathbb{I}$-set.

Theorem 3. Let (w_a, p_a), $(w_b, p_b) \in \mathsf{f}$-Teaching Book, with p_a, p_b in L and $p_a \nvDash_{\mathsf{f}} w_b$. Consider the library $B = \{p_a\}$. Let p'_b an equivalent program to p_b for L_B. Then, the cardinal of $\mathbb{I}^{\mathsf{f}}_{w_b}(p'_b|B)$ is bounded by $\sum_i \left(\sum_j \binom{i-\dot{\ell}(p_b)\cdot j+j}{j} \cdot (|\Upsilon|-1)^{(i-j\cdot\dot{\ell}(p_b))} \right)$ with i, $j \in \mathbb{N}$ ranging in the intervals: (1) $i_{min} = \dot{\ell}(p_b)$, $i_{max} = 1 + (\dot{\ell}(p'_b) - 1) \cdot \dot{\ell}(p_a)$, $j_{min} = \lceil \frac{i-\dot{\ell}(p'_b)}{\dot{\ell}(p_a)-1} \rceil$ and $j_{max} = \lfloor \frac{i}{\dot{\ell}(p_a)} \rfloor$, when $1 < \dot{\ell}(p_a) < \dot{\ell}(p_b)$; (2) $i_{min} = \dot{\ell}(p_a)+1$ and the rest is as (1), when $\dot{\ell}(p_a) \geq \dot{\ell}(p_b)$.

Could we identify an empty $\mathbb{I}$-set, based just on the sizes of the programs involved? It happens when the *s/c-ranges* define an *empty region*. In Theorem 3 (1), it occurs whenever $i_{max} < i_{min}$. Namely, we have $\mathbb{I}^{\mathsf{f}}_{w_b}(p_b{}'|B) = \emptyset$, when:

$$\dot{\ell}(p_b) > 1 + (\dot{\ell}(p'_b) - 1) \cdot \dot{\ell}(p_a) \tag{2}$$

For instance, if $\dot{\ell}(p_a) = 4$, $\dot{\ell}(p_b) = 8$ and we know that $\dot{\ell}(p_b{}') = 2$, then $i_{min} = 8$ and $i_{max} = 1 + (2-1) \cdot 4 = 5$. We see that this becomes more likely as p_b is much greater than p_a and the program for b using B, i.e., $p_b{}'$, is significantly reduced by the use of $B = \{p_a\}$.

Let p' be the first program in $[b]_{L_B}$ such that $p' \vDash_{\mathsf{f}} w_b$. With the conditions of Theorem 3 (1), p' must be equivalent to p_b and operating with Eq. 2 we get $\dot{\ell}(p') < \frac{\dot{\ell}(p_b)-1}{\dot{\ell}(p_a)} + 1$, which means there is no interposition for any program for b by including $B = \{a\}$ and $TS^{\mathsf{f}}_{\ell}(b|a) \leq TS^{\mathsf{f}}_{\ell}(b)$. But, since $\ell(p'_b) \geq K(b|a)$ we also have that Eq. 2 is impossible when $K(b|a) \geq (\frac{\dot{\ell}(p_b)-1}{\dot{\ell}(p_a)} + 1) \cdot \log_2 |\Upsilon|$.

We now consider a library with more than one primitive. We cannot extend Theorem 3 as a Corollary, since the relationships involved change completely, but we can connect both cases through the *s/c-ranges*.

Theorem 4. Let $\{(w_m, p_m)\}_{m=1}^{n}$, $(w_c, p_c) \in$ f-Teaching Book, with p_c, p_m in L, $\forall m$. Consider $B = \{p_m\}_{m=1}^{n}$ with $p_m \nvDash_{\mathsf{f}} w_c$, $\forall m$, and $1 < |B|$. Let $p_c{}'$ be an equivalent program to p_c for L_B. Let $D, r \in \mathbb{N}$ such that $\ell(p'_c) = D \cdot \ell(@\mathsf{i}) + r$, i.e., they are the *divisor* and the *remainder* of the division $\ell(p'_c)/\ell(@\mathsf{i})$. Note that $\ell(@\mathsf{i}) = \log_2 |\Upsilon| + \log_2 |B|$. Let $p_{max} = \max^{\prec}\{p_m\}_1^n$ and $p_{min} = \min^{\prec}\{p_m\}_1^n$. Then, the cardinal of $\mathbb{I}^{\mathsf{f}}_{w_c}(p_c{}'|B)$ is bounded by $|B| \cdot \sum_{s=2}^{\dot{\ell}(p'_c)} \left(\sum_{t=1}^{s} (|\Upsilon| - 1)^{s-t} \cdot |B|^{t-1} \right)$ and the *s/c-intervals* are: (1) if $1 < \dot{\ell}(p_{min}) \leq \dot{\ell}(p_c)$, then $i_{min} = \dot{\ell}(p_c)$, $i_{max} = D \cdot \dot{\ell}(p_{max}) + \lfloor r / \log_2 |\Upsilon| \rfloor$, $j_{min} = \lfloor \frac{\dot{\ell}(p'_c) - \dot{\ell}(\circ(q))}{\dot{\ell}(@\mathsf{i}) - \dot{\ell}(p_{max})} \rfloor$ and $j_{max} = \min\{D, \lfloor \frac{\dot{\ell}(\circ(q))}{\dot{\ell}(p_{min})} \rfloor\}$; (2) if $\dot{\ell}(p_c) < \dot{\ell}(p_{min})$, then $i_{min} = \dot{\ell}(p_{min}) + 1$ and the rest is as in (1).

We need $D \cdot \dot{\ell}(p_{max}) + \lfloor r / \log_2 |\Upsilon| \rfloor < \dot{\ell}(p_{min}) + 1$, to avoid interposition directly, in the same conditions as in Theorem 4 (1). It entails $\ell(p_c{}') < \ell(@\mathsf{i})$ when $\lfloor r / \log_2 |\Upsilon| \rfloor = 0$ in the extreme case. For Theorem 4 (2), an unfeasible *s-range* implies $D < \frac{\dot{\ell}(p_c) - \lfloor r / \log_2 |\Upsilon| \rfloor}{\dot{\ell}(p_{max})}$, which is restrictive.

5.2 Teaching size upper bounds: $\mathbb{I}$-safe

In practice, we deal with a program p that has the desired behaviour for a given witness set, but there may be interposition. If we know which the interposed programs are, then it is possible to get an upper bound of the teaching size of the concept that defines p, by *deflecting* interposition, refining the witness sets.

We employ $\mathbb{I}$-*safe* witnesses: example sets attached to input/output pairs. For instance, if we want to teach exponentiation, a set of examples might be $\{(3,1) \to 3, (2,2) \to 4\}$. This witness set is compatible with exponentiation, but also compatible with multiplication. To avoid multiplication being interposed, we can add another example to distinguish both concepts: $\{(3,1) \to 3, (2,2) \to 4, (2,3) \to 8\}$. We can always replace the original witness set by an $\mathbb{I}$-safe witness set, where, in general, we need to add examples to avoid interposition.

Proposition 3. Let f be a complexity function and (w, p), $\{(w_m, p_m)\}_{m=1}^{n} \in$ f-Teaching Book, with p, p_m in L, $\forall m$. Let $B = \{p_m\}_{m=1}^{n}$ be a library such that

$p_m \not\models_{\mathtt{f}} w$, $\forall m$. Let $c \in C_L$ such that $c \vDash w$. Let $p'_c \in [c]_{L_B}$ be the first program, using order $\prec$, such that $p'_c \vDash_{\mathtt{f}} w$. If $n = |\mathbb{I}^{\mathtt{f}}_w(p'_c|B)|$, there exist $\{\langle \mathtt{i}_k, \mathtt{o}_k\rangle\}_{k=1}^n$ such that $TS^{\mathtt{f}}_\ell(c|B) \leq \delta\big(w \bigcup_{k=1}^n \{\langle \mathtt{i}_k, \mathtt{o}_k\rangle\}\big)$.

For a library B, if we find an example set w that can be converted into an $\mathbb{I}$-safe witness set $\overline{w} = w \bigcup_{k=1}^n \{\langle \mathtt{i}_k, \mathtt{o}_k\rangle\}$ with $\delta(\overline{w}) < TS^{\mathtt{f}}_\ell(c)$ using B, then we reduce the teaching size. This is a sufficient and necessary condition to avoid interposition and get $TS^{\mathtt{f}}_\ell(c|B) \leq TS^{\mathtt{f}}_\ell(c)$.

Finally, given these general bounds: *how can we find minimal curricula?* Let us consider, for example, the set of concepts $Q = \{a, b\}$, where (w_a, p_a) and (w_b, p_b) are in the $\mathtt{f}$-Teaching Book. We also know that their behaviours are not parallel, i.e., $p_a \not\models_{\mathtt{f}} w_b$ and $p_b \not\models_{\mathtt{f}} w_a$. There are three different curricula $\{a, b\}$, $\{a \to b\}$ or $\{b \to a\}$. There is an $\mathbb{I}$-safe witness set $\overline{w}$, such that $\delta(\overline{w}) \leq TS^{\mathtt{f}}_\ell(b|a)$ (or $\delta(\overline{w}) \leq TS^{\mathtt{f}}_\ell(a|b)$). Thus, we can choose a curriculum, with less overall teaching size than the non-incremental version.

5.3 Minimal curriculum algorithm: $\mathbb{I}$-search

We now *search* minimal curricula. For example, let $Q = \{c_+, c_\times\}$ be a set of two concepts from Fig. 1, which appear in the non-incremental $\mathtt{f}$-Teaching Book as (w_+, p_+) and $(w_\times, p_\times)$. The set of possible curricula, $\overline{Q}$, is $\pi_0 = \{c_+, c_\times\}$, $\pi_1 = \{c_+ \to c_\times\}$ and $\pi_2 = \{c_\times \to c_+\}$.

The starting point for our algorithm will be π_0, the non-incremental curriculum, and its overall teaching size $TS^{\mathtt{f}}_\ell$. Then, we generate another curriculum: π_1. We know $TS^{\mathtt{f}}_\ell(c_+) = \delta(w_+)$ and we need to add $TS^{\mathtt{f}}_\ell(c_\times|c_+)$. We compare this total size to the best TS so far. We explore all the curricula in $\overline{Q}$ but, in order to save computational steps, we generate successive witness sets w_k, using order $\lessdot$, such that $c_\times \vDash w_k$ (Fig. 3).

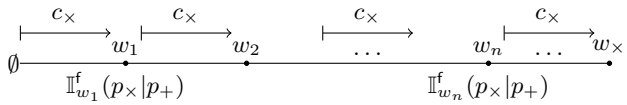

Fig. 3. Non-decreasing sequence of witness sets w_k, through $c_\times$ with $\delta(w_k) \leq \delta(w_\times)$.

For each w_k, we get the first program p_k of $\mathbb{I}^{\mathtt{f}}_{w_k}(p_\times|p_+)$. We then investigate whether $p_k \in [p_\times]_{L_B}$ or not. If p_k acts like $p_\times$ to certain witness size limit, H, then we can identify p_k and $p_\times$. The $\mathbb{I}$-*search algorithm* (5.3) shown below extends this strategy.

Note that the *s/c-ranges* reduce, *drastically*, the computational effort of executing the teacher-learner protocol (calculating teaching book and TS).

Algorithm: $\mathbb{I}$-search
Input: $Q = \{a, b, \ldots\}$; f-Teaching Book (w_a, p_a), (w_b, p_b)...; Witness size limit H

1. **For each** distinct pair of concepts $\langle x, y\rangle \in Q \times Q$**:**
   (a) **If** $[TS^{\mathsf{f}}_{\ell}(y|x) \leq TS^{\mathsf{f}}_{\ell}(y) \wedge TS(x|y)^{\mathsf{f}}_{\ell} \geq TS^{\mathsf{f}}_{\ell}(x)]$
   then $\overline{Q} = \overline{Q} \setminus \{\pi : \exists$ a branch starting as $y \to x\}$
2. $\pi^* = \{a, b, \ldots\}$, $TS^{\mathsf{f}}_{\ell}(\pi^*) = \sum_{x \in Q} TS^{\mathsf{f}}_{\ell}(x)$ and $\overline{Q} = \overline{Q} \setminus \{\pi^*\}$
3. **For each** $\pi \in \overline{Q}$**:**
   (a) $TS^{\mathsf{f}}_{\ell}(\pi) = 0$
   (b) **For each** branch $\sigma \in \pi$**:**
      i. **For each** concept $x \in \sigma$ (ordered by σ)**:**
         – $B = \{p_y : (y \in \sigma) \wedge (y$ precedes $x)\}$
         – Let p'_x be the first program equivalent to p_x in L_B, using order $\prec$
         – **For each** $w_k \in \{w \subset X : p'_x \vDash_{\mathsf{f}} w_k\}$, using order $\lessdot$**:**
            • **If** $[TS^{\mathsf{f}}_{\ell}(\pi^*) \leq TS^{\mathsf{f}}_{\ell}(\pi) + \delta(w_k)]$ **then break** to 3
            • $p = \min^{\prec}\{\mathbb{I}^{\mathsf{f}}_{w_k}(p'_x|B)\}$; use *s/c ranges* to refine the calculation
            • **If** $[p \vDash_{\mathsf{f}} w \longleftrightarrow p_x \vDash_{\mathsf{f}} w, \forall w$ such that $\delta(w) < H]$
            then $[\, TS^{\mathsf{f}}_{\ell}(\pi) = TS^{\mathsf{f}}_{\ell}(\pi) + \delta(w_k)$ **and break** to 3(b)i $]$
   (c) $\pi^* = \pi$ and $TS^{\mathsf{f}}_{\ell}(\pi^*) = TS^{\mathsf{f}}_{\ell}(\pi)$

Output: π^* and $TS^{\mathsf{f}}_{\ell}(\pi^*)$

In the previous example, e.g., if there is a w_n such that $TS^{\mathsf{f}}_{\ell}(c_\times|c_+) = \delta(w_n) < TS^{\mathsf{f}}_{\ell}(c_\times)$, then we set $\pi^* = \pi_1$ (and $TS^{\mathsf{f}}_{\ell}(\pi^*) = \delta(w_+) + \delta(w_n)$). Finally, we test π_2 and follow the same steps as with π_1. If, at some stage, there is a witness set w_m such that $TS^{\mathsf{f}}_{\ell}(c_\times) + \delta(w_m) \geq TS^{\mathsf{f}}_{\ell}(\pi^*)$, then π_1 is minimal and we stop.

The algorithm is complete but the search is not *exhaustive*, since we can *discard* curricula that contain a *branch* starting in a way that does not decrease the overall teaching size for sure. For example, if $TS^{\mathsf{f}}_{\ell}(c_\times|c_+) \leq TS^{\mathsf{f}}_{\ell}(c_\times)$ and $TS^{\mathsf{f}}_{\ell}(c_+|c_\times) \geq TS^{\mathsf{f}}_{\ell}(c_+)$, the branch $\sigma = \{c_+ \to c_\times \to c_\wedge\}$ has less or equal overall teaching size than $\sigma' = \{c_\times \to c_+ \to c_\wedge\}$. Consequently, we can remove all branches starting with $c_\times \to c_+$. We can test this for every pair of distinct concepts at the beginning of the branches.

The $\mathbb{I}$-search algorithm (5.3) satisfies the following theorem.

Theorem 5. Let H be certain witness size limit, f be a complexity function and Q be a set of concepts registered in the f-Teaching Book. We also assume, for each $c \in Q$, that $c \vDash w \to p_c \vDash_{\mathsf{f}} w$, $\forall w$ verifying $\delta(w) \leq \sum_{x \in Q} TS^{\mathsf{f}}_{\ell}(x)$. Then, the $\mathbb{I}$-search algorithm expressed in algorithm 5.3 returns a minimal curriculum.

The $\mathbb{I}$-search algorithm shows that: (1) We should create curricula containing concepts that significantly reduce the complexity of another ones. For instance, if concepts $c_\times$ and c_+ (Fig. 1) satisfy $K(c_\times|c_+) < K(c_\times)$, then the chances to minimise the teaching size increase significantly. (2) Given a set of concepts, it may be useful to implement some kind of *isolation* (or even forgetting by separating concepts in different branches[9]). For instance, c_{Θ} might be f-compatible with a considerable number of witness sets w_k and it may cause *interposition*

[9] Forgetting may simply refer to a lesson not using primitives that are considered out of the context of a "lesson".

with c_+, $c_\times$ or $c_\wedge$. This is why we should allocate c_θ in a different branch. (3) The branches (or lessons) could simply suggest ways in which we can arrange, *classify* and organise large sets of concepts. The tree-structure for curricula proposed here is a solution for the problem posed in [26].

6 Conclusions and future work

The teaching *size* —rather than teaching dimension— opened a new avenue for a more realistic and powerful analysis of machine teaching [38], its connections with information theory (both programs and examples can be measured in bits) and a proper handling of concept classes where examples and programs are compositional and possibly universal, such as natural language.

The intuitive concept of how much of the description of a concept is reused for the definition of another dates back to Leibniz's *règle pour passer de pensée en pensée* [18], and has been vindicated in cognitive science since Vigotsky's zone of proximal development [39,29], to more modern accounts of compositionality based on what has been learnt previously [24,22,30].

In mathematical terms, a gradient-based or continuous account of this view of incremental teaching, and the reuse of concepts, is not well accommodated. Incremental teaching is usually characterised as a compositional process, which is a more appropriate view for the acquisition of high-level concepts. The learning counterpart is still very elegantly captured by conditional Kolmogorov complexity, and some incremental learning schemata have followed this inspiration [17,13,31,20,23]. However, even if the concept of teaching *size* suggests that a mapping was possible, we have had to face a series of phenomena in order to translate some of these intuitions to the machine teaching scenario, and a new setting for curriculum teaching.

The absence of monotonicity because of interposition presents some difficulties for implementing curriculum teaching for compositional languages. Theorems 3 and 4 and its consequences make possible such an implementation: either through sufficient conditions to avoid interposition, by implementing $\mathbb{I}$-safe witness sets or through the $\mathbb{I}$-search.

Given the theoretical bounds and the algorithms for the optimal curricula, we can now start exploring novel algorithms and strategies for curriculum teaching that are suboptimal, but more efficient, such as (1) greedy algorithms introducing the next concept as the one with maximum local TS reduction, (2) approximations based on Vigotsky's zone of proximal development principles [39,29] where each step is bounded by some teaching length Z, i.e., such that $TS(c_{i+1}|c_1,\ldots,c_i) \leq Z, \forall i$; or (3) variations of the *incremental combinatorial optimal path* algorithm [32]. All these new research possibilities in curriculum teaching, and even others, are now wide open to exploration.

Because of the fundamental (re-)connection we have done between K and TS in this paper, another novel possibility for curriculum teaching would be the combination of teaching by examples *and* descriptions of the concepts themselves. This is actually the way humans teach other humans, combining examples and

descriptions, but it is nevertheless unprecedented in the application of machine teaching in natural language processing [25,33]. However, it is beginning to become common with language models, with prompts that combine examples and some indications of the task to perform [4,14].

Acknowledgements

This work was funded by the EU (FEDER) and Spanish MINECO under RTI2018-094403-B-C32, G. Valenciana under PROMETEO/2019/098 and EU's Horizon 2020 research and innovation programme under grant 952215 (TAILOR).

References

1. Antoniol, G., Di Penta, M.: Library miniaturization using static and dynamic information. In: International Conference on Software Maintenance. pp. 235–244 (2003)
2. Balbach, F.J.: Models for algorithmic teaching. Ph.D. thesis, U. of Lübeck (2007)
3. Balbach, F.J.: Measuring teachability using variants of the teaching dimension. Theoretical Computer Science **397**(1-3), 94–113 (2008)
4. Brown, T.B., Mann, B., Ryder, N., et al.: Language models are few-shot learners. arXiv: 2005.14165 (2020)
5. Cicalese, F., Laber, E., Molinaro, M., et al.: Teaching with limited information on the learner's behaviour. In: ICML. pp. 2016–2026. PMLR (2020)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv: 1810.04805 (2018)
7. Elias, P.: Universal codeword sets and representations of the integers. IEEE Transactions on Information Theory **21**(2), 194–203 (1975)
8. Gao, Z., Ries, C., Simon, H.U., Zilles, S.: Preference-based teaching. The Journal of Machine Learning Research **18**(1), 1012–1043 (2017)
9. Garcia-Piqueras, M., Hernández-Orallo, J.: Conditional teaching size. arXiv preprint p. 26 (2021)
10. Gong, C.: Exploring commonality and individuality for multi-modal curriculum learning. In: AAAI. vol. 31 (2017)
11. Gong, C., Yang, J., Tao, D.: Multi-modal curriculum learning over graphs. ACM Transactions on Intelligent Systems and Technology (TIST) **10**(4), 1–25 (2019)
12. Gong, T., Zhao, Q., Meng, D., Xu, Z.: Why curriculum learning & self-paced learning work in big/noisy data: A theoretical perspective. BDIA. **1**(1), 111 (2016)
13. Gulwani, S., Hernández-Orallo, J., Kitzelmann, E., Muggleton, S.H., Schmid, U., Zorn, B.: Inductive programming meets the real world. Comm. ACM **58**(11) (2015)
14. Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., Steinhardt, J.: Measuring massive multitask language understanding. ICLR (2021)
15. Hernández-Orallo, J., Telle, J.A.: Finite and confident teaching in expectation: Sampling from infinite concept classes. In: ECAI (2020)
16. Kumar, A., Ithapu, V.: A sequential self teaching approach for improving generalization in sound event recognition. In: ICML. pp. 5447–5457 (2020)
17. Lake, B.M., Salakhutdinov, R., Tenenbaum, J.B.: Human-level concept learning through probabilistic program induction. Science **350**(6266), 1332–1338 (2015)
18. Leibniz, G.W., Rabouin, D.: Mathesis universalis: écrits sur la mathématique universelle. Mathesis (Paris, France), Librairie philosophique J. Vrin (2018)

19. Li, M., Vitányi, P.M.: An Introduction to Kolmogorov Complexity and Its Applications. Springer Publishing Company, Incorporated, 3rd edn. (2008)
20. Li, Y., Mao, J., Zhang, X., Freeman, W.T., Tenenbaum, J.B., Wu, J.: Perspective plane program induction from a single image. In: CVPR. pp. 4434–4443 (2020)
21. Liu, W., Dai, B., Humayun, A., Tay, C., Yu, C., Smith, L.B., Rehg, J.M., Song, L.: Iterative machine teaching. In: ICML. p. 2149–2158 (2017)
22. Manohar, S., Zokaei, N., Fallon, S., Vogels, T., Husain, M.: Neural mechanisms of attending to items in working memory. Neur. & Biob. Rev. **101**, 1–12 (2019)
23. Nye, M.I., Solar-Lezama, A., Tenenbaum, J.B., Lake, B.M.: Learning compositional rules via neural program synthesis. arXiv: 2003.05562 (2020)
24. Oberauer, K., Lin, H.Y.: An interference model of visual working memory. Psychological review **124**(1), 21 (2017)
25. Peng, B., Li, C., Li, J., Shayandeh, S., Liden, L., Gao, J.: Soloist: Building task bots at scale with transfer learning and machine teaching. arXiv: 2005.05298 (2020)
26. Pentina, A., Sharmanska, V., Lampert, C.H.: Curriculum learning of multiple tasks. In: Proc. of Computer Vision and Pattern Recognition (June 2015)
27. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)
28. Rakhsha, A., Radanovic, G., Devidze, R., Zhu, X., Singla, A.: Policy teaching via environment poisoning: Training-time adversarial attacks against reinforcement learning. In: ICML. pp. 7974–7984 (2020)
29. Salkind, N.: An introduction to theories of human development. Sage P. (2004)
30. Schneider, W.X., Albert, J., Ritter, H.: Enabling cognitive behavior of humans, animals, and machines: A situation model framework. ZiF **1**, 21–34 (2020)
31. Shi, Y., Mi, Y., Li, J., Liu, W.: Concept-cognitive learning model for incremental concept learning. IEEE Trans. on Systems, Man, and Cybernetics: Systems (2018)
32. Shindyalov, I., Bourne, P.: Protein structure alignment by incremental combinatorial extension of the optimal path. Prot. Eng. Des. & Sel. **11**(9), 739–747 (1998)
33. Shukla, S., Liden, L., Shayandeh, S., Kamal, E., Li, J., Mazzola, M., Park, T., Peng, B., Gao, J.: Conversation learner–a machine teaching tool for building dialog managers for task-oriented dialog systems. arXiv: 2004.04305 (2020)
34. Solomonoff, R.J.: A formal theory of inductive inference I. IC **7**(1), 1–22 (1964)
35. Solomonoff, R.J.: A system for incremental learning based on algorithmic probability. In: Proc. Sixth Israeli Conf. AICVPR. pp. 515–527 (1989)
36. Soviany, P., Ionescu, R.T., Rota, P., Sebe, N.: Curriculum learning: A survey. arXiv: 2101.10382 (2021)
37. Such, F.P., Rawal, A., Lehman, J., Stanley, K., Clune, J.: Generative teaching networks: Accelerating neural architecture search by learning to generate synthetic training data. In: ICML. pp. 9206–9216 (2020)
38. Telle, J.A., Hernández-Orallo, J., Ferri, C.: The teaching size: computable teachers and learners for universal languages. Machine Learning **108**, 1653–1675 (2019)
39. Vygotsky, L.S.: Mind in Society: Development of Higher Psychological Processes. Harvard University Press (1978)
40. Weinshall, D., Cohen, G., Amir, D.: Curriculum learning by transfer learning: Theory and experiments with deep networks. In: ICML. pp. 5235–5243 (2018)
41. Zhou, T., Bilmes, J.A.: Minimax curriculum learning: Machine teaching with desirable difficulties and scheduled diversity. In: ICLR (Poster) (2018)
42. Zhu, X.: Machine teaching: An inverse problem to machine learning and an approach toward optimal education. In: AAAI. pp. 4083–4087 (2015)
43. Zhu, X., Singla, A., Zilles, S., Rafferty, A.: An overview of machine teaching. arXiv: 1801.05927 (2018)